Dietary Recommendations for Lightweight Block Ciphers: Power, Energy and Area Analysis of Recently Developed Architectures

Lejla Batina[1,2], Amitabh Das[2], Barış Ege[1]
Elif Bilge Kavun[3], Nele Mentens[2,4], Christof Paar[3]
Ingrid Verbauwhede[2], Tolga Yalçın[3]

[1]Inst. for Computing and Information Sciences, Radboud University Nijmegen, The Netherlands
[2]ESAT/COSIC, KU Leuven & iMinds, Belgium
[3]Horst Görtz Institute for IT-Security, Ruhr University Bochum, Germany
[4]ACRO/ES&S, Katholieke Hogeschool Limburg, Belgium

Abstract. In this paper we perform a comprehensive area, power, and energy analysis of some of the most recently-developed lightweight block ciphers and we compare them to the standard AES algorithm. We do this for several different architectures of the considered block ciphers. Our evaluation method consists of estimating the pre-layout power consumption and the derived energy using Cadence Encounter RTL Compiler and ModelSIM simulations. We show that the area is not always correlated to the power and energy consumption, which is of importance for mobile battery-fed devices. As a result, this paper can be used to make a choice of architecture when the algorithm has already been fixed; or it can help deciding which algorithm to choose based on energy and key/block length requirements.

1 Introduction

In the past decade various proposals for "lightweight" symmetric ciphers have been made. Among more carefully investigated ones are Clefia [20], HIGHT [21], KATAN [4], mCrypton [22], and PRESENT [3]. This turned into a very active area of research as evident by several algorithms proposed over the course of the past two years, including KLEIN [18], LED [1], Piccolo [2] and PRINCE [16]. The dominant metric used in the majority of the proposals has been the number of gate equivalence, or GE, needed for realizing the cipher in hardware. This number is derived by dividing the silicon area used for a cipher with a given standard-cell library by the area of a two-input NAND gate. Hence, the popular gate equivalence count can be thought of as a normalized area measure. Even though helpful, the metric does not answer all questions regarding lightweight ciphers.

The purpose of the investigation at hand is to perform a comprehensive area, power, and energy analysis of some of the most recently-developed lightweight block ciphers along with the well-known block ciphers, which can be helpful for both the engineering and theoretical communities concerned with lightweight cryptography. Given that

lightweight algorithms are particularly interesting for battery-powered or passive systems such as RFID tags, a valid energy prediction is very desirable.

In the most recent work of Kerckhoff et al. [15], the area, power consumption, throughput and energy of 6 block ciphers are evaluated. Conclusions are drawn with respect to round unrolling, parallelism and pipelining. The paper at hand covers 11 lightweight block cipher architectures (of 7 different lightweight block ciphers) and 6 different AES architectures. The differences between the AES architectures are not limited to round unrolling, parallelism and pipelining, but are based on specific design choices. This gives a more fair comparison of the standardized AES to recently-developed lightweight block ciphers.

Other related works in the past were focused mainly on other, more specific aspects of low-cost applications. For instance, the work of Singelée et al. [14] focuses on the computation and communication energy budget of authentication protocols for active RFID tags. Accordingly, they consider other cryptographic primitives that can be used for authentication i.e. ECC-based protocols. On the other hand, the AES algorithm as the main standard for encryption in the past decade, has been already evaluated on the energy consumption in several previous studies [8, 11, 17]. In particular the work of Tilich et al. [17] examines several different AES S-box implementations that are based on three different design strategies. The results addressed the consequences of different design strategies on critical path delay, silicon area, and power consumption.

All those works contributed to the better understanding of low-cost design principles. As mentioned above, the requirements of extreme low-cost applications of today require more lightweight solutions as advocated by the new block ciphers' proposals. Our work extends those studies with an extensive suite of recent lightweight symmetric schemes.

The paper is organized as follows. In Sect. 2, the considered block ciphers are briefly revisited together with the specific architectures. Sect. 3 elaborates on our analysis methodology. Finally, Sect. 4 presents and discusses the results and Sect. 5 concludes the paper.

