Buskeeper PUFs, a Promising Alternative to D Flip-Flop PUFs

Peter Simons
Intrinsic-ID
Eindhoven, The Netherlands
peter.simons@intrinsic-id.com

Erik van der Sluis
Intrinsic-ID
Eindhoven, The Netherlands
erik.van.der.sluis@intrinsic-id.com

Vincent van der Leest
Intrinsic-ID
Eindhoven, The Netherlands
vincent.van.der.leest@intrinsic-id.com

***Abstract*—Cloning, theft of service and tampering have become serious threats on 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, so the strength of the security depends on the effort required from attackers to compromise them. Tools for attacking hardware have become very advanced, which has decreased the protection provided by storing a key in memory to a minimum. To protect devices against attacks on their keys, Physically Unclonable Functions (PUFs) can be used. PUFs are primitives that extract secrets from physical characteristics of integrated circuits (ICs) and can be used, amongst others, for secure key storage.**

This paper introduces a new type of PUF, the Buskeeper. In our study this new type of PUF is evaluated on the properties of reliability and uniqueness. For this purpose several tests have been performed in order to compare the results of Buskeeper PUFs to those of D Flip-Flop (DFF) PUFs from [4] and [14]. This comparison shows that the Buskeeper PUF performs as well as, if not better than, this (already known and generally accepted) PUF type. Since Buskeepers are much more efficient than DFFs in regard to the amount of hardware resources required, we conclude that the Buskeeper PUF is a viable (and probably preferable) alternative to DFF PUFs.

I. Introduction

Due to submicron process variations during manufacturing, every transistor in an integrated circuit (IC) has slightly different physical properties, which are measurable. Since these process variations are uncontrollable (even for the manufacturer), the physical properties of a device cannot be copied or cloned. Therefore, it is very difficult and economically not viable to create a device with a selected electronic fingerprint. PUF implementations require an electronic circuit that measures the responses of hardware to specifically provided inputs. These responses depend on the unique and uncontrollable physical properties of the device. This way PUFs are functions that are easy to challenge and whose responses are easy to measure, but also are very hard to clone due to their physical structures.

A. Related Work

Pappu [12] introduced the concept of PUFs in 2001 using 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 this principle was translated by Gassend et al. [6] into Silicon Physical Random Functions. These functions make use of the manufacturing process variations in ICs, with identical masks, to uniquely characterize each IC. For this purpose ring oscillators were supplied with frequency measuring circuitry. Based on measurements from these circuits, the statistical delay variations of transistors and wires in the IC were used to characterize ICs. This method of characterization is now known as a Ring Oscillator PUF. In 2004 Lee et al. [9] proposed another PUF that is based on delay measurements, the Arbiter PUF.

Besides these PUFs based on delay measurements a second hardware intrinsic type 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 [7]. Furthermore, so-called Butterfly PUFs were introduced in 2008 by Kumar et al. [8] and D Flip-Flop PUFs (also in 2008) were proposed by Maes et al. [11]. Implementations of the hardware intrinsic PUF types, as described here, exist for dedicated ICs, programmable logic devices such as Field Programmable Gate Arrays (FPGAs) and also for programmable ICs (such as microcontrollers).

B. Our Contribution

In this paper a known cell structure, the Buskeeper, is evaluated based on its potential as a PUF. To the knowledge of the authors the Buskeeper has never before been considered as a PUF. The results in this paper demonstrate that the Buskeeper is very suitable for use as a PUF and should even be considered as a replacement for the generally known DFF PUFs. This is because the Buskeeper PUF properties are comparable to those of DFF PUFs from literature, while the hardware comparison of the two types is in favour of the Buskeeper PUF (see section IV). Therefore, the authors conclude that this paper successfully introduces the Buskeeper PUF as a valuable security primitive.

C. Paper Outline

This paper is constructed as follows: Section II describes how Buskeepers can be used as PUFs. The tests that have been performed to evaluate the PUF behaviour (including their results) can be found in section III. In section IV the results of Buskeeper PUFs are compared to results of DFF PUFs from literature. Finally, the paper is concluded in section V.

