Extended Abstract: The Butterfly PUF Protecting IP on every FPGA

Sandeep S. Kumar*, Jorge Guajardo*, Roel Maes†‡, Geert-Jan Schrijen* and Pim Tuyls*
*Philips Research Europe,
5656 AE, Eindhoven, THE NETHERLANDS
Email: {sandeep.kumar,jorge.guajardo,geert.jan.schrijen,pim.tuyls}@philips.com
†K.U.Leuven, ESAT/COSIC,
B-3001 Leuven-Heverlee, BELGIUM
Email: Roel.Maes@esat.kuleuven.be

***Abstract*—IP protection of hardware designs is the most important requirement for many FPGA IP vendors. To this end, various solutions have been proposed by FPGA manufacturers based on the idea of bitstream encryption. An alternative solution was advocated in [18]. Simpson and Schaumont proposed in [18] a new approach based on Physical Unclonable Functions (PUFs) for IP protection on FPGAs. PUFs are a unique class of physical systems that extract secrets from complex physical characteristics of the integrated circuits which along with the properties of unclonability provide a highly secure means of generating volatile secret keys for cryptographic operations. However, the first practical PUF on an FPGA was proposed only later in [7] based on the startup values of embedded SRAM memories which are intrinsic in some of the current FPGAs. The disadvantage of these intrinsic SRAM PUFs is that not all FPGAs support uninitialized SRAM memory. In this paper, we propose a new PUF structure called the *Butterfly PUF* that can be used on all types of FPGAs. We also present experimental results showing their identification and key generation capabilities.**

Key Words. Physical Unclonable Functions, Intrinsic PUFs, SRAM, cross-coupled circuits, identification, secret-key storage, FPGAs, IP Protection.

I. INTRODUCTION

Reusable IP is a major source of revenue for IP design vendors and their protection is of high importance. The main issues that are involved are: (a) the IP being leaked to parties other than those originally intended to obtain it and (b) the over usage of the IP with respect to the licensed amount. IP to be used on SRAM FPGAs is more vulnerable to these types of attacks as the programming bitstream has to be stored on external non-volatile memory. Hence, the bitstream can be easily copied by an attacker and used on a similar off-the-shelf FPGA. FPGA manufacturers have tried to solve the problem using various ways to encrypt the bitstream stored in external non-volatile memory. However, these methods rely on battery backed [12] or flash based [1], [22] key storage on the FPGA, which themselves introduce other problems for a field deployment and hence are not very widespread. Apart from the deployment, solutions based on non-volatile memory are vulnerable to invasive attacks as the secret is present in the memory during such an attack. Only tamper-sensing circuitry with continuous battery power can solve this problem with the associated higher price. Physical Unclonable Functions (PUFs) [15], [16] with its unique circuit based on the intrinsic physical characteristics of integrated circuits provide a significantly higher security assurance as keys are volatile and derived only when required. In addition, an invasive attacker will destroy the PUF (with very high probability) during the invasive process, making it very hard to obtain the key. The main advantage of using a PUF is that all this additional physical security is achievable without any special manufacturing steps as PUFs are based on process variations introduced during the manufacturing process. These process variations are beyond the control of the manufacturers. Notice that with advancing technologies, variations are even more conducive to more efficient PUFs. Simpson and Schaumont proposed in [18] a new approach based on these PUFs for IP protection on FPGAs. Their work only assumes the existence of a PUF without proposing a PUF construction. In [7], the authors introduced intrinsic PUFs for FPGAs based on the startup values of SRAM memories. An SRAM cell is a cross-coupled inverter circuit which maintains its state using positive feedback. During startup, a slight difference in the voltage on one of the floating inverters output is driven positively within the loop to force the SRAM to go to a 1 or a 0. Though this was a practical construction of a PUF on an FPGA, the main disadvantage is that SRAM memories in most FPGAs are forcibly set to a known state upon startup. Hence, it is of interest to develop new intrinsic PUFs that can be instantiated on a large family of FPGAs. In this paper, we present a new PUF construction called the *Butterfly PUF* that can be used on all types of FPGAs. We first introduce the concept of cross-coupled circuits which is the basis for the Butterfly PUF.

