2017 Seventh International Conference on Emerging Security Technologies (EST)

Random Stream Cipher as a PUF-like Identity in FPGA Environment

Ayoub Mars, Wael Adi, Saleh Mulhem, Emad Hamadaqa
IDA, Institute of Computer and Network Engineering
Technical University of Braunschweig, Germany
{a.mars, w.adi, s.mulhem, e.hamadaqa}@tu-bs.de

***Abstract*—Digital Secret Unknown Cipher (SUC) has been proposed in the last decade targeting to counteract the drawbacks of the traditional analog Physical Unclonable Functions (PUF). The SUCs, as pure-digital units, exhibit consistent operation during the whole digital unit's lifetime. This makes SUCs as PUF alternatives attractive for practical creation of clone-resistant units for a broad spectrum of applications especially in automotive and other mass-product cases. In this paper, we propose a novel SUC creation scenario as a random stream cipher to be embedded in Microsemi SmartFusion®2 System on Chip (SoC) FPGAs. The resulting cipher as a Key Stream Generators (KSG) is T-functions-based creating individual and randomized secret-key-streams which are individual for each unit. The proposed SUC-based Random Stream Cipher (RSC-SUC) is a lightweight SUC realization optimized for the SmartFusion®2 SoC FPGA technology. The security analysis and hardware complexity issues of the resulting RSC-SUC structures are also discussed and evaluated.**

Keywords*—*Secret Unknown Cipher; T-functions; Key Stream Generator; PUFs; self-reconfiguring SoC FPGAs.

I. INTRODUCTION

Modern systems require clone-resistant units to be embedded and initiated as system security anchors. PUFs [1][2] have been proposed in the last two decades as unclonable units for key storage and identification [3]. Traditional PUFs are however inconsistent due to their sensitivity to aging and operating conditions such as temperature and voltage variations. They are also vulnerable to modeling attacks as in [4] and hybrid side channel attacks and modeling attacks [5]. To overcome such PUFs inconstancy, fuzzy extractors were proposed beside PUFs to provide consistent long lasting operation. Unfortunately, fuzzy extractors involve additional, relatively high hardware complexity making them impractical for constrained devices and mass productions. Moreover, many attack proposals on PUFs with fuzzy extractors have been found showing that PUFs with fuzzy extractors are less vulnerable to modeling attacks, however opening new attacking gaps due to the existing fuzzy extractors [6]. Besides PUFs, other concept for creating digital DNA-like identity by creating random unknown ciphers called "Secret Unknown Cipher" (SUC) were proposed in [7]. SUCs are conceptually designed to be embedded in non-volatile digital SoC FPGAs to operate consistently, in difference to analog PUFs during the whole lifetime of the digital units. Unfortunately, no real-world proposal has been yet published for practically creating such units. In this paper, we propose a first attempt for creating SUC-based class of Random Stream Cipher (RSC-SUC). The proposed RSC-SUC uses single cycle T-functions (Triangular-Functions) and few S-Boxes. In [8][9], Klimov and Shamir proposed a class of single cycle T-functions and presented their security analysis. These functions have a cycle length of 2^n for n-bit integer arithmetic in Z_{2^n}, which makes them suitable as building blocks in stream ciphers. Deploying T-function as Key Stream Generator (KSG) generates long cycles, however with modest security level of ($2^{n/3}$) according to [8]. We overcome this security drawback by combining a T-function with few random S-Boxes from a class of FPGA-inspired involutive S-Boxes to create a random KSG as RSC-SUC. The resulting mappings-cardinality is relatively huge enhancing the overall cryptographic system entropy.

II. SECRET UNKNOWN CIPHERS

SUCs have been proposed a decade ago in [7]. Fig. 1 illustrates the key idea for creating such SUCs. Assume that a Trusted Authority (TA) injects for a short time into each System on Chip (SoC) unit a smart program called "GENIE" and lets it run just for one time. After being loaded into the chip, the GENIE creates a permanent (non-volatile) and unpredictable random cipher by consuming random bits from the True Random Number Generator (TRNG).