II. Buskeepers as PUFs

A. The Buskeeper

A Buskeeper (which can be seen in Figure 1), also known as bus holder, is a weak Latch that usually has no control signals. It is intended to be used with on-chip buses that have multiple drivers (see Figure 2). Depending on the system these buses could become floating, which increases the power consumption of the chip. To prevent that, the Buskeeper is added which maintains the last driven state of the bus. Because it has a low strength the bus drivers can override the Buskeeper's output when it has to send data over the bus.

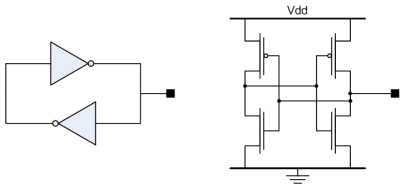

Fig. 1. High-level Buskeeper cell picture (left) and transistor level (right).

The Buskeeper is functionally equivalent to a D-latch with the enable signal connected to Vdd. Most standard cell libraries include this element. The big advantage over using a Latch or a DFF for constructing a PUF is that the Buskeeper is very small (due to its low drive strength and lack of control signals), usually about 1 GE[1], where Latches are usually more than 4 GE and DFFs are usually between 6 and 8 GE.

When using Buskeepers in a memory-based PUF construction the most likely way to read out their start-up pattern is using a mux tree. This method is also often used for Latch and DFF PUFs.

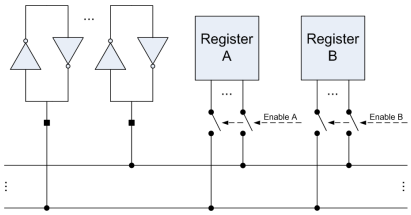

Fig. 2. Buskeepers in an example of their regular implementation.

B. PUFs for Secure Key Storage

It becomes clear from the description of Buskeeper cells that when used as a PUF, Buskeepers belong to the category of memory-based PUFs. A very common purpose for this type of PUF is its use in secure key storage implementations [13]. Therefore will this section provide a description of this suitable application for Buskeeper PUFs.

In secure key storage we distinguish two phases (see also Figure 3): Enrollment and Reconstruction.

[1]GE - Gate Equivalent is a measure of area in any technology. 1 GE is the area of a NAND2 (standard drive strength).

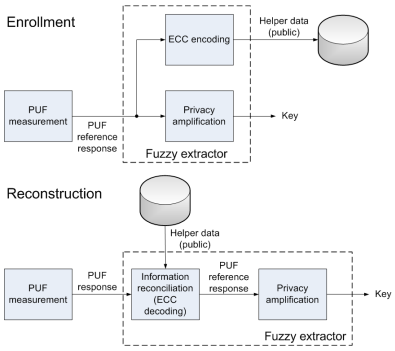

Fig. 3. Enrollment and Key Reconstruction for the described PUF model.

1) 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, the response of the targeted PUF is measured. This response is called the reference PUF response and is the input of the Fuzzy Extractor [3], [5], [10]. The Fuzzy Extractor (FE) 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.

2) 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).

C. PUF Properties

In order to be able to use PUFs in security applications, like the described secure key storage (or, for example, as unique device identifiers), they should possess two properties that determine the quality of these PUFs. These properties are reliability and uniqueness, which are described in detail below.

1) Reliability: The first important property for PUFs is reliability. Defining reliability we say that for any device, when a PUF response is measured during "Reconstruction", the FE should 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 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. In this paper, we show the reliability of Buskeeper PUFs by performing these tests:

- Temperature Variation Test. To study the reliability of PUF responses at varying ambient temperatures, they are measured while the devices are subjected to temperatures varying from -40°C to +85°C.
- Voltage Variation Test. In order to evaluate the effect the level of the supply voltage has on PUF responses, the devices are subjected to levels varying from 90% to 110% of the nominal Vdd.
- Voltage Ramp-up Test. The time required for a supply voltage to rise from 0V to Vdd can also influence the stability of PUF responses. Therefore PUF responses are measured while varying this time.
- Ageing Test. It is known that silicon degrades over time, which has repercussions on PUFs. The most important mechanism responsible for ageing of memory-based PUFs is Negative Bias Temperature Instability (NBTI). In this test NBTI is simulated by stressing devices at high temperature with high supply voltage for a long time.