II. CROSS-COUPLED CIRCUITS

A cross-coupled circuit is a basic building block used in all types of storage elements in electronic circuits such as latches, flip-flops and SRAM memories. A cross-coupled circuit is constructed such that it provides a positive-feedback loop to store the required bit value within the loop. An example of

‡Part of this work was performed while the author was visiting Philips Research. The author's research is also funded by IWT-Vlaanderen under grant number 71369.

such a circuit is a simple latch built using two cross-coupled inverters as shown in Figure 1.

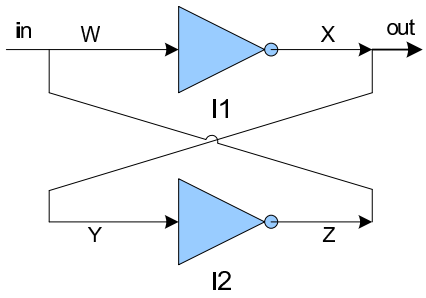

Fig. 1. Cross-coupled inverter

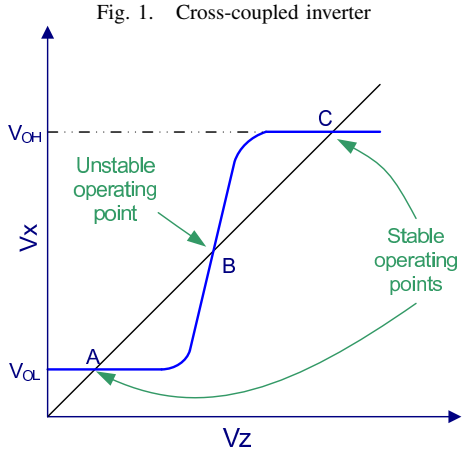

Fig. 2. Cross-coupled inverter stable states

Notice that such cross-coupled circuits have two different stable operating points (to store the bit value) and an unstable operating point as shown in Figure 2. The circuit can be easily driven from the unstable state to a stable state by an external signal on the input or due to slight differences in the elements used to build the circuit (here inverters). We use this fact to build a PUF where the circuit is initially at the unstable operating point and left to attain one of the two stable operating points without any external excitation. We find that with high probability the circuit goes more often to one of the stable states. This behavior is due to small differences in the wire delays and cross-coupled element's (here inverter) voltage transfer characteristics. It is important to note that these circuits are constructed as symmetrically as possible and all variations are due to randomness in the circuit which is beyond the control of the designer. Different cross-coupled devices can be built using different elements like NOR gates or NAND gates.

III. The Butterfly PUF

The concept of the Butterfly PUF (BPUF) is based on the idea of creating structures within the FPGA matrix which behave similarly to a SRAM cell during the startup phase. A BPUF cell is a cross-coupled circuit which can be brought to a floating/unstable state before allowing it to settle to one of the two stable states that are possible. Implementing a cross-coupled element using combinational logic on an FPGA is not straightforward due to the inability to create combinational loops. To overcome this problem we simulate a cross-coupled combinational loop using latches present in the FPGA. We create a cross-coupled structure using latches. This allows for an unstable state set by an excite signal, which then settles down to one of two possible stable states after some time.