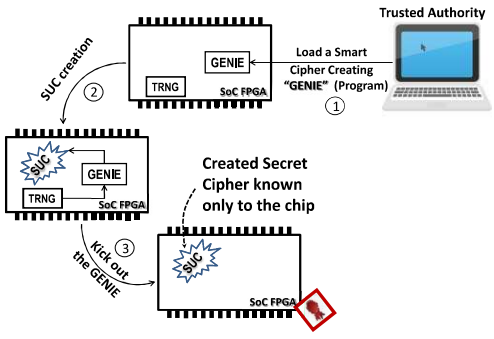

Fig. 1. Key idea for creating SUCs

After the GENIE completes the SUC creation, it will be fully

978-1-5386-4018-0/17/$31.00 ©2017 IEEE

deleted. What remains in the SoC unit/chip, is a random unknown selection of an operational cipher from a huge (theoretically infinite) number of possibilities.

The SUC technique tries to approach the biological DNA quality in terms of uniqueness and high security robustness against cloning attacks. This makes such units attractive for a broad spectrum of applications such as IP protections, identification/authentication in IoT networks, etc.

Modern SoC FPGAs include logic fabrics having programmable hardware resources such as Look Up Tables (LUT), D-Flip Flops (DFF), DSP blocks, SRAM blocks, etc. Additional one or more microcontroller/processor cores are also integrated with memory blocks and necessary communication interfaces. Non-volatile FPGAs technologies, are offered worldwide by few manufacturers. Microsemi SmartFusion®2 SoC FPGA [10] is one of the most advanced non-volatile technologies incorporating a FPGA fabric, ARM Cortex M3 processor cores, and security-supporting modules such as Random Number Bit Generator (RNBG).

Most modern FPGA technologies use bitstream encryption to protect their design contents. A successful attack on the bitstream encryption engine integrated in the Virtex-II Pro FPGAs from Xilinx, using power analysis is presented in [8]. Several generations of SRAM FPGAs were vulnerable to Differential Power Analysis (DPA) and Differential Electromagnetic Analysis (DEMA). The targeted devices for our SUC concept (Microsemi SmartFusion®2 SoC FPGA) are fabricated as tamper resistant units with countermeasures against DPA and DEMA [10]. This makes such devices secure against key extraction of the decryption engine by using side channel analysis. A practical approach to mitigate side channel analysis vulnerabilities inherent in the bitstream decryption engines in SRAM-based FPGAs is presented in [11], by using SmartFusion®2 flash-based FPGA as the secure root-of-trust.

By adopting the offered security techniques, the generated SUC will be protected against most known side channel analysis techniques which constitute the main threats for most hardware designs. Since the proposed creation process generates random ciphers that are most probably unique according to the properties of the RNBG, breaking one SUC provides no information about other SUCs and hence the attacker is forced to break each unit individually.

III. T-FUNCTIONS BASED KEY STREAM GENERATOR (KSG)

A. T-functions(Triangular Function):

Definition [8]**:** A T-function is a mapping, in which the *i*-th bit of the output depends only on the bits 0, 1, . . . , *i* of the input. All the bitwise machine operations and most of the numeric machine operations in modern processors are equivalent to T-functions, and their compositions are also T-functions.

T-Functions have been studied by Anashin in [12], who proved[1] that a T-function is invertible if and only if it can be represented in the form: $c+x+2v(x)$, and it defines a single cycle if and only if it can be represented in the form $c+x+2(v(x+1)-v(x))$ where c is some constant and $v(x)$ is some T-function.

B. Minimal T-functions Based Key Stream Generators KSG:

T-Functions are relatively new cryptographic building blocks suitable for stream ciphers. They can be used as Key Stream Generators (KSGs). In [8][9], Klimov and Shamir introduced a minimal single cycle T-function defined over Z_{2^n} as follows:

$$F_1(x) = x+(x^2 \vee C) \quad in \ Z_{2^n} \tag{1}$$

Where $C = \bullet\ldots\bullet 1\bullet 1_2$, corresponds to all odd numbers greater than or equal to 5. Another non-linear mapping with three operations is $x-(x^2 \vee C)$. These T-functions are single cycle with a cycle length of 2^n.