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.

In order to assess the uniqueness of the Buskeeper PUFs we perform the following tests:

- Between-class Distribution Test. To find out whether it is possible to identify Buskeeper PUF responses individually a Between-Class Distribution can be used. Using this distribution it is possible to find out whether responses from different PUFs are sufficiently different from each other to be able to distinguish between them.
- Entropy Estimation. For estimating entropy PUF responses can be compressed (to find an upper bound) and their min-entropy can be calculated (lower bound). The actual entropy of PUFs will be somewhere between these two boundaries.

III. Test Results

A. ASIC and Environment

In order to evaluate the properties of Buskeeper PUFs an ASIC has been used, which has been developed in the (EU-funded) FP7 project UNIQUE. This ASIC has been designed by different partners from this project and was produced through IMEC/Europractice at TSMC on a 65nm Multi Project Wafer (MPW). The ASIC contains two identical Buskeeper PUF instantiations of 1kB each.

When testing the reliability of the Buskeeper PUFs environmental changes in temperature and power supply are required. For this purpose a climate chamber and programmable power supply have been used.

B. Reliability Tests

1) Temperature Variation Test: To test the reliability of Buskeeper PUFs under temperature variations 96 ICs (with two Buskeeper PUFs of 1kB) have been placed in the climate chamber. This way a set of 192 Buskeepers can be evaluated. Measurements of PUF start-up patterns have been taken at three different temperatures: -40°C, +25°C, and +85°C (industrial standard for temperature testing of ICs ranges from -40°C to +85°C). In this case +25°C is the enrollment temperature of the PUFs, while the other two temperatures are the most extreme deviations from enrollment available for this test. At each temperature the Buskeeper PUFs have been measured 40 times. These measurements are all compared to one enrollment pattern for each PUF at +25°C using fractional Hamming Distance (FHD)[2]. The results of this test can be found in Figure 4. The number of measurements per device is set to the horizontal axis, while the vertical axis presents the FHD between start-up patterns and enrollment of the chip. At the top of the graph the different conditions, in this case temperatures (Temp_m40 = -40°C, Temp_025 = +25°C, Temp_085 = +85°C), are specified. Each line in the Figure represents a different Buskeeper PUF. The spike to FHD = 0 represents the enrollment measurement of each Buskeeper (since FHD to itself is 0). A similar representation is used in this paper for all the other test results.

It can be seen in this Figure that the FHD increases when the temperature deviates from the enrollment temperature. The Buskeeper PUF appears to be more sensitive for high temperatures than low temperatures. Furthermore, it can be concluded that the noise on Buskeeper PUFs due to these temperature variations remains below 20% (FHD = 0.2). This

[2]Hamming Distance (HD) is defined as the number of bits that differ between two bit strings. In case of fractional Hamming Distance (FHD) the HD is divided by the length of the compared strings.

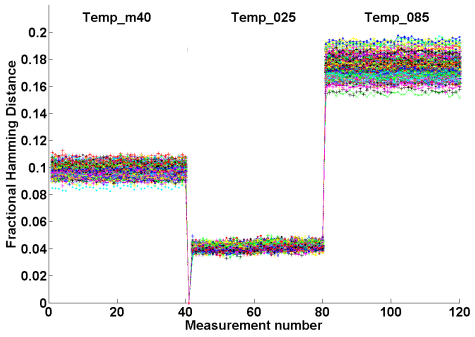

Fig. 4. Measurement results from Temperature Variation Test.

is well within the boundaries, as specified earlier, for error correction using a Fuzzy Extractor.