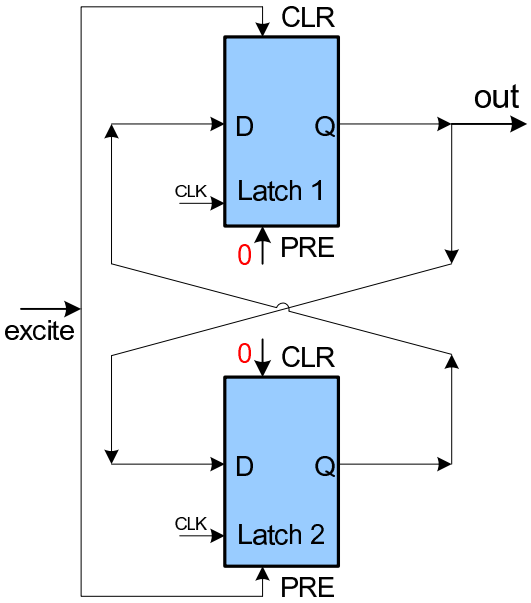

Fig. 3. Butterfly PUF: Cross-coupled Latches

The structure of the BPUF cell is as shown in Figure 3 constructed *as symmetrically as possible* by manual routing of the signal wires. It consists of two latches, each with a preset (PRE) signal (which turns output Q to 1 on high) and a clear (CLR) signal (which turns output Q to 0 on high). The data D is transferred to the output Q when the CLK is high. In the construction, the PRE of $Latch$ 1 and CLR of $Latch$ 2 are always set to low. The $excite$ signal is connected to CLR of Latch 1 and PRE of Latch 2. The outputs of the latch are cross-coupled. We set CLK in both latches to always high, effectively simulating a combinational loop. To start the PUF operation, the $excite$ signal is set to high. This brings the BPUF circuit to an unstable operating point (as both latches have opposite signals on their inputs and outputs). After a few clock cycles the excite signal is set to low. This starts the process of the PUF circuit to attain either one of the two possible stable states, 0 or 1, on the out signal. The stable state depends on the slight differences in the delays of the connecting wires which are designed using symmetrical paths on the FPGA matrix. Hence, these slight variations are only based on the intrinsic characteristics of the integrated circuit and vary from device to device and position on the FPGA. However, for the same FPGA, latch locations, and routing resources, the stable state tends to be the same over time and over a large temperature range. An attacker cannot derive theses stable states as the location and routing used for the BPUF structures are not easily visible based on

current advances in bitstream reverse engineering [4].

IV. Experimental Validation

For our experimental validation, we used Virtex-5 Xilinx FPGAs. We constructed an array of 64 BPUF structures on 36 devices from each FPGA family. The PUF measurements were repeated 200 times (which involves exciting and recording the stable state) for each temperature step of $20°C$ starting from $-20°C$ to $+80°C$. The important parameters for validating a PUF are the *within-class variation in Hamming distance* for measurements performed on the same FPGA and the *between-class variation in Hamming distance* for measurements performed on different FPGA devices. Figure 4 shows that the within-class Hamming distance for the BPUF is within 6% from a reference measurement performed at $+20°C$, whereas the between-class Hamming distance shows a mean close to 50%. The separation between the two classes allows for a clear threshold for identification of FPGAs based on the PUF responses. Experiments at various temperatures also show the PUF responses to be relatively stable over a large temperature range. Figure 5 shows the within-class errors for two FPGA devices at various temperatures ranging from $-20°C$ to $+80°C$. Various other tests like varying operating frequency from 50 MHz to 120 MHz and FPGA core voltage were also performed, which validated the Butterfly PUF to be very stable. Since the BPUF is built within the FPGA matrix, it was also important to validate that there were no effects on the stable states from adjoining circuits using various designs along with the PUF. Hence the Butterfly circuit is a promising implementation of a PUF circuit within the FPGA matrix whose properties depend only on the intrinsic physical characteristics of the integrated circuit and can be used for identification.

To end this section, we observe that from these 64 BPUF cells, we can derive an identifier for the FPGA requiring 130 slices for a full entropy 50-bit identifier. If, however, we wanted to derive a cryptographic key, then we must be able to generate the same key based on a noisy response with a 6% noise level assuming pre-processing during the enrollment stage [7]. To cope with the noisy nature of PUFs, we can use a fuzzy extractor or helper data algorithm as introduced in [14], [3]. Using a helper data algorithm construction [14], [3] and assuming a 0.78 bits of entropy for every BPUF output bit, we would need about 1500 Butterfly PUF cells to derive a uniformly distributed random 128-bit key with a failure rate of 10^{-6} based on the techniques described in [2].

V. Related Work

In 2001, Pappu et al. [15], [16], introduced the concept of Physical Unclonable Functions (PUFs) or Physical Random Functions. The original construction of [15] is based on the response (scattering) obtained when shining a laser on a bubble-filled transparent epoxy wafer. Gassend et al. introduce Silicon Physical Random Functions (SPUF) [6] which use manufacturing process variations in ICs with identical masks