In [8], Klimov describes two other families of minimal mappings which are single cycle with cycle length 2^n, defined over Z_{2^n} as follows:

$$F_2(x)=x \pm (x \vee C)^2 \tag{2}$$

Where, C is any odd integer from Z_{2^n}.

We propose to use these T-functions as building block to create a class of stream ciphers; each randomly built stream cipher from these classes constitutes a RSC-SUC according to the concept described in Fig. 1. Fig. 2 shows the general structure of the proposed T-function based stream cipher.

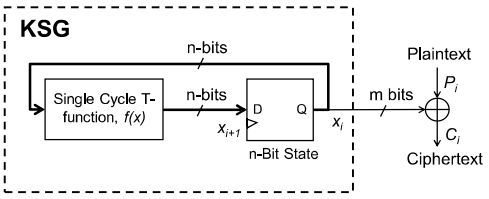

Fig. 2. General structure of T-function-based key stream generator

C. Analysis of $f(x)=x \pm (x^2 \vee C)$ based Key stream Generator

In this section, we present the security analysis of the mappings $x\pm(x^2 \vee C)$ in Z_{2^n} when used as key stream generator.

In [3], the authors present two methods to attack the key stream generator based on $x+(x^2 \vee C)$ (can be applied also to $x-(x^2 \vee C)$), where the attacker deploys special input patterns and investigates the output:

- In the first attack, which has a complexity of $O(2^{n/2})$, the attacker provides an input $x_i = 2^{n/2} y / y \in Z_{2^n}$ so $x_{i+1} = x_i \pm C \mod 2^n$.
- An improved attack has been proposed in the same paper, requiring $O(2^{n/3})$ data, memory and time complexity.

The proposed key stream generator, as shown in Fig. 2, uses a secret random initial state to create its secret key-sequence. The attacker can't manipulate this initial state, but he/she can reveal

[1] In [12] different terminology is used; the provided terminology appears in [8]

the state by a simple Chosen Plaintext Attack (CPA): as $x_i = C_i \oplus P_i$, where x_i is state-*i* and C_i is the corresponding ciphertext of the plaintext P_i in this state. An attacker willing to apply the two above proposed attacks needs to look for a state x_i satisfying at least one of the above conditions on x_i correspondingly.

We conclude that using T-function alone as key stream generator for stream ciphers is not secure enough. One possible solution to overcome this security issue is to use a large state length *n*, in this case the security level can become acceptable but this would increase the implementation complexity tremendously. Moreover, we can partition the block size into small blocks of suitable bit lengths for the targeted environment and run many T-functions in parallel.

In the next section, we propose a new FPGA optimized technique to prohibit the presented attacks on T-function-based stream ciphers.

IV. Stream Cipher Based SUC in Microsemi SmartFusion®2 SoC FPGA

A. KSG Deploying T-function with Random S-Boxes:

One weakness in deploying only a single cycle T-function as KSG is that the adversary is able to figure out the KSG states by simple operations as described in the previous section. We propose to overcome this weakness by deploying *m/4* random S-Boxes as 4-to-4 bit mappings on *m*-bits from the T-function state. In this case, the attacker needs to find the mapping of each used S-Box before attacking the KSG.

Fig. 4 describes the proposal KSG by combining a single cycle T-function and some S-Boxes with good cryptographic properties. The most significant *m*-bits of the single cycle T-function will feed *m/4* 4-bits S-Boxes; the single cycle T-function has a cycle length 2^n which will be also the cycle length of the entire KSG ($m<n$).

In [13], Lui et. al studied 4-bit involutive S-Boxes, coming up with $145920 \approx 2^{17}$ 4-bits involutive S-Boxes satisfying both linear and differential attack resistance conditions. Note that, this is a small class from the class of all possible 4-bits S-Boxes which has a cardinality of $2^4!$. These S-Boxes are also resistant to algebraic attacks according to [14], where S-Boxes which are optimal against linear and differential attacks are also optimal against algebraic attacks as well.