2) Voltage Variation Test: To investigate the influence of varying supply voltage levels on the reliability of Buskeeper PUFs, the 96 ICs (192 Buskeepers) have been placed in a set-up suitable for varying the supply voltage from 90% of Vdd to 110% of Vdd. An enrollment measurement for each Buskeeper has been taken with supply voltage Vdd. At each voltage level the Buskeeper PUFs have been measured 20 times. The results of performing this test at +25°C can be found in Figure 5 (minus_10perc = 90% of Vdd and plus_10perc = 110% of Vdd). This test has also been performed at +85°C and -40°C for which the results were similar to those at +25°C.

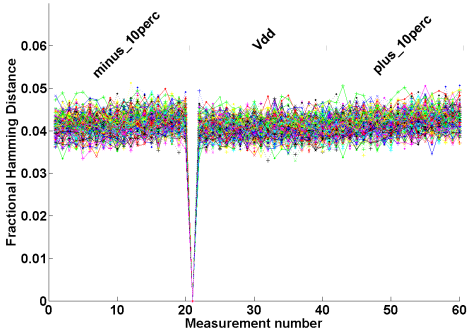

Fig. 5. Measurement results from Voltage Variation Test at +25°C.

It becomes clear from these results that varying the supply voltage of Buskeeper PUFs does not influence the reliability of their start-up patterns (at any ambient temperature). With a noise level below 5% the Buskeeper PUFs are very stable at different supply voltages.

3) Voltage Ramp-up Test: To investigate the influence of varying the ramp-up time of the supply voltage on the reliability of Buskeeper PUFs, the 96 ICs (192 Buskeepers) have been placed in a set-up suitable for varying this ramp. An enrollment measurement for each Buskeeper has been taken with ramp-up time of 10μs. At each ramp-up time the Buskeeper PUFs have been measured 10 times. The results of performing this test at +25°C can be found in Figure 5. This test has also been performed at +85°C and -40°C for which the results were similar to those at +25°C.

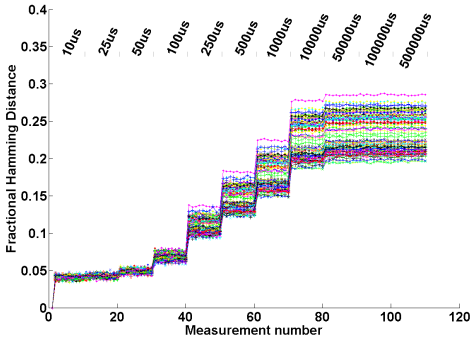

Fig. 6. Measurement results from Voltage Ramp-up Test at +25°C.

It becomes clear from these results that varying the ramp-up time of the supply of Buskeeper PUFs influences the reliability of their start-up patterns significantly (at any ambient temperature). Patterns created using a longer ramp-up time differ from those created with shorter ones. Based on these results it is advised to keep the ramp-up time of Buskeeper supplies as close to the time used for enrollment as possible, in this case below 100μs which will keep the FHD below 7%.

4) Ageing Test: For the ageing tests, five ICs have been placed in an oven at +85°C and supply voltage set to 120% of Vdd (1.44V). Under these conditions, we accelerate the ageing effect of ICs. The total acceleration factor [1] is computed as the product of the Thermal Acceleration Factor (TAF) and the Voltage Acceleration Factor (VAF), which are computed as:

$$\text{TAF} = e^{\frac{E_a}{k}(\frac{1}{T_{op}} - \frac{1}{T_{stress}})} \text{ and } \text{VAF} = e^{\gamma(V_{stress} - V_{op})}$$

Typical values in these formulas are: activation energy E_a (0.5eV), Boltzmann's constant k (8.62 10^{-5}eV/°K), voltage exponent factor γ (2.6) and normal operating temperature T_{op} (313°K (+40°C)). For our test set-up T_{stress} (358°K (+85°C)) is the stress temperature used, V_{stress} (1.44V) is the stress core voltage and V_{op} (1.2V) is the core voltage under normal operating conditions. This results in a total estimated acceleration factor of TAF×VAF = 10.27×1.77 = 18.2.

Every week both the ambient temperature and the voltage level have been lowered temporarily (to +25°C and 1.2V respectively), in order to measure the Buskeeper start-up values. One measurement for each PUF at an ambient temperature of +25°C and supply voltage of 1.2V from before starting the ageing test has been used as enrollment, to which all other measurements are compared. Comparison between measurements is based on the FHD between the start-up patterns.

