AKARI-X: a pseudorandom number generator for secure lightweight systems

Honorio Martín, Enrique San Millán, Luis Entrena
Electronic Technology Department
Carlos III University
Leganés, Spain
hmartin, quique, entrena@ing.uc3m.es

Pedro Peris López
Information and Communication Theory Group
Delft University of Technology
Delft, The Netherlands
P.PerisLopez@tudelft.nl

Julio César Hernández Castro
School of Computing
University of Portsmouth
Portsmouth, United Kingdom
julio.hernandez-castro@port.ac.uk

***Abstract* [1]— In order to obtain more secure and reliable systems, the vast majority of RFID protocols include a Pseudorandom Number Generator (PRNG) in its design. However, the authors often do not specify the PRNG to use and standard solutions exceed the capabilities of low-cost RFID tags. In this paper, we propose two lightweight PRNGs (AKARI-1 and AKARI-2) that meet the requirements of these systems while improving their reliability and security. They may be supported on commercial tags of low price.**

Keywords: Security, Reliability, PRNG, RFID, ASIC, VHDL, Lightweight

I. INTRODUCTION

Radio Frequency Identification (RFID) is an automatic identification technology in which a small transponder (tag), attached to an object, receives and responds to radio-frequency queries from a transceiver (reader). The main advantage of the RFID compared to other identification systems is not only that it is capable of identifying items of different types, but it also distinguishes between items of the same type without mistake, while not requiring physical or visual contact.

The way of operation of the RFID systems is simple. The RFID tag contains some identification information for an object and generates a signal of radio frequency with the mentioned information. This signal can be received by a RFID reader, which reads the information and transforms it into a digital format to the specific application that RFID uses. The most common method of reading the RFID tags is the one that has been named as 'inductive connection' in the passive tags. In this method the reader's antenna creates a magnetic field in a nearby area. The energy generated by this field is used by the tag to return a signal to the reader containing the information stored in the tag. The reader transmits this information to an application that associates the identifier stored in the tag with the information of the product to that the tag. Once it has been processed, this information is transmitted to the management systems that update the information of corresponding inventory [1].

The implantation of the RFID tags will depend on the cost of the implementation of these. At present, the cost of a RFID tag is about 0.20 €, so for this technology to be competitive on the market it would be necessary to obtain a cost lower than 0.10 € [2]. To reduce the cost of the RFID systems, it is necessary to reduce the silicon area that is used in them. This reduction in area must not suppose a reduction in the security of the tags.

Tags usually respond with a constant value, which facilitates their association with their holders. So an attacker may track user's movements putting location privacy at risk. The inclusion of random numbers in tag answers is necessary – but not sufficient – to thwart traceability attacks.

Generally, two main approaches may be considered for random number generation. A Random Bit Generator (RBG), based on a physical source such as thermal noise of zener diodes or radioactive decay, can be used to generate truly random numbers. Due to its inefficiency and the impossibility of replicating their outputs at the other side of the communication channel, even starting from identical initial states, RBGs are commonly substituted by Pseudorandom Number Generators (PRNGs). Linear Congruential Generator (LCG) and Linear Feedback Shift Register (LFSR) are the most well-known alternatives in this category. Although their

[1] This work is partially supported in part by the Spanish Ministry of Science and Technology, code TEC2010-22095-C03-03)

978-1-4577-1056-8/11/$26.00 ©2011 IEEE

2011 IEEE 17th International On-Line Testing Symposium

outputs have good statistical properties, these PRNGs are cryptographically insecure.

Actually, few PRNG proposals have been presented in the scientific literature specifically designed for EPC tags. To the best of our knowledge, only some papers explicitly propose PRNG for EPC tags. On one hand, Peris-Lopez et al. present in [12] a deterministic algorithm that relies on the use of 32-bit keys and pre-established initial states. The set of functions mainly consists of bit rotations, bitwise operations, and modular algebra, building a 32-bit PRNG. The authors also propose an alternative 16-bit version of their PRNG for EPC Gen2 compatibility. To reduce the output length from 32 to 16 bits, Peris et al. divide the 32-bit output in two halves and XOR them to obtain the 16-bit output sequence.