2 Background

In this section, we provide background information on the evaluated block cipher architectures. The information is grouped according to similarities in the architectures.

2.1 Parallel Implementation of Block Ciphers

We have implemented fully parallel versions of AES-128, CLEFIA-128, PRESENT-80, LED-128, KLEIN-64, mCrypton-96 and PRINCE-128, where the number next to each cipher represents the key length chosen for the implementation. Among these, AES and CLEFIA are 128-bit block ciphers, whereas all others are 64-bit. For a fair comparison, we have implemented the encryption-only version of each cipher. All the implemented block ciphers share the same structure. Any such block cipher can be implemented as shown in Fig. 1, where the round function is instantiated only once. In this case, the initial input (upon a start signal) of the round function is the sum of the

input key and the plaintext (i.e. the initial state). It is processed by the combinational round function and the next state is generated. It is then stored in the state register, whose output becomes the input to the round function in the next cycle. The iteration is as many rounds as the cipher is defined for. Finally, the ciphertext can be taken either from the state register output or some internal node of the round function block. In some cases, a final whitening key may also be added onto the value from the output node to generate the ciphertext (as in the case of PRINCE). The datapath width inside the round function block is equal to the cipher block size, i.e. 128 bits for AES and CLEFIA, 64 bits for all others, while the datapath width of the key scheduling block depends on the selected key size. In the case of LED, we use a fixed key (fixed means either the original input key, or a function of it). Fig. 2 illustrates the no key-update case for such a round-based implementation.

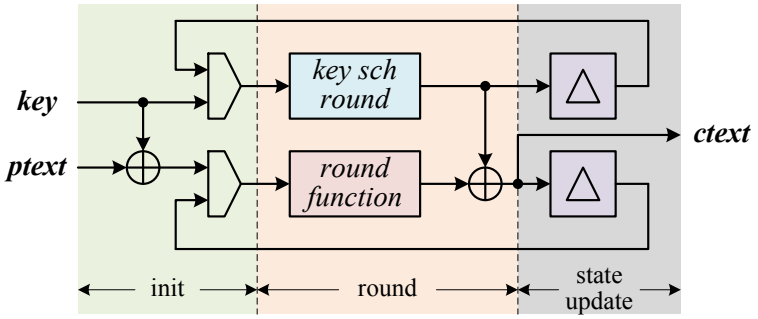

Fig. 1. Folded (round-based) implementation of a generic block cipher

This is basically the design strategy we used in all our parallel implementations. In order to keep the round numbers minimum, we got the ciphertext from internal nodes inside the round function instead of the outputs of the state registers. This way, it was ensured that the cipher could be run in maximal throughput, that is the distance between two consecutive starts is equal to the number of rounds.

We furthermore implemented parallel versions of AES in various flavors. Mainly, we focused on the S-box, which is the most area consuming unit inside the AES algorithm. The first implementation (AES_lut_128) uses lookup-table based S-boxes. The second version implements the S-box in the composite field $GF((2^4)^2)$ (AES_4_2_128), and the third version in the composite field $GF(((2^2)^2)^2)$ (AES_2_2_2_128), both as explained in [5]. We have also observed that the isomorphic transform matrices used for composite implementations are very area-consuming due to the high number of 1's inside, that correspond to XOR gates in hardware. In order to reduce these number of 1's,