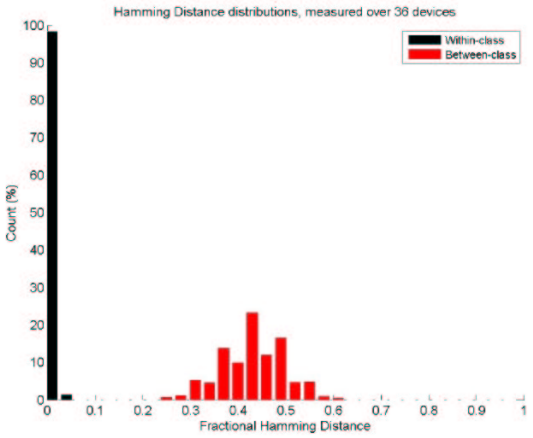

Fig. 4. Hamming Distance: within-class and between-class

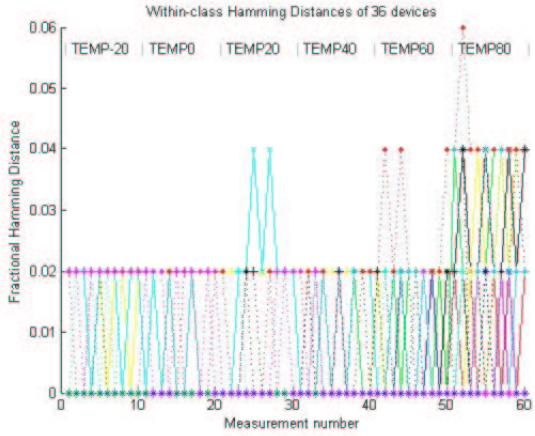

Fig. 5. Temperature variance: within-class Hamming distance

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. Silicon PUFs are very sensitive to environmental variations like temperature and voltage. Lim et al. [13] 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 "weak" PUF from external attacks. Recently, Su et al. [19] 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. Suh and Devadas [20] present a PUF based on ring oscillators which can also be implemented with the FPGA. However, the Butterfly PUF has distinct advantages in terms of the area resources required in addition to the fact that Butterfly PUF responses are binary as opposed to delay-based PUFs, whose

responses are analog and thus require further processing. It is also important to point out that the Butterfly PUF is superior to delay-based methods in terms of performance and power. Both advantages are thanks to the fact that the response of a Butterfly-PUF is obtained almost instantaneously whereas a ring-based oscillator PUF requires of time to obtain the desired response.

In [18], Simpson and Schaumont showed that by using a PUF on an FPGA they could develop protocols which allow binding of a particular IP to a particular FPGA. Their protocols also allow proving authenticity of the IP to the hardware platform. In [7], the authors reduce the computation and communication complexity of the protocols in [18] and introduce the idea (and an explicit construction) of Intrinsic-PUFs based on the start-up values of SRAM memory values. Both, based their protocols on symmetric-key primitives. A similar idea to SRAM-based Intrinsic-PUFs is presented in [11] but the focus is on ultra-low power micro-controllers. In [8], the authors observe that by introducing public-key cryptography, the corresponding private-key does not need to ever leave the FPGA, even during the enrollment stage, thus increasing the security of the overall system.

VI. Application Areas

Since their introduction in [15], [16], PUFs have received considerable attention from the security community because of the unclonability and randomness properties inherent to them. Applications include: IC identification [6], IP protection in FPGAs [18], [7], [8], remote service and feature activation [9], secret-key storage [21], authentication via challenge-response protocols [15], key distribution in wireless sensor networks [10], and trusted computing [17], to name a few.

VII. Conclusions

We presented a new construction of a PUF which can be used for all the various types of FPGAs. The Butterfly PUF uses the internal matrix of the FPGA to uniquely identify it based on the intrinsic physical characteristics of the integrated circuits. Experimental results show that it is very stable to environmental and other FPGA operating parameter variations. Hence, the Butterfly PUF promises to be a significantly secure way to protect IP with no additional costs in manufacturing. Other secure features like volatile key generation for cryptographic applications are also shown to be feasible due to the low noise levels.

References

[1] ALTERA. Application Note 341 v2.0. Using the Design Security Feature in Stratix II and Stratix II GX Devices, February 2007. Available at http://www.altera.com/literature/an/an341.pdf.

[2] C. Bösch, J. Guajardo, A.-R. Sadeghi, J. Shokrollahi, and P. Tuyls. Efficient Helper Data Key Extractor on FPGAs. In P. Rohatgi and E. Oswald, editors, *Cryptographic Hardware and Embedded Systems — CHES 2008*, LNCS. Springer, 2008. To appear.