The ageing test has been running for 2150 hours. With the estimated acceleration factor of 18.2, we simulate an effective ageing of around 53.5 months, hence almost 4.5 years. The results show that within this time frame the ageing is quite limited. The maximum FHD remains below 7%, which can easily be corrected by common FE implementations.

C. Uniqueness Tests

1) Between-class Distribution Test: When performing uniqueness tests, we are interested in finding out whether it is possible to distinguish between different devices given their PUF responses. This is required to make sure that unique keys can be derived from different Buskeeper PUFs. The first evaluation is performed by creating a Between-Class Distribution of the different enrollment patterns from the Temperature Variation Test using the FHDs between the different PUFs. This results in a distribution of FHDs that can be approximated as a Gaussian curve with an average value μ and a standard deviation σ. To get an indication whether PUFs are uniquely identifiable, the value of μ should be close to 0.5 and σ should be small. In case of the tested Buskeeper PUFs $\mu = 0.49015$ and $\sigma = 0.007162$ and therefore this is a good first indication that these PUFs are uniquely distinguishable.

2) Entropy Estimation: To estimate the entropy of Buskeeper PUFs, we use a compression algorithm (to estimate an upper bound) and calculate the min-entropy (which leads to a lower bound). The actual entropy of these PUFs will be somewhere between these boundaries. Context-Tree Weighting (CTW) [15] is an optimal compression method for a stationary ergodic source, which we assume the PUF data to be. This algorithm can be used to check the ability to compress PUF response strings, as shown in [14] and [4]. The amount of compression will give an estimate of the upper bound of the entropy of our PUF responses. When the algorithm is capable of compressing the PUF responses, the responses do not have full entropy. This test was carried out by first concatenating all enrollment patterns from the Temperature Variation Test into one string. As can be seen in Table I, very little compression is achieved by CTW. This indicates that only little non-randomness is present in these PUF responses.

TABLE I
RESULTS OF CTW COMPRESSION TEST

Original size	Size after CTW	Compression ratio
192*8192 = 1572864	1553605	98.8%

Besides the compression factor, it is also possible to estimate the min-entropy of the Buskeepers. Min-entropy is the worst-case (i.e., the greatest lower bound) measure of uncertainty for a random variable. For this purpose we use the method that is described below (taken from appendix C of NIST specification 800-90 [2]).

Output values of binary sources have a probability of occurring p_0 and p_1 respectively (sum of these probabilities is 1). When p_{max} is the maximum value of these two probabilities, the definition for min-entropy of a binary source is:

$$H_{min} = -log_2(p_{max})$$

Assuming that all bits from the PUF start-up pattern are independent (which is plausible, since Buskeepers can be spread randomly over the entire surface of an IC), each bit of the pattern can be viewed as an individual binary source. For n independent sources (in this case n is the length of the start-up pattern) the definition below holds, which is a summation of the entropy from each individual bit.

$$(H_{min})_{total} = \sum_{i=1}^{n} -log_2(p_{i\ max})$$

For our calculations we take the enrollment patterns that we have used during the Temperature Variation Test. These patterns are bitwise summarized to calculate a weight W per bit, which can have a value between 0 and the number of enrollment patterns (m). Based on this W, p_{max} can be calculated for each individual bit of the start-up pattern:

$$\begin{aligned} \text{if } \mathrm{W}_i > m/2 : p_{i\ max} &= \mathrm{W}_i/m, \\ \text{else: } p_{i\ max} &= (m - \mathrm{W}_i)/m \end{aligned}$$

Based on these values for p_{max}, the min-entropy of each individual bit (source) and the total min-entropy of the start-up pattern can be calculated using the formulas above. Finally, the average min-entropy per bit of a memory is calculated by dividing $(H_{min})_{total}$ by the length of the pattern n.