Since the SUC is targeting to create random unknown ciphers, it is proposed to use just few random S-Boxes (*m/4*), to prohibit the attacks described in the previous section, where the attacker tries to find a suitable state by predicting the next state equation, and then reveals the secret key by analyzing few plaintext-ciphertext pairs. Using S-Boxes would also increase the overall cardinality of the resulting SUC class.

In the proposed KSG, the adversary should first get the random S-Boxes to reveal the m bits coming out of the single cycle T-function generator. However, deducing the random S-Boxes by just analyzing the KSG output is unlikely as the attacker has no access to the S-Boxes inputs.

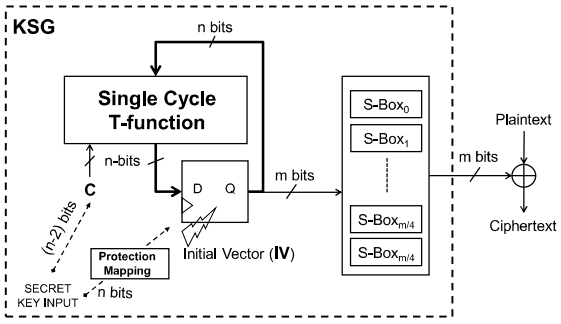

Fig. 3. Combining T-function with S-Boxes

B. Creation Concept of RSC-SUC

A novel methodology for creating RSC-SUC in non-volatile SoC FPGAs is proposed. In a first step, the custom design is created, and then a RSC-SUC template is designed and compiled incrementally. Note that, incremental layout is possible in most modern FPGA EDA tools. This would allow reducing the time needed for creating unique and secure identities for electronic devices. A functional bitstream with empty templates for RSC-SUC is now ready for downloading in each FPGA unit.

Fig. 4 describes a novel concept for personalizing SoC FPGA units with unique RSC-SUC as digital electronic DNA. The realization concept supposes that the GENIE needs only to take care of creating randomly the "random contents" of C_t, IV_t and the S-Boxes S_t. This would simplify the task of the GENIE to perform just a bitstream manipulation at the right locations through the switch SW in the targeted FPGA.

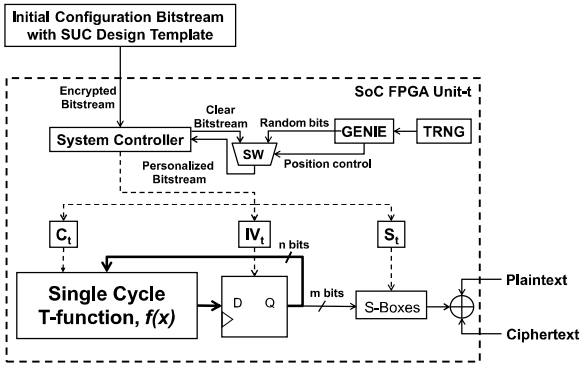

Fig. 4. Concept for personalizing individual RSC-SUCs in the enrolled units

The S-Box selections and other contents are completely randomly controlled by the unpredictable TRNG. Such operations can be done on-the-fly at a very high-speed with a GENIE having relatively low hard and software complexities.

Again, the RSC-SUC layout with empty-template is implemented and the personalization of the whole unit (RSC-SUC and custom design) is generated as a single non-

repeatable event in the programming phase. The GENIE gets unpredictable random stream of bits from the TRNG as for the parameter C_t, the Initial Vector (IV_t) and for randomly choosing S-Boxes (S_t) from a pre-compiled list of all 2^{17} optimal involutive 4-Bits S-Boxes.

When uploading the configuration bitstream to the unit, it is decrypted first to get the clear bitstream and then the software Genie modifies the desired bits corresponding to C_t, IV_t and S_t. The new configuration bitstream passes again through the system controller to configure the FPGA fabric and uniquely personalize the unit.