[3] Y. Dodis, M. Reyzin, and A. Smith. Fuzzy extractors: How to generate strong keys from biometrics and other noisy data. In C. Cachin and J. Camenisch, editors, *Advances in Cryptology —- EUROCRYPT 2004*, volume 3027 of *LNCS*, pages 523–540. Springer-Verlag, 2004.

[4] Saar Drimer. Volatile FPGA design security – a survey, December 2007. http://www.cl.cam.ac.uk/~sd410/papers/fpga_security.pdf.

[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*, page 149, Washington, DC, USA, 2002. IEEE Computer Society.

[6] B. Gassend, D. E. Clarke, M. van Dijk, and S. Devadas. Silicon physical unknown functions. In V. Atluri, editor, *ACM Conference on Computer and Communications Security — CCS 2002*, pages 148–160. ACM, November 2002.

[7] J. Guajardo, S. S. Kumar, G.-J. Schrijen, and P. Tuyls. FPGA Intrinsic PUFs and Their Use for IP Protection. In P. Paillier and I. Verbauwhede, editors, *Cryptographic Hardware and Embedded Systems — CHES 2007*, volume 4727 of *LNCS*, pages 63–80. Springer, September 10-13, 2007.

[8] J. Guajardo, S. S. Kumar, G.-J. Schrijen, and P. Tuyls. Physical Unclonable Functions and Public Key Crypto for FPGA IP Protection. In *Proceedings of the 2007 International Conference on Field Programmable Logic and Applications — FPL 2007, Amsterdam, The Netherlands*, pages 189–195. IEEE, August 27-30, 2007.

[9] J. Guajardo, S. S. Kumar, G.-J. Schrijen, and P. Tuyls. Brand and IP Protection with Physical Unclonable Functions. In *IEEE International Symposium on Circuits and Systems — ISCAS 2008*. IEEE, May 18-21, 2008.

[10] J. Guajardo, S. S. Kumar, and P. Tuyls. Key Distribution for Wireless Sensor Networks and Physical Unclonable Functions. Printed handout of Secure Component and System Identification — SECSI 2008, March 17-18, 2008.

[11] 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.

[12] R. Krueger. Using High Security Features in Virtex-II Series FPGAs. Xapp766 (v1.0), Xilinx, July 8, 2004. Available at http://www.xilinx.com/bvdocs/appnotes/xapp766.pdf.

[13] 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*, 13(10):1200–1205, October 2005.

[14] J.-P. M. G. Linnartz and P. Tuyls. New Shielding Functions to Enhance Privacy and Prevent Misuse of Biometric Templates. In J. Kittler and M. S. Nixon, editors, *Audio-and Video-Based Biometrie Person Authentication — AVBPA 2003*, volume 2688 of *LNCS*, pages 393–402. Springer, June 9-11, 2003.

[15] R. S. Pappu. *Physical one-way functions*. PhD thesis, Massachusetts Institute of Technology, March 2001. Available at http://pubs.media.mit.edu/pubs/papers/01.03.pappuphd.powf.pdf.

[16] R. S. Pappu, B. Recht, J. Taylor, and N. Gershenfeld. Physical one-way functions. *Science*, 297(6):2026–2030, 2002. Available at http://web.media.mit.edu/~brecht/papers/02.PapEA.powf.pdf.

[17] D. Schellekens, P. Tuyls, and B. Preneel. Embedded Trusted Computing without Non-Volatile Memory. In A.-R. Sadeghi and P. Lipp, editors, *TRUST Conference*, LNCS. Springer, March 11-12, 2008.

[18] E. Simpson and P. Schaumont. Offline Hardware/Software Authentication for Reconfigurable Platforms. In L. Goubin and M. Matsui, editors, *Cryptographic Hardware and Embedded Systems — CHES 2006*, volume 4249 of *LNCS*, pages 311–323. Springer, October 10-13, 2006.

[19] 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*, pages 406–408, Washington, DC, USA, 2007. IEEE Computer Society.

[20] G. Edward Suh and Srinivas Devadas. Physical unclonable functions for device authentication and secret key generation. In *Proceedings of the 44th Design Automation Conference*, pages 9–14. IEEE, 2007.

[21] 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*, volume 4249 of *Lecture Notes in Computer Science*, pages 369–383. Springer, October 10-13, 2006.

[22] Xilinx, editor. *Security Solutions Using Spartan-3 Generation FPGAs*. Xilinx, San Jose, 2008.