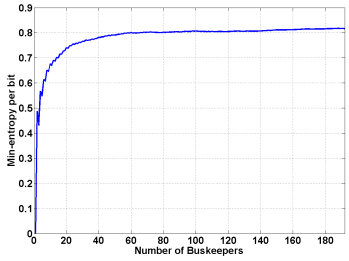

Fig. 7. Min-entropy development over the number of enrollment files (m).

Figure 7 displays how the average min-entropy per bit of the Buskeepers develops over an increasing m. It can be seen that after using 192 devices for this min-entropy test (the total number of instances measured for this paper), the average min-entropy per bit is 0.82 and still rising. This means that the values found by this test are conservative estimates, since these values would increase with more devices. From the results we conclude that the entropy of the tested Buskeeper PUFs is a value between 0.82 and 0.988 per bit.

IV. COMPARISON TO DFF PUFS

A. Comparing Test Results

Although DFF PUFs were introduced in [11], that paper does not contain sufficient statistical analysis for comparison to the Buskeeper results from this paper. Therefore, we compare our results to those from the following two references: [14] and [4]. The results of this comparison can be found in Table II. In this overview a + sign indicates that the Buskeeper PUFs are better than the DFF from literature, 0 means similar, with - DFF is better, and n.a. denotes that a specific test was not part of the reference.

Based on these comparison results we conclude that both the reliability and uniqueness properties of Buskeeper PUFs

TABLE II
COMPARISON TO DFF RESULTS FROM LITERATURE

Test	Results [14] (130nm)	Compare	Results [4] (65nm)	Compare	Remarks
Temperature Variation	Max. FHD: 13%	-	Max. FHD: 40%	+	Max. FHD of Buskeeper: 20%
Voltage Variation	n.a.	n.a.	Max. FHD: 7%	0	Does not influence either PUF type
Voltage Ramp-up	n.a.	n.a.	Max. ramp time: 10μs	+	Longer max. ramp-up (100μs) for Buskeeper
Ageing	Max. FHD: 10%	+	Max. FHD: 10%	+	Max. FHD of Buskeeper: 7%
Between-class Uniqueness	μ = 0.36	+	μ = 0.4992	-	μ of Buskeeper is 0.49015
Entropy Est. (upper-bound)	Compres.: 81.3%	+	Compres.: 100%	-	Compression rate of Buskeeper is 98.8%
Entropy Est. (lower-bound)	n.a.	n.a.	Min-entropy: 0.77	-	Min-Entropy Buskeeper is 0.75 at 20 devices

are comparable to those of DFF PUFs from literature. Both on reliability and uniqueness, the results from the Buskeeper PUFs are somewhere between the DFF results (results are more reliable than those from [4] and more unique than from [14]) and always in the same order of magnitude.

B. Hardware Comparison

Comparing the hardware implementations of Buskeeper and DFF PUFs consists of three important properties. First, it is important for PUFs to be constructed from standard CMOS components. This makes integration into the design flow of an IC easy. Both PUF types consist of standard CMOS components. Even though both PUF types are standard components, it is not recommended to use Buskeeper or DFF cells of PUF implementations for their regular purpose at the same time. This could decrease their performance as PUF instances.

A second important property of DFF PUFs is that they can be freely distributed over the surface of an IC. Buskeepers have this property as well (although the ones tested for this paper were clustered), which makes both PUFs very difficult to reverse-engineer when used to hide a secret key in a design. Some other PUF types (like SRAM) cannot be distributed.

The final implementation property to consider is the required amount of resources. Many different types of DFFs are available and their cell sizes vary from 6 to 8 GE (without resources for addressing). For Buskeeper PUFs, the cell size is only 1 GE (without addressing). The addressing of DFFs can be done using a (scan-)chain. This is more efficient than the addressing of Buskeepers using MUX trees. However, this is only a marginal difference in favour of DFFs, which does not change the fact that the total amount of resources required for Buskeeper PUFs is significantly lower than for DFFs.