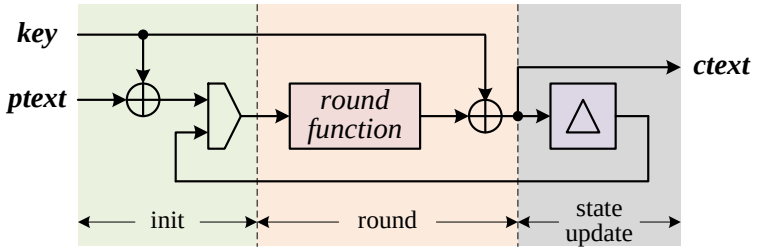

Fig. 2. Folded (round-based) implementation of a generic block cipher – no key schedule

it is possible to use other isomorphic matrices, that are affine equivalent to the original matrices. Of course, this requires that the corresponding affine and inverse affine transformations have to be applied to the plaintext and ciphertext, respectively. Similarly, the key scheduling has to be also carried on the affine equivalent "domain". Depending on the choice of the affine transformation, it is possible to reduce the area or the power consumption or both of the overall design. We chose transforms that would minimize the power consumption, and have implemented two more flavors of AES, namely affine-transformed in the composite field $GF((2^4)^2)$ (AES_iso_4_2_128), and affine-transformed in the composite field $GF(((2^2)^2)^2)$ (AES_iso_2_2_2_128).

2.2 Unfolded Implementation of PRINCE

Since PRINCE is originally designed for unfolded implementation, i.e. all round functions are realized within a single cycle without need for a state register, we have also added an unfolded version of PRINCE. It is basically a realization of PRINCE with all 12 rounds unfolded. In our unfolded implementation strategy, the key is added to the input plaintext to generate the initial state followed by various numbers of identical rounds in order to update the state. This is shown in Fig. 3 for the unfolded version of a generic three-round block cipher. However, as PRINCE uses a fixed key, we actually implement the no key-update version of this implementation. Fig. 4 illustrates this case.

2.3 Implementation of KATAN

KATAN is a block cipher that belongs to a family of small and efficient hardware-oriented block ciphers. KATAN ciphers include KATAN32, KATAN48, and KATAN64. All three ciphers use 80-bit keys and have a different block size (KATANn has an n-bit

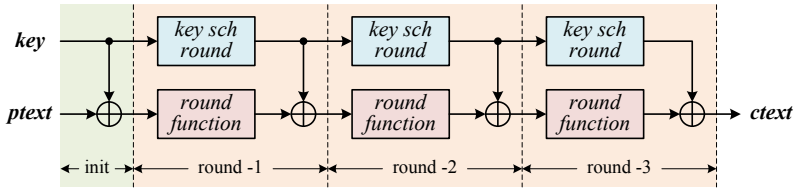

Fig. 3. Unfolded implementation of a generic 3-round block cipher

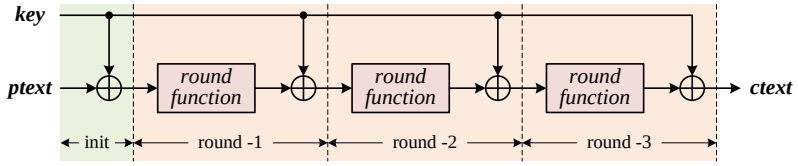

Fig. 4. Unfolded implementation of a generic 3-round block cipher – no key schedule

block size). All three block ciphers are highly compact,with the one having the smallest block size resulting in the smallest circuit area of only 801 GEs.

In KATAN, the plaintext is loaded in two registers. In each round, several bits are taken from the registers and entered in two nonlinear Boolean functions. The output of the Boolean functions is loaded to the least significant bits of the registers (after they are shifted). This is done in an invertible manner. To ensure sufficient mixing, 254 rounds of the cipher are executed. A round counting LFSR is used instead of a counter, for counting the rounds to stop the encryption after 254 rounds, and to introduce more diffusion as well. As there are 254 rounds, an 8-bit LFSR with a sparse feedback polynomial can be used. The LFSR is initialized with some state, and the cipher has to stop running the moment the LFSR arrives to the predetermined state. The key schedule of the KATAN cipher loads the 80-bit key into another LFSR (the least significant bit of the key is loaded to position 0 of the LFSR). In each round, positions 79 and 78 of the LFSR are generated as the round's subkey, and the LFSR is clocked twice [4].