On the other hand, Che et al. describe in [13] a hybrid approach that combines the use of Linear Feedback Shift Registers (LFSR) and physical properties to build random sequences. A similar idea has been also used in [14] to design a PRNG. In this case the authors handle the inherent linearity of LFSRs by means of a multiple-polynomial approach. The authors present a secure PRNG design suitable to the current EPC Gen2 technology, providing evidences of statistical and hardware compatibility.

In this paper, we pursue the design of new PRNGs that satisfy the two alluded requirements: 1) good statistical properties; 2) adequate for the intended security applications. Additionally, hardware limitations need to be taken into consideration to be compliant with the requirements of low-cost RFID tags,. That is, a tiny footprint, high throughput and low power consumption have to be met by the proposed designs.

II. LIGHTWEIGHT PRNG

In 2004, Klimov and Shamir introduced the concept of T-functions. A T-function is a mapping from m-bit words to m-bit words in which, for each $0 \leq i \leq m$, bit i of the output can depend only on bits 0, 1, … i of the input. All the bitwise operations (e.g. bitwise XOR ($a \oplus b$), OR ($a \vee b$) and AND ($a \wedge b$)) and most of the modern machine operations (e.g. addition ($a + b \bmod 2m$) and multiplication ($a \cdot b \bmod 2m$)) are T-functions and their composition are also T-functions.

Particularly, the mapping $x \rightarrow x + (x2 \vee C) \pmod{2m}$ results very interesting [3]. For any m, it is a permutation with a single cycle of length 2m if both the least significant bit and the third significant bit in the constant C are 1. If we analyze the output provided by the permutation $x \rightarrow x + (x2 \vee C) \pmod{2m}$, we observe that its output looks as a random variable. However, this function is not cryptographically secure as same as LCGs and LFSRs. More precisely, an attacker can exploit the fact that when a T-function is executed there is not propagation of information from the left to the right, which facilitates its cryptanalysis.

In our design, we include a non-linear filter function to overcome the above mentioned drawback and to guarantee a high degree of diffusion. The possible candidates are obtained through evolving compositions of extremely light operands – in terms of computation and hardware demands – by means of genetic programming. We use the Strict Avalanche effect to measure the non-linearity of each candidate. For a detailed description of the methodology used, we refer the reader to [4]. Specifically, we analyze two alternatives. AKARI-1 uses a filter function, which is iterated a relative high number of times (r = 64). In AKARI-2, two filter functions are mixed, facilitating the reduction of the number of iterations in the loop (r = 24). Finally, in both cases the sequence of lower halves (bottom m/2 bits) of the m output bits represents the final output. The pseudo-code of the two lightweight PRNGs proposed is displayed in Figure 1, where (>>) and (<<) symbolizes right and left circular shift, respectively.

AKARI-1

```
Initialize x0 and x1 of m-bits
    x0 = x0 + ((x0 * x0)  ∨ 5)
    x1 = x1 + ((x1 * x1)  ∨ 13)
    z = x0
for r from 0 to 63
    z = (z >>1) + (z << 1) + z + x1
%Output m/2 bits
   Lower half of z
```

AKARI-2

```
Initialize x0 and x1 of m-bits
    x0 = x0 + ((x0 * x0)  ∨ 5)
    x1 = x1 + ((x1 * x1)  ∨ 13)
    z = x0 ^ x1
for r from 0 to 24
    z = (z << 1) + ((z + (0x56AB0A)) >1)
    y = x1 ^ z
for r from 0 to 24
    y = (y >> 1) +  (y << 1) + y + (0x72A4FB))
    z = z ^ y
%Output m/2 bits
    Lower half of z
```

Figure 1. Pseudo-code of AKARI-1 and AKARI-2.

We use ENT [5], DIEHARD [6] and NIST [7] randomness test batteries to analyze the sequences generated by AKARI-1 and AKARI-2. From the results, we can conclude that these outputs do not look different from a random variable (see Table 1). From the security point of view, the use of filter/s functions

guarantees the non-linearity of the function. That is, it assures good diffusion effect in the PRNG.