In PUF systems, additional hardware is required for error correction. These resources increase with the number of errors that need to be corrected. References [14] and [4] have shown that the maximum amount of noise for DFF PUFs can vary between 14 and 40%. Given the fact that the maximum amount of noise measured for the Buskeeper PUFs is 20%, we conclude that Buskeeper PUFs in principle do not require more hardware resources for error correction than DFF PUFs.

V. CONCLUSIONS

In this paper we have introduced a new PUF type, which is a promising alternative to the generally known DFF PUF. We have proven that Buskeeper cells can be used as PUFs with reliability and uniqueness properties comparable to those of DFF PUFs from literature. Also, Buskeepers are standard CMOS components that can be distributed freely over the surface of an IC (like DFFs). Besides comparable properties, Buskeepers are much smaller than DFFs. Therefore, we conclude that Buskeepers are an exciting new type of PUF and a viable alternative to DFF PUFs, requiring less resources.

ACKNOWLEDGEMENT

This work has been supported by the European Commission through the FP7 programme under contract 238811 UNIQUE. The authors would like to thank all partners from the UNIQUE project, who have taken part in the design and production of the ASICs and test boards that have been used for this research.

REFERENCES

[1] Altera. Reliability report 52 q3 2011. http://www.altera.com/literature/rr/rr.pdf.
[2] E. Barker and J. Kelsey. NIST Special Publication 800-90: Recommendation for random number generation using deterministic random bit generators (revised), March 2007 NIST. Technical report.
[3] X. Boyen. Reusable cryptographic fuzzy extractors. In *ACM Conference on Computer and Communications Security*, pages 82–91, 2004.
[4] M. Claes, V. van der Leest, and A. Braeken. Comparison of SRAM and FF PUF in 65nm technology. In *Proc. of NordSec'11*. LNCS, 2011.
[5] Y. Dodis, R. Ostrovsky, L. Reyzin, and A. Smith. Fuzzy extractors: How to generate strong keys from biometrics and other noisy data. *SIAM J. Comput.*, 38:97–139, March 2008.
[6] B. Gassend, D. Clarke, M. van Dijk, and S. Devadas. Silicon physical random functions. In *Proceedings of ACM CCS'02*, pages 148–160, New York, NY, USA, 2002. ACM.
[7] J. Guajardo, S. S. Kumar, G.-J. Schrijen, and P. Tuyls. FPGA intrinsic PUFs and their use for IP protection. In *Proceedings of CHES'07*, pages 63–80, Berlin, Heidelberg, 2007. Springer-Verlag.
[8] S. Kumar, J. Guajardo, R. Maes, G.-J. Schrijen, and P. Tuyls. Extended abstract: The butterfly PUF protecting IP on every FPGA. In *Proceedings of HOST'08*, pages 67 –70, june 2008.
[9] 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 *VLSI Circuits, 2004. Digest of Technical Papers. 2004 Symposium on*, pages 176 – 179, june 2004.
[10] J.-P. Linnartz and P. Tuyls. New shielding functions to enhance privacy and prevent misuse of biometric templates. In *Proceedings of AVBPA'03*, pages 393–402, Berlin, Heidelberg, 2003. Springer-Verlag.
[11] R. Maes, P. Tuyls, and I. Verbauwhede. Intrinsic PUFs from Flip-flops on reconfigurable devices. In *Workshop on Information and System Security (WISSec 2008)*, page 17, Eindhoven,NL, 2008.
[12] P. S. Ravikanth. *Physical one-way functions*. PhD thesis, 2001. AAI0803255.
[13] B. Skoric, P. Tuyls, and W. Ophey. Robust key extraction from Physical Uncloneable Functions. In *Applied Cryptography and Network Security*, volume 3531 of *LNCS*, pages 99–135. Springer, 2005.
[14] V. van der Leest, G.-J. Schrijen, H. Handschuh, and P. Tuyls. Hardware intrinsic security from D flip-flops. In *Proceedings of ACM STC'10*, STC 2010, pages 53–62, New York, NY, USA, 2010. ACM.
[15] F. Willems, Y. Shtarkov, and T. Tjalkens. Context-Tree Weighting: Basic properties. *IEEE Trans. on Inf. Theory*, 41:653–644, 1995.