2.4 Compact Implementation of AES

The compact AES core, as described by Moradi et al. [12] (AES_small_core), is byte-based. It uses only one S-box, which is implemented using composite field arithmetic in $GF(((2^2)^2)^2)$ [5]. Note that Moradi et al. suggest to use scan-flip-flops, while we follow the conventional tool flow that does not introduce a scan-flip-flop for the combination of a multiplexer and a flip-flop. We also did not make an effort in reducing the area of the control logic, which results in an area (reported in Table 1) that is larger than the area reported by Moradi et al.

3 Analysis Strategy

3.1 Architectural Decisions

In order to perform a fair comparison, we make the following architectural decisions:

- All inputs and outputs of the cipher are buffered through a flip-flop.
- We consider encryption-only architectures. This is justified by the fact that the most popular modes of operation do not need decryption [24].

3.2 Evaluation of Design Parameters

The area reports are generated after hierarchical synthesis in *Cadence Encounter RTL Compiler* using UMC 130 nm low-leakage Faraday technology library. The area numbers in the tables are given in terms of two-input NAND gate equivalents (GEs).

Each module is first synthesized for the best power. We used *Cadence Encounter RTL Compiler* in this step. The implementations have been synthesized in UMC 130 nm low-leakage Faraday technology library. The generated netlists are then used to simulate the actual module with 100 random keys together with 10 random plaintexts per key to get the best statistics. From these simulations, SAIF files are generated, which also contain the toggle counts. All simulations are performed using Modelsim. In the last step, the SAIF files are sent back to the synthesis tool together with the netlist from the initial synthesis to run power analysis.

4 Results

In this section, we list the generated design properties for the different architectures. Further, we detect anomalies based on the fact that we expect the dynamic power consumption to be larger for designs with a larger area. The anomalies represent architectures of which (some) gates contribute less to the static and/or dynamic power consumption than the gates of other architectures. This is mainly related to the number of transistors that are conducting (for the static power consumption) and the number of nodes that switch (for the dynamic power consumption). The energy per bit is calculated by dividing the total power by the clock frequency and then multiplying by the cycle count, followed by dividing by the block length.

A comparison of the AES architectures in Table 1 shows that AES_small_core consumes less area and power than the other 5 parallel cores, as expected. However, because of the large number of required cycles to perform the computation, the energy consumption is higher than the parallel architectures. The parallel architectures only differ in the way the S-box was implemented. The table shows that the architectural differences in the S-boxes cause significant differences in area, power and energy. The reason for some architectures to have a larger dynamic power consumption compared to others while they have a smaller area is probably caused by the fact that these architectures give rise to more internal glitches. The reason for some architectures to have a larger static power consumption compared to others while they have a smaller area is probably caused by the fact that parts of the architecture are not being used all the time during the computation.

KLEIN_parallel in Table 2 is a round-wise implementation which processes 1 round of KLEIN-64 in one clock cycle. However, KLEIN_serial in the same table is a byte serialized implementation of the same version of KLEIN. Since some of the resources are re-used in the serial implementation, the dynamic power is decreased by half. However, this has a negative effect on the number of rounds that need to be run and therefore we can see the serialized approach is not energy efficient. In the KATAN designs, the block size changes while the key size is the same. Therefore a slight change in area and also in static power consumption is observed in Table 2. But the dynamic power is quite low due to the simplistic round operations included in KATAN. When comparing LED and CLEFIA, both with block length 128, it is noticeable that the area ratio is much smaller than the power consumption ratio, in favor of LED. The reason is that LED is based on a fixed key and a simpler round computation compared to CLEFIA. The energy comparison also turns out in favor of LED. In the same way, the round function of PRESENT is much simpler in terms of logic depth compared to mCrypton, resulting in a similar trend. The energy comparison turns out in favor of mCrypton though. This is due to the fact that PRESENT needs more cycles. The reason that the unfolded version of PRINCE is only 13 times larger than the folded version but consumes 60 times more power is because of the fact that the unfolded version does not contain any registers. A comparison of the energy consumption also turns out in favor of the folded version.

Note that all the implementations of the ciphers and architectures listed in Tables 1 and 2 are re-implemented from the references.