III. IMPLEMENTATION ARCHITECTURES

Usually, the number of gate equivalents (GE) devoted to security in low-cost tags cannot exceed 4,000 GE [8, 9]. The operation frequency is often set to 100 KHz and the number of estimated clock cycles for the generation of a random number is around 500 or 600 cycles (e.g. EPC tags [10]). The power consumption is also a very important part of the design [11] and should not exceed the range of microwatts. We present several different architecture implementations for two PRNGs, AKARI-1 and AKARI-2 that attempt to fulfill these requirements.

		AKARI-1	**AKARI-2**
ENT	Entropy	8.000000	8.000000
	Compression Rate	0%	0 %
	χ^2 Statistic	259.09 (41.70%)	250.99 (55.93%)
	Arithmetic Mean	127.4976	127.5031
	Monte Carlo π estimation	3.141447036 (error 0.00%).	3.141512474 (error 0.00%).
	Serial correlation coefficient	-0.000026	0.000013
Diehard	Overall p-value	0.352645	0.551129
NIST[1]		Pass	Pass

Table 1: Test results obtained with ENT, Diehard and NIST suits (m = 32 bits)

1Due to the huge amount of p-values generated, the report can be downloaded from http://www.lightweightcryptography.com/research/AKARI/AKARI.html

The goal of the proposed architectures is meeting the different technology requirements of RFID while trying to maximize the security, in our case trying to be able to use a higher number of bits for the PRNGs. We have focused on the area constraint, because if we reduce area, then we can exploit this reduction to add additional bits for the operations, or alternatively we can just reduce the cost of silicon area.

As the starting point is a lightweight algorithm, reducing the area is not an easy task, because all the involved operations are already very simple and have very low area cost. However, when the number of bits increases, the area increases as well and then there is some headroom for some optimizations. This is the case for example for adders, which can grow too much when increasing the number of bits of the operands, and get way beyond the limits of the area constraint. We propose our architectures with these optimizations in mind.

A. AKARI-1implementation

The first architecture (AKARI-1A) tries to minimize the number of clock cycles. Therefore, in this architecture, it is necessary to run each operation in only one cycle when it is feasible. For that purpose, different m-bit operation blocks are used, where m is the bit length of the variables. The control of the inputs and outputs of each block is implemented by means of a Finite State Machine (FSM).

The second proposed architecture (AKARI-1B), attempts to reduce the chip area. One of the alternatives for reducing area is the use of an adder with half of the number of bits (m/2). In this case, the benefits in area have a penalty in throughput. Similarly, the control is implemented with a FSM. As mentioned, in this architecture, the circuit needs more clock cycles because the result of each sum is obtained now in 3 cycles, instead of only 1. Figure 2 shows a scheme of the adder and auxiliary register of the second architecture proposed for AKARI-1. We can observe in figure 2 that using an adder of m/2 bits involves the use of some additional multiplexers. For a good architecture design we need to find a balance between the size of the adder and the number of multiplexers that we need add. In the synthesis results section we can observe that for a low number of bits (less than 16 bits), the area that we improve with the reduction of the adder, is smaller than the area occupied by the additional necessary logic.

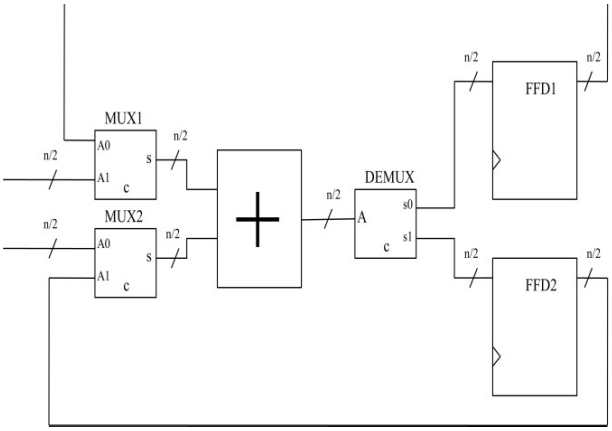

Figure 2. Scheme of the adder and auxiliary register in AKARI-1B

B. AKARI-2 implementation

For this second PRNG, we present 3 different architectures. The first implementation (AKAR1-2A) attempts to minimize the number of clock cycles. As in AKARI-1A, it performs the main operations in only one cycle using a big adder.