The Genie would (and should) be completely deleted after completing the personalization/FPGA-programming process. No back tracing is possible for the unpredictable and non-repeatable single event creation process. The resulting SUC will include a consistent and individual DNA-like electronic identity chain for the unit *t* for all its following lifetime.

The FPGA manufacturer Microsemi prohibits intentionally making the encryption/decryption operations transparent to the unit programmer in current products for design security reasons. The designer is even not allowed to have any access to the bitstream in current product versions. Therefore, self-reconfiguration is still not possible in contemporary off-shelf products. However, we postulate that self-reconfiguration is expected to become available in future non-volatile FPGA products. To protect intellectual properties, all bitstreams are encrypted using AES cipher with 256-bit keys and fully authenticated by a 256-bit tag. This protection was attacked in [15] showing discovered backdoors in Microsemi ProASIC3 products. As this protection is still a market requirement, self-reconfiguring products are expected to appear as separate types of FPGA products in the future.

C. Cardinality of the Proposed RSC SUC:

With the increasing number of connected network nodes, securing the communication between nodes in the Internet of Things (IoT) requires a hardware token that provides unique and clone-resistant physical identity for each node/unit. Considering the huge number of nodes, the possible number of RSC-SUCs that can be generated should be high enough to cover all devices. Additionally, SUCs with input keys can be optionally used to provide unique key (K_{TA}) as a personalization key for each Trusted Authority TA.

In this section, the cardinality of the proposed RSC-SUC is evaluated. As described in Fig. 4, the KSG has three initialization vectors as random secret key:

- The initialization vector (IV) with size n bits resulting with:

$$\#IV=2^{n} \tag{3}$$

- The random vector C is chosen according to the condition in (1) such that:

$$\#C=2^{n-2} \tag{4}$$

- The S-Boxes are chosen also randomly from a class of optimal involutive S-Boxes having the cardinality of 2^{17}. Deploying *m*/4 S-Boxes, results with an S-Box cardinality:

$$\#S\text{-}Boxes=2^{17*m/4} \tag{5}$$

Therefore, the cardinality of the proposed KSGs as SUCs is:

$$\#KSG = 2^{2n-2+17*m/4} \tag{6}$$

In comparison, the initial KSG with pure T-function as shown in Fig. 2 would result with a SUC cardinality of only 2^{2n-2}.

In the next section, a sample realization for $n=68$ and $m=12$ is demonstrated. This results with $\#KSG=2^{2n-2+17*12/4}=2^{185}$. The parameter selections are optimized to show the efficiency of the resulting RSC-SUC in this particular technology.

V. Hardware Complexity

The main objective of this work was targeting *ultra-low cost* realization of SUCs as Clone-resistant units to be embedded in non-volatile electronic units for providing permanent, secret/unknown, non-removable and tamper-proof physical identities. The SUC creation-process requires relatively small software program that should run in the initialization process as a single-event-process in each unit. Microsemi SmartFusion®2 SoC FPGA technology was selected as a suitable environment for our proposal; it includes flash-based (permanent) logic fabric incorporated in the same chip with an ARM Cortex™-M3 processor. The main motivation for using multiplications in the ring of integers modulo 2^n is that Microsemi SoC chips include also programmable DSP hard-cores having hardwired multipliers allowing accumulation options embedded in the FPGA fabric. Each Mathblock contains also an adder, carry in and carry out such that multiplication can be performed for wide integers.

Fig. 5 describes a proposed architecture for the created KSG using the T-function $f(x) = x+(x^2 \vee C)$ with $n=68$ bits, deploying three 4-to-4 bit S-Boxes such that $m=12$ *bits.*