Cipher Architecture	Block length	Encryption time (# cycles)	Freq. (KHz)	Area (GEs)	Static power (μW)	Dynamic power (μW)	Energy per bit (pJ/bit)	Energy (nJ)
AES_small_core	128	211	100	3685	6.25	11.31	289.47	37.05
AES_2_2_2_128	128	10	100	12405	24.46	210.15	183.29	23.46
AES_4_2_128	128	10	100	11453	21.37	135.26	122.37	15.66
AES_iso_2_2_2_128	128	10	100	15442	30.41	52.85	65.05	8.33
AES_iso_4_2_128	128	10	100	13052	25.19	37.06	48.63	6.23
AES_lut_128	128	10	100	19591	30.81	96.11	99.16	12.69

Table 1. Performance and energy numbers of various AES realizations all using 128-bit block lengths.

Cipher Architecture	Block/Key length	Encryption time (# cycles)	Freq. (KHz)	Area (GEs)	Static power (μW)	Dynamic power (μW)	Energy per bit (pJ/bit)	Energy (nJ)
CLEFIA	128/128	18	100	6941	13.24	37.09	70.78	9.06
KLEIN_parallel	64/64	12	100	2760	4.88	2.18	13.24	0.85
KLEIN_serial	64/64	98	100	1432	2.56	1.48	61.86	3.96
LED	64/128	48	100	3194	5.62	2.34	59.70	3.82
mCrypton	64/96	13	100	3197	5.80	2.50	16.86	1.08
PRESENT	64/80	31	100	2195	3.75	1.14	23.69	3.82
PRINCE_folded	64/128	12	100	2953	5.75	2.80	16.03	1.03
PRINCE_unfolded	64/128	1	100	8577	16.13	120.20	21.30	1.36
Katan_32	32/80	254	100	801	1.52	0.43	154.78	4.94
Katan_48	48/80	254	100	925	1.71	0.49	116.42	5.60
Katan_64	64/80	254	100	1048	1.94	0.56	99.22	6.34

Table 2. Performance and energy numbers of lightweight block ciphers implementations.

5 Conclusions and Future Work

In this paper, we evaluated the area, power consumption, and energy of 11 lightweight block cipher architectures (parallel, serial, and unfolded implementations) and 6 AES architectures (1 byte-based core and 5 parallel cores). We discussed the differences in dynamic power consumption in relation to the area. The results show that the parallel AES core with LUT-based S-boxes is the largest in terms of GEs, but consumes the least dynamic power of all parallel cores. For the other ciphers, the comparison between parallel and serialized versions depends on the algorithm, namely on the complexity of the round function.

It is evident from the results presented in this paper, dynamic power consumption plays an important role on the energy/power consumption of cryptographic chips. Although in this work, industrial tools are used to generate dynamic power consumption results, one can also investigate the HDL implementation of a cipher and estimate the dynamic power consumption through measuring the toggle activity. The basic idea is to measure the total number of bit toggles, i.e., bit flips, that happen during the encryption of a single block of a given algorithm. This metric can be in particular helpful when considering energy. The energy consumption in CMOS circuits is dominated by the dynamic power dissipation. This is directly proportional to the number of bit toggles and hence provides a good prediction for the energy consumption of a cipher. As future work, we will investigate the relation between the number of toggles from an HDL implementation and power reports from synthesis. We believe coming up with a high-level method to estimate power consumption of a cryptographic chip is very important and we will argue this in our future work as well.

Acknowledgments

This work was supported in part by Technology Foundation STW, The Netherlands as STW project SIDES and by the Research Council KU Leuven: TENSE (GOA/11/007), by iMinds, by the Flemish Government, FWO G.0550.12N and by the Hercules Foundation AKUL/11/19.