The second architecture (AKARI-2B) tries to improve the occupied area, and for this reason, the same strategy as in

AKARI-1B is used: using an adder with the half the number of bits.

Finally, a third architecture (AKARI-2C) is proposed, trying to reduce even more the area at the cost of more clock cycles. In this case, the area of the adder is divided by 4.

IV. SYNTHESIS RESULTS

The results have been obtained using the program Synopsys, with library Faraday 90 of UMC (90nm.) which contains information of the layout of the basic cells and allows us to have a good estimation of the area and consumption of the circuit. All the tests have been performed for an operation frequency of 100 KHz. Power supply is 1.2 V - imposed by the library. The final results have been obtained using a medium effort in map, area and consumption, because with these options we obtained the best results.

As the Faraday 90 is a manufacturing library that provides information about the layout of the basic cells, we can obtain a lot of interesting information that cannot be found with other generic libraries, commonly used in other works by other authors. With this manufacturing library the results that we present should very similar to the results that we would obtain in a manufactured circuit.

A. AKARI-1 Results

In tables 3, 4, 5 and 6 we present the synthesis results for the two implementations with different number of bits, where we show:

- Area μm^2 : is the area of the full circuit in μm^2
- Area GE: the area expressed in *Gate Equivalents*. The number of *Gate Equivalents* is obtained by dividing the area of the circuit by the area of a basic logic NAND gate. This number gives an independent of the technology estimation of the area of the circuit.
- Power (nW): this value gives an estimation of the power consumption of the circuit considering the model of the manufacturing library we are using.
- Throughput (kbps): gives a measure of how fast is the PRNG calculating values.

Area µm²	8 bits	16 bits	32 bits	64 bits	128 bits	256 bits	512 bits
AKARI1A	835	1.494	3.191	6209	12224	24340	48563
AKARI1B	1016	1643	2892	5484	10669	20912	41406

Table 3: Area in µm² for AKARI-1 PRNG

Area GE	8 bits	16 bits	32 bits	64 bits	128 bits	256 bits	512 bits
AKARI1A	266	476	1.018	1.980	3.898	7.762	15.486
AKARI1B	324	524	922	1.749	3.402	6.669	13.204

Table 4: GE for AKARI-1 PRNG

Power(nW)	8 bits	16 bits	32 bits	64 bits	128 bits	256 bits	512 bits
AKARI1A	26,7	47,35	89,95	173,8	343,2	712,17	1410
AKARI1B	33,95	54,61	95,71	182,32	350,36	710, 2	1460

Table 5: Power comsumption for AKARI-1 PRNG

Throughput (kbps)	8 bits	16 bits	32 bits	64 bits	128 bits	256 bits	512 bits
AKARI1A	12,12	24,24	48,48	96,96	193,92	387,84	775,68
AKARI1B	1,77	3,55	7,11	14,22	28,44	56,88	113,77

Table 6: Throughput for AKARI-1 PRNG

With the two proposed architectures we can choose between the generation of a random number consuming a minimal number of clock cycles, 66, or an improvement in area of up to 15 % for 512 bits. The strategy of reduction improves for higher number of bits, because the more bits we have the less impact the additional logic and flip-flops required for the sequential adder has. With future improvements in fabrication technology it is expected that more GEs could be used in the same area, so this approach would give even better improvements.

In figure 3 we can see the GE obtained for PRNG AKARI-1 for different number of bits. As we introduced previously, for a low number of bits (less than 16 bits), the area occupied by the added multiplexers and flip-flops is bigger than the area improved by the adder reduction. For that reason in figure 3, we can observe that for less than 32 bits, the number of GE for the architecture AKARI-1A is smaller than the architecture AKARI-1B, and the improvements start to show at 32 bits.