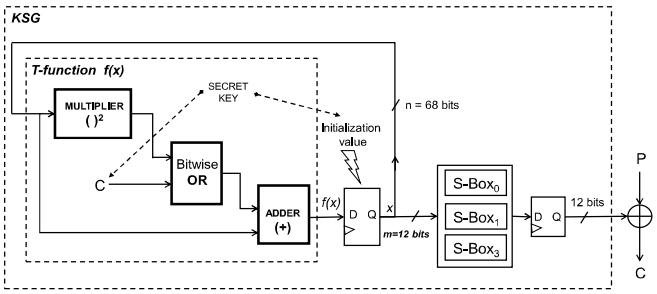

Fig. 5. General structure of T-Function $x+(x^2 \vee C)$ based KSG

A. Multiplier Implementation

Microsemi SmartFusion®2 SoC FPGAs incorporate mathblocks performing multiplications of two 17 bits unsigned numbers. Each Mathblock contain also an adder, carry in and carry out such that larger multipliers can be constructed by cascading such units. To perform wide multiplications, a 17 bits shift is embedded in each Mathblock and can be activated to shift the result of the previous Mathblock before performing addition.

In this section, the adopted concept is shown for realizing

wide multiplication between two n-bit integers by deploying such mathblocks in modern Microsemi FPGAs. The number of required 17-bit mathblocks for n-bits multiplier precision is computed as follows:

Lemma: The number of 17x17 bits multipliers required to perform a single-cycle multiplication between two n bits wide integers in the ring of integers Z_{2^n} is:

$$N-Mul=\sum_{i=1}^{k} i = k\times\frac{[k+1]}{2}$$
$$Where: k=\frac{n}{17} \quad (7)$$

Proof:

Let $a_i = 2^{17i} A_{17i...17i+16}$ and $b_i = 2^{17i} B_{17i...17i+16}$, where A and B are two n-bits integers with $n=17k, k\in Z$

A and B can be expressed as follows:

$$A = a_{k-1}+a_{k-2}+a_{k-3}+.....+a_2+a_1+a_0$$
$$B = b_{k-1}+b_{k-2}+b_{k-3}+.....+b_2+b_1+b_0$$

where a_i and b_i denotes 17 bits of A and B respectively, and a_ia_j or b_ib_j denotes the concatenation operations between a_i and a_j or b_i and b_j respectively. Multiplying A by B yields:

$$A\times B = \left(a_{k-1}+a_{k-2}+a_{k-3}+.....+a_2+a_1+a_0\right) \times\left(b_{k-1}+b_{k-2}+b_{k-3}+.....+b_2+b_1+b_0\right)$$

Reducing the integer result in Z_{2^n}, yields:

$$\begin{aligned} &A\times B \bmod 2^n = \\ &a_0b_0+a_0b_1+a_0b_2+.....+a_0b_{k-3}+a_0b_{k-2}+a_0b_{k-1}+ \\ &a_1b_0+a_1b_1+a_1b_2+.....+a_1b_{k-3}+a_1b_{k-2}+ \\ &a_2b_0+a_2b_1+a_2b_2+.....+a_2b_{k-3}+ \\ &...+ \\ &a_{k-1}b_0 \end{aligned}$$

That is, the number of multipliers required for computing $A\times B \bmod 2^n$ is $\sum_{i=1}^{k} i$ *q.e.d*

Each mathblock includes a 17 bits shift at one of the adder inputs; which can be activated or deactivated to perform the operations shown in the previous proof. For example a_0b_0+ a_0b_1 requires two multiplications and one addition where a_0b_0 are the 17 LSB bits and a_0b_1 are the 17 MSB bits. In this case, before adding the carry out of a_0b_0 to a_0b_1, this carry out should be shifted once by 17-bits.

For the sample implementation of Fig. 5, n=68=17×4 bits, k=68/17=4 that is *N-Mul* = 4x5/2 =10 multipliers. In the following, the required operations for performing the squaring operation by deploying 17 bits mathblocks are shown:

Let $X = x_3+x_2+x_1+x_0$, where $x_i = 2^{17i} X_{17i...17i+16}$, with $X_{i...j}$ denotes the concatenation of bits i to j of X.