References

1. J. Guo, T. Peyrin, A. Poschmann, M. J. B. Robshaw. The LED Block Cipher. In B. Preneel, and T. Takagi (Eds.), *Proceedings of the 13rd International Workshop on Cryptographic Hardware and Embedded Systems (CHES '11), Lecture Notes in Computer Science*, volume 6917, pages 326–341. Springer-Verlag, 2011.
2. K. Shibutani, T. Isobe, H. Hiwatari, A. Mitsuda, T. Akishita, T. Shirai. Piccolo: An Ultra-Lightweight Blockcipher. In B. Preneel, and T. Takagi (Eds.), *Proceedings of the 13rd International Workshop on Cryptographic Hardware and Embedded Systems (CHES '11), Lecture Notes in Computer Science*, volume 6917, pages 342–357. Springer-Verlag, 2011.
3. A. Bogdanov, L. Knudsen, G. Leander, C. Paar, A. Poschmann, M. Robshaw, Y. Seurin, and C. Vikkelsoe. PRESENT: An Ultra-Lightweight Block Cipher. In P. Paillier, and I. Verbauwhede (Eds.), *Proceedings of the 9th International Workshop on Cryptographic Hardware and Embedded Systems (CHES '07), Lecture Notes in Computer Science*, volume 4727, pages 450–466. Springer-Verlag, 2007.

4. C. De Cannière, O. Dunkelman, and M. Knezevic. KATAN and KTANTAN - A Family of Small and Efficient Hardware-Oriented Block Ciphers. In C. Clavier, and K. Gaj (Eds.), *Proceedings of the 11th International Workshop on Cryptographic Hardware and Embedded Systems (CHES '09), Lecture Notes in Computer Science*, volume 5747, pages 272–288. Springer-Verlag, 2009.
5. D. Canright. A Very Compact S-box for AES. In J.R. Rao and B. Sunar (Eds.), *Proceedings of the 7th International Workshop on Cryptographic Hardware and Embedded Systems (CHES '05), Lecture Notes in Computer Science*, volume 3659, pages 441–455. Springer-Verlag, 2005.
6. M. Feldhofer, J. Wolkerstorfer, and V. Rijmen. AES Implementation on a Grain of Sand. In *IEE Proceedings of Information Security*, volume 152(1), pages 13–20. 2005.
7. D. Hein, J. Wolkerstorfer, and N. Felber. ECC is Ready for RFID - A Proof in Silicon. In R. Avanzi, L. Keliher, and F. Sica (Eds.), *Selected Areas in Cryptography, Lecture Notes in Computer Science*, volume 5381, pages 401–413. Springer-Verlag, 2009.
8. A. Hodjat, and I. Verbauwhede. The Energy Cost of Embedded Security for Wireless Sensor Networks. In G. Griffin, T. La Porta, and S. Phoha (Eds.), *Sensor Network Operations*, John Wiley & Sons, pages 510–522, 2006.
9. M. Knezevic. Efficient Hardware Implementations of Cryptographic primitives. Ph.D. thesis, Katholieke Universiteit Leuven, Belgium, 208 pages, 2011.
10. Y. K. Lee, K. Sakiyama, L. Batina, and I. Verbauwhede. Elliptic Curve Based Security Processor for RFID. In *IEEE Transactions on Computer*, volume 57(11), pages 1514–1527, November 2008.
11. G. de Meulenaer, F. Gosset, F.-X. Standaert, and O. Pereira. On the Energy Cost of Communications and Cryptography in Wireless Sensor Networks, (extended version). In *IEEE Int. Workshop on Security and Privacy in Wireless and Mobile Computing, Networking and Communications (SecPriWiMob '08)*, pages 580–585. IEEE, October 2008.
12. A. Moradi, A. Poschmann, S. Ling, C. Paar, H. Wang. Pushing the Limits: A Very Compact and a Threshold Implementation of AES. In K. Patterson (Ed.), *Advances in Cryptology (EUROCRYPT 2011), Lecture Notes in Computer Science*, volume 6632, pages 69–88, Springer-Verlag, 2011.
13. C. Rolfes, A. Poschmann, G. Leander, and C. Paar. Ultra-Lightweight Implementations for Smart Devices - Security for 1000 Gate Equivalents. In *proceedings of the 8th IFIP WG 8.8/11.2 International Conference on Smart Card Research and Advanced Applications (CARDIS '08), Lecture Notes in Computer Science*, volume 5189, pages 89–103, Springer-Verlag, 2008.
14. D. Singelée, S. Seys, L. Batina, and I. Verbauwhede. The Communication and Computation Cost of Wireless Security – Extended Abstract. In G. Tsudik, and N. Asokan (Eds.), *Proceedings of the 4th ACM Conference on Wireless Network Security (WiSec '11)*, pages 1–3. ACM, 2011.
15. S. Kerckhof, F. Durvaux, C. Hocquet, D. Bol, F.-X. Standaert. Towards Green Cryptography: a Comparison of Lightweight Ciphers from the Energy Viewpoint. in E. Prouff and P. Schaumont (Eds.), *Proceedings of CHES 2012*, Lecture Notes in Computer Science, vol 7428, pp 390-407, Leuven, Belgium, September 2012, Springer.
16. J. Borghoff, A. Canteaut, T. Güneysu, E. Bilge Kavun, M. Knezevic, L. R. Knudsen, G. Leander, V. Nikov, C. Paar, C. Rechberger, P. Rombouts, S. S. Thomsen, T. Yalçın. PRINCE - A Low-Latency Block Cipher for Pervasive Computing Applications - Extended Abstract. In Xiaoyun Wang, Kazue Sako (Eds.), *Advances in Cryptology (ASIACRYPT 2012)* Lecture Notes in Computer Science, volume 7658, pages 208–225, Springer-Verlag, 2012.
17. S. Tillich, M. Feldhofer, T. Popp, J. Großschädl: Area, Delay, and Power Characteristics of Standard-Cell Implementations of the AES S-box. Signal Processing Systems 50(2): 251-261 (2008)