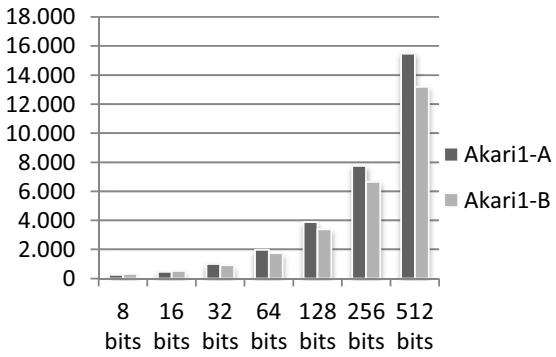

Figure 3. GE AKARI-1

In figure 4, we present the main parameters (GE, Throughput and Power consumption) for both AKARI-1 architectures for the three more interesting cases that meet the area requirements of low cost RFID: 32, 64 and 128 bits.

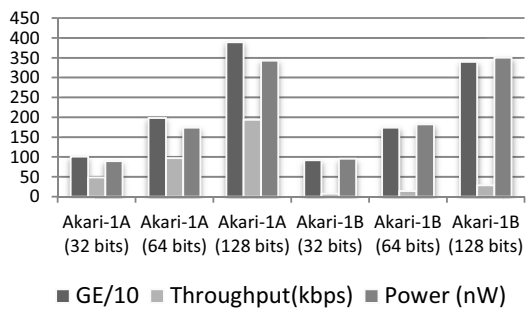

Figure 4. Main parameters for AKARI-1

In conclusion for AKARI-1, as shown in Table 7, the results obtained in the first architecture (AKARI-1A) fulfills the minimal area requirements while the number of necessary clock cycles is quite below the limit, for an implementation of maximal = 128 bits. In the second architecture (AKARI-1B), we attempt to reduce the area using an adder of half size in comparison with the previous architecture.

128 bits	Gate Equivalents	Power (nW)	Clock cycles
AKARI-1A	3898	343	66
AKARI-1B	3402	350	450

Table 7: AKARI-1 PRNG (m_{aximal} = 128 bits)

For an application that requires a high throughput we should choose the architecture AKARI-1A that meets with EPC standard for 128 bits or less. For an application where area consumption is important, we should choose the architecture AKARI-1B that improves the area a 15%.

B. AKARI-2 Results

In tables 8, 9, 10 and 11 we present the synthesis results for the three implementations of Akari-2 for different numbers of bits.

Area μm^2	8 bits	16 bits	32 bits	64 bits	128 bits	256 bits	512 bits
AKARI2A	1310	2582	5837	11740	23462	46955	93257
AKARI2B	1639	2794	5173	10014	19656	38641	76910
AKARI2C	1650	2831	5081	9534	18421	36241	71579

Table 8: Area in μm^2 for AKARI-2 PRNG

Area GE	8 bits	16 bits	32 bits	64 bits	128 bits	256 bits	512 bits
AKARI2A	418	824	1.861	3.744	7.482	14.973	29.738
AKARI2B	523	891	1.650	3.193	6.268	12.322	24.525
AKARI2C	526	903	1.620	3.040	5.874	11.557	22.825

Table 9: GE for AKARI-2 PRNG

Power(nW)	8 bits	16 bits	32 bits	64 bits	128 bits	256 bits	512 bits
AKARI2A	31,92	57,38	109,88	216,06	439,37	902,49	1790
AKARI2B	45,78	76,95	135,81	255,28	522,04	1070	2300
AKARI2C	41,52	72,33	126,02	231,25	454,34	924,81	1810

Table 10: Power comsumption for AKARI-2 PRNG

Throughput (kbps)	8 bits	16 bits	32 bits	64 bits	128 bits	256 bits	512 bits
AKARI2A	15,68	31,37	62,74	125,49	250,98	501,96	1003
AKARI2B	2,75	5,5	11,03	22,06	44,13	88,27	176,5
AKARI2C	1,5	3,01	6,03	12,07	24,150	48,30	96,60

Table 11: Throughput for AKARI-1 PRNG

For this second generator we can choose between an architecture that consumes few clock cycles, AKARI-2A (51 cycles), or an architecture that improves the area penalizing the consumption of cycles of clock, AKARI-2C. Also we can choose an architecture in the middle in which we improve the area, with a small cost on the number of clock cycles, AKARI-2B. The improvement in area that can be reached with the architecture AKARI-2C with respect to architecture AKARI-2A is 24 %.