Thus,

$$\begin{aligned} X^2 \bmod 2^N &= (x_3+x_2+x_1+x_0)\times(x_3+x_2+x_1+x_0) \\ &= x_3x_0+ \\ &\quad x_2x_1+x_2x_0+ \\ &\quad x_1x_2+x_1x_1+x_1x_0+ \\ &\quad x_0x_3+x_0x_2+x_0x_1+x_0x_0 \end{aligned}$$

Fig. 6 shows the hardware structure required for squaring 68 bits unsigned integer by deploying only 17 bits Mathblocks according to the proposed scheme.

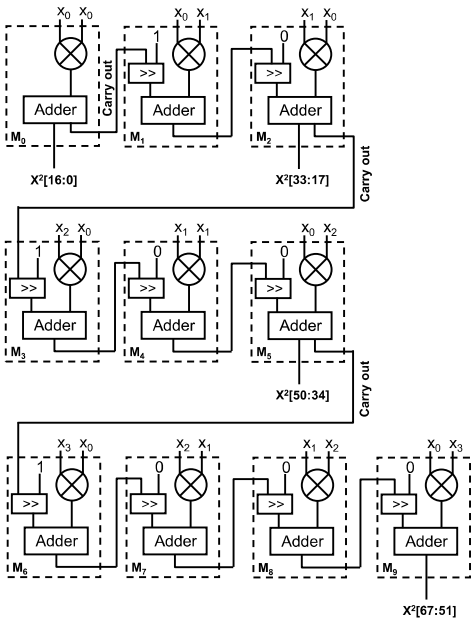

Fig. 6. Squaring n=68-bit integers in Z_{2^n} by deploying FPGA Mathblocks

B. Implementatig a Combined Addition and Bitwise OR Operation

SmartFusion®2 SoC FPGAs incorporate Logic Elements (LE), where each LE integrates 4-bits LUT and a D-Flip Flop (DFF). Each LUT has a carry-in and carry-out bit allowing cascading arithmetic operations such as addition. Fig. 7 shows a RTL level description of a combined implementation of *n*-bits adder with *n*-bits bitwise-OR operation.

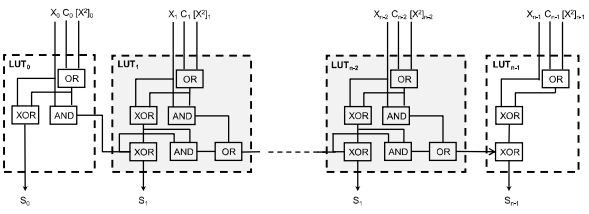

Fig. 7. Block diagram of the FPGA implementation of the Bitwise OR and Adder components using 4-bits Look Up Tables

C. S-Boxes

The required S-Boxes consume 4 LUTs each, resulting with

a total of 12 LUTs needed. The S-Boxes outputs are register buffered to provide best performance control, each output bit requires one DFF, and hence the S-Boxes and their output storages require just one logic cluster that is 12 Logic elements where each logic element includes one LUT and one DFF.

D. Hardware complexity

Mass products require lightweight identification mechanisms for economic reasons. SUCs as identities can gain special interest when being able to implement at possibly zero-cost. Most FPGA applications don't consume the total resources of the deployed FPGA, especially Mathblocks when the applications don't require digital signal processing. In such cases, our proposed SUC unit requires very few FPGA resources and can be added incrementally to the existing application for zero-costs when consuming unused FPGA resources.

The designed RSC-SUC was modeled in VHDL and synthesized to check its hardware complexity and performance. Mentor Graphics Modelsim ME design tool was used for simulation and Synplify pro ME for synthesis. When implementing the proposed KSG as in Fig. 5, the Mathblocks and their logic interfaces (MLI), Look-Up Tables (LUT) and D-Flip Flops (DFF) constitute the basic consumed resources. Table I shows the resources consumed for different devices of SmartFusion®2 SoC FPGA family. The KSG requires 10 MLI, 81 LUTs and 80 DFFs, this can be considered as zero cost in many real world applications.