18. Z. Gong, S. Nikova, Y. W. Law. KLEIN: A New Family of Lightweight Block Ciphers. In Ari Juels, Christof Paar (Eds.), *Proceedings of RFID. Security and Privacy - 7th International Workshop, RFIDSec 2011*, Lecture Notes in Computer Science, volume 7056, pp. 1– 18, Springer 2011.
19. P. Hämäläinen, T. Alho, M. Hännikäinen, T. D. Hämäläinen Design and Implementation of Low-area and Low-power AES Encryption Hardware Core *Proceedings of the 9th EUROMICRO Conference on Digital System Design (DSD'06)*, pp. 577-583, 2006.
20. T. Shirai, K. Shibutani, T. Akishita, S. Moriai, T. Iwata. The 128-bit Block-cipher CLEFIA (Extended Abstract). In *Proceedings of FSE 2007*. LNCS, vol. 4593, pp. 181-195. Springer, 2007.
21. D. Hong, J. Sung, S. Hong, J. Lim, S. Lee, B. Ko, C. Lee, D. Chang, J. Lee, K. Jeong, H. Kim, J. Kim, S. Chee. HIGHT: A New Block Cipher Suitable forLow-Resource Device. In L. Goubin, M. Matsui, (eds.), In *Proceedings of CHES 2006*. LNCS, vol. 4249, pp. 46-59. Springer, 2006.
22. C. Lim, T. Korkishko. mCrypton - A Lightweight Block Cipher for Security of Low-cost RFID Tags and Sensors. In J. Song, T. Kwon, M. Yung, M. (eds.) *Proceedings of WISA 2005*. LNCS, vol. 3786, pp. 243-258. Springer, 2006.
23. J. Guo, T. Peyrin, A. Poschmann, A. J. B. Robshaw. The LED Block Cipher. In *Proceedings of CHES 2011*. LNCS, pp. 326-341. Springer, 2011.
24. Morris Dworkin. NIST Recommendation for Block Cipher Modes of Operation, Methods and Techniques. NIST Special Publication 800-38A, 2001.