In figure 5 we present the GE for several number of bits and for different architectures that we propose for AKARI-2. We can observe the same effect for low number of bits as in AKARI-1. The new adder with the additional required multiplexers and flip-flops needs more area than the original adder with twice the size.

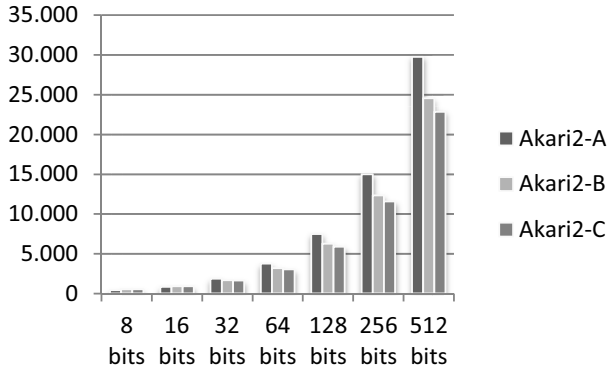

Figure 5. GE AKARI-2

In figure 6, we present the main parameters (GE, Throughput and Power consumption) for two AKARI-2 architectures.

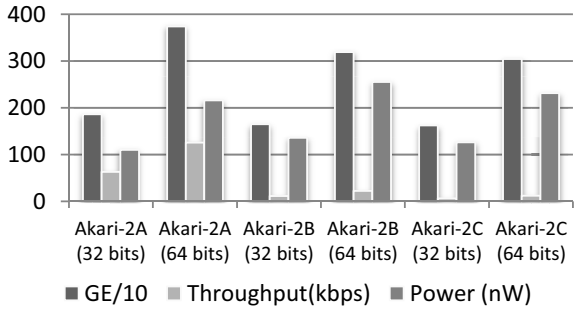

Figure 6. Main parameters for AKARI-2

In conclusion for AKARI-2, we observe that in AKARI-2A the restrictions in area and clock cycles are met for implementations of maximal = 64 bits (see Table 12). In comparison with AKARI-1A, we observe that this algorithm consumes approximately the same area, but it is a bit faster due to the lower number of rounds in the filter functions. In AKARI-2B and contrary to what happens in AKARI-1A, we obtain a slight reduction in the area and power consumption. Finally, for those applications with low throughput demands, AKARI-2C results the most convenient option.

64 bits	Gate Equivalents	Power (nW)	Clock cycles
AKARI-2A	3743	216 nW	51
AKARI-2B	3193	255 nW	290
AKARI-2C	3040	231 nW	530

Table 12: AKARI-2 PRNG (m_{aximal} = 64 bits)

For an application that needs a high throughput, we should use the architecture AKARI-2A, and for an application that requires a low cost in area, we should choose the architecture AKARI-2C. Finally we would recommend the architecture AKARI-2B for an improvement in area with a medium throughput.

V. COMPARISON WITH OTHERS PRNGS

Existing commercial Gen2 tags do implement a PRNG, as it is an EPC standard mandatory, but companies are often reluctant to present the design of their PRNGs. Manufacturers simply refer to testbenches that show the accomplishment of some expected requirements, most of them for compatibility purposes.

We have searched for other implementations of other proposed PRNGs suitable for RFID, and in them we have only found estimations of the hardware results without the use of a library with information of the cells in the real layout of the circuit, and therefore these results may not correspond with the real generated hardware.

For example, in [14] authors calculate an estimation of 761 GE for a 16-bit implementation. With our approach we can use a lightweight secure 16bit PRNG using 476 GE for AKARI-1 or 824 GE for AKARI-2.

We have not found other works that present power consumption results for this kind of PRNGs, and for this reason the comparison in this aspect is not possible.