TABLE I

HARDWARE COMPLEXITY

SmartFusion®2 SoC FPGA Family devices	*Resources usage*		
	#MLI=10 % of usage	*#LUTs=81 % of usage*	*#DFFs=80 % of usage*
M2S005	90.9	1.33	1.32
M2S010	45.45	0.67	0.66
M2S025	29.41	0.29	0.29
M2S050	13.88	0.14	0.14
M2S060	13.88	0.14	0.14
M2S090	11.9	0.09	0.09
M2S150	4.16	0.05	0.05

VI. CONCLUSIONS

Secret Unknown Cipher (SUC) is a digital PUF-like clone-resistant module that can be used as an ultra-low-cost "digital-substitute" to the traditional analog PUFs. SUCs, as digital modules have no drawbacks in terms of inconsistency in aging and sensitivity to operation conditions like temperature and supply power. SUCs can be deployed in various applications requiring secure clone-resistant physical identity chains. In this paper, we presented an implementation scenario for a randomized low-complexity SUC based stream cipher requiring very few Mathblocks and logic elements in Microsemi SmartFusion®2 SoC FPGAs. The concept can generally be deployed in all SoC systems having non-volatile FPGA logic area. By deploying such hardwired unknown stream ciphers; cloning attacks may be counteracted in mass-products such as in automotive and consumer products. Thus protecting the often abused intellectual property rights at reasonable or even no costs at all in some cases.

ACKNOWLEDGMENT: The authors would like to thank Microsemi, USA, DAAD and VW-Germany for supporting this basic research activity.

REFERENCES

[1] R. Pappu, "Physical One-Way Functions. Massachusetts Ave," 2001.

[2] R. Maes and I. Verbauwhede, "Physically unclonable functions: A study on the state of the art and future research directions," in Towards Hardware-Intrinsic Security, Springer, 2010, pp. 3–37.

[3] G. E. Suh and S. Devadas, "Physical unclonable functions for device authentication and secret key generation," in Proceedings of the 44th annual design automation conference, 2007, pp. 9–14.

[4] U. Rührmair, F. Sehnke, J. Sölter, G. Dror, S. Devadas, and J. Schmidhuber, "Modeling attacks on physical unclonable functions," in Proceedings of the 17th ACM conference on Computer and communications security, 2010, pp. 237–249.

[5] X. Xu and W. Burleson, "Hybrid side-channel/machine-learning attacks on PUFs: a new threat?," Proc. Conf. Des., 2014.

[6] D. Merli, D. Schuster, F. Stumpf, and G. Sigl, "Side-Channel Analysis of PUFs and Fuzzy Extractors.," Trust, 2011.

[7] W. Adi, "Clone-resistant DNA-like secured dynamic identity," in Bio-inspired Learning and Intelligent Systems for Security, 2008. BLISS'08. ECSIS Symposium on, 2008, pp. 148–153.

[8] A. Klimov and A. Shamir, "Cryptographic applications of T-functions," in International Workshop on Selected Areas in Cryptography, 2003, pp. 248–261.

[9] A. Klimov, A. Shamir, and others, "A new class of invertible mappings," in CHES, 2002, vol. 2, pp. 470–483.

[10] Microsemi, "SmartFusion2 SoC FPGA Family." [Online]. Available: https://www.microsemi.com/products/fpga-soc/soc-fpga/smartfusion2.

[11] W. Luis, G. Newell, and K. Alexander, "Differential power analysis countermeasures for the configuration of SRAM FPGAs," Conf. MILCOM 2015, 2015.

[12] V. Anashin, "Uniformly distributed sequences of p-adic integers, II," arXiv Prepr. math/0209407, 2002.

[13] B. Liu, Z. Gong, W. Qiu, and D. Zheng, "On the security of 4-bit involutive s-boxes for lightweight designs," Int. Conf. Inf., 2011.

[14] G. Leander and A. Poschmann, "On the Classification of 4 Bit S-Boxes," pp. 159–176, 2007.

[15] S. Skorobogatov and C. Woods, "Breakthrough silicon scanning discovers backdoor in military chip," in International Workshop on Cryptographic Hardware and Embedded Systems, 2012, pp. 23–40.