VI. CONCLUSIONS

In this paper, we have presented two secure lightweight PRNGs suitable for constrained devices such as low-cost RFID tags or sensor nodes are proposed. Several architectures have been proposed for them, focusing on getting the smaller possible footprint or the maximum possible throughput when no area improvement is possible. Although we did not obtain a manufactured circuit for them, we used a manufacturing library with information of the layout of the basic cells that allows us to provide some good estimations of the area and power consumption of the circuit.

With the smaller footprint that we get with some of the architectures we can increase the number of bits of the PRNGs while meeting the low cost RFID requirements. Increasing the number of bits results in an improvement on the security of the PRNG.

We have obtained a maximum number of 128 bits for AKARI-1 or a maximum of 64 bits for AKARI-2. With the proposed architectures we need around 3000-4000 GEs, meeting therefore the EPC requirements.

Due to the tiny footprint, the feasibility or potential use of these PRNGs in commercial products, such as EPC Gen-2 tags, is beyond question.

VII. REFERENCES

[1] P. Peris, J.C. Hernandez, A. Ribagorda, “Lightweight Cryptography in Radio Frequency Identification (RFID) Systems”

[2] W. Sean and L. Thomas. Automatic identification and data collection technologies in the transportation industry: BarCode and RFID. Technical report, 2001.

[3] A. Klimov and A. Shamir. A new class of invertible mappings. In Cryptographic Hardware and Embedded Systems (CHES), volume 2523 of LNCS, pages 471-484. Springer-Verlag, 2002

[4] J. C. Hernandez-Castro, J. M. Estevez-Tapiador, A. Ribagorda-Garnacho, B. Ramos-Alvarez. Wheedham: an automatically designed block cipher by means of genetic programming. In Proc. of IEEE Congress on Evolutionary Computation, pages 192–199. IEEE Computer Society, 2006.

[5] J. Walker. Randomness Battery, http://www.fourmilab.ch/random/, 1998.

[6] G. Marsaglia. The Marsaglia Random Number CDROM Including the DIEHARD Battery of Tests of Randomness, http://stat.fsu.edu/pub/diehard, 1996.

[7] A. Rukhin, J. Soto, J. Nechvatal, M. Smid, E. Barker, S. Leigh, M. Levenson, M. Vangel, D. Banks, A. Heckert, J. Dray and S. Vo. A statistical test suite for random and pseudorandom number generators for cryptographic applications, http://csrc.nist.gov/rng/. Technical Report, 2001.

[8] S. Karthikeyan and M. Nesterenko. RFID security without extensive cryptography. In Proc. of SASN’05, 2005.

[9] A. Juels and S. Weis. Authenticating Pervasive Devices with Human Protocols.In V. Shoup, editor, Advances in Cryptology - Crypto 05, LNCS 3126, 293-198, Springer- Verlag, 2005.

[10] EPCGlobal, “Class-1 generation 2 UHF air interface protocol standard version 1.2.0: "Gen 2", http://www.epcglobalinc.org/standards/, 2008.

[11] M. O'Neill, “Low-Cost SHA-1 Hash Function Architecture”.

[12] Peris-Lopez, P., Hernandez-Castro, J., Estevez-Tapiador, J. and Ribagorda, J. (2009). LAMED APRNG for EPC Class-1 Generation-2 RFID specification. Computer Standards & Interfaces, 31(1), 88–97.

[13] Che, W., Deng, H., Tan, X., and Wang, J. (2008). Chapter 16, A Random Number Generator for Application in RFID Tags. In Cole, P.H. and Ranasinghe, D.C. (Eds.), Networked RFID Systems and Lightweight Cryptography (pp. 279–287). Berlin: Springer-Verlag.

[14] Melia-Segui, J., Garcia-Alfaro J. and Herrera-Joancomarti, J. (2010). Analysis and Improvement of a Pseudorandom Number Generator for EPC Gen2 Tags. In Curtmola, R. et al. (Eds.), Financial Cryptography and Data Security 2010 Workshops, LNCS (pp. 34–46). Berlin: Springer-Verlag.