A review of lightweight block ciphers

Article *in* Journal of Cryptographic Engineering · April 2017

DOI: 10.1007/s13389-017-0160-y

CITATIONS
26

READS
1,502

4 authors:

George Hatzivasilis
Foundation for Research and Technology - Hellas
38 PUBLICATIONS **282** CITATIONS
SEE PROFILE

Konstantinos Fysarakis
Sphynx Technology Solutions AG
51 PUBLICATIONS **351** CITATIONS
SEE PROFILE

Ioannis Papaefstathiou
Technical University of Crete
174 PUBLICATIONS **1,209** CITATIONS
SEE PROFILE

Harry Manifavas
Qatar University
57 PUBLICATIONS **862** CITATIONS
SEE PROFILE

Noname manuscript No.
(will be inserted by the editor)

A Review of Lightweight Block Ciphers

George Hatzivasilis · Konstantinos Fysarakis · Ioannis Papaefstathiou · Charalampos Manifavas

Received: date / Accepted: date

Abstract Embedded Systems are deployed in various domains, including industrial installations, critical and nomadic environments, private spaces and public infrastructures. Their operation typically involves access, storage and communication of sensitive and/or critical information that requires protection, making the security of their resources and services an imperative design concern. The demand for applicable cryptographic components is therefore strong and growing. However, the limited resources of these devices, in conjunction with the ever-present need for smaller size and lower production costs, hinder the deployment of secure algorithms typically found in other environments and necessitate the adoption of lightweight alternatives. This paper provides a survey of lightweight cryptographic algorithms, presenting recent advances in the field and identifying opportunities for future research. More specifically, we examine lightweight implementations of symmetric-key block ciphers in hardware and software architectures. We evaluate 52 block ciphers and 360 implementations based on their security, performance and cost, classifying them with regard to their applicability to different types of embedded devices and referring to the most important cryptanalysis pertaining to these ciphers.

Keywords symmetric cryptography · lightweight cryptography · block ciphers · embedded systems security

G. Hatzivasilis, K. Fysarakis and I. Papaefstathiou
Department of Electronic & Computer Engineering, Technical University of Crete, Chania, Crete, Greece, Akrotiri Campus, 73100
Tel.: +30-28210-37218
Fax: +30-28210-37542
E-mail: gchatzivasilis@isc.tuc.gr, E-mail: kfysarakis@isc.tuc.gr, E-mail: ygp@mhl.tuc.gr

C. Manifavas
Department of Electrical Engineering & Computer Sciences, Rochester Institute of Technology Dubai, Silicon Oasis, Dubai, UAE, Techno Building, 341055
E-mail: cxmcad@rit.edu

1 Introduction

Pervasive computing is a growing trend for where devices with embedded microprocessors are used to enhance various aspects of our everyday lives (e.g. [43,44,57,58]). Such devices operate on limited resources and therefore, data processing, communication protocols and underlying technologies must be carefully chosen to meet strict operating requirements. Considering, however, that the information they handle is in many cases private or safety critical and must be appropriately protected from malicious attackers, appliance of secure cryptographic components becomes imperative. Lightweight cryptography (LWC) investigates the integration of cryptographic primitives and algorithms into constrained devices [55,59].

Between the two main categories of cryptographic algorithms, symmetric- and asymmetric-key, the former exhibit good performance and are widely used for confidentiality, integrity checks and authentication protocols. The tested robustness and known levels of security of well-known block ciphers, like AES [132], make them good candidates as long as they can be adjusted to meet the corresponding resource constraints. AES is the standard symmetric-key cipher used in cryptographic applications. Newer block ciphers designed specifically for this domain are gaining ground, introducing novelties and improving the overall efficiency.

In the following sections we present a survey of lightweight block ciphers for embedded systems as well as their implementations (in hardware and software). Similar surveys on LWC were first carried out in 2007. In [36] and [96], the authors evaluate hardware and software implementations for lightweight symmetric and asymmetric cryptography. In [118], the authors investigate the lightweight hardware and software solutions for Wireless Sensor Networks (WSNs), i.e. groups of highly-constrained hardware platforms. In [105], the authors report new trends for lightweight hardware block and stream ciphers. In [78], hardware implementations of block ciphers are examined while in [27] and [37] the authors implement and evaluate lightweight block ciphers on the same platform for fair comparison. Software implementations of lightweight block ciphers on three different platforms are presented in [35]. The latest survey on cryptanalytic attacks on lightweight block ciphers was carried out in [8].

This paper is a comprehensive survey in LWC and includes recently proposed ciphers. We further present the latest trends in hardware/software implementations of lightweight block ciphers and summarize the state-of-the-art design directions. The latest cryptanalysis results are mentioned and the vulnerable ciphers are reported. We analyze the features of block ciphers and identify the more suitable ciphers for different types of embedded devices. Based on the devices' capabilities, we categorize the implementations in three groups: ultra-lightweight, low-cost and lightweight. Ultra-lightweight implementations fit in the most constrained devices (in terms of computation capability, memory, power), like the standard 8051 microcontroller and the ATtiny45. Low-cost devices are affordable and perform a little better than ultra-lightweight ones,

like the ATmega128. Lightweight devices include the remaining devices that are reported in LWC.

Hardware-based algorithm implementations are categorized based on chip area and complexity. Ultra-lightweight implementations occupy up to 1000 logic gates, low-cost implementations occupy up to 2000 logic gates and lightweight implementations occupy up to 3000 logic gates. The software implementations are categorized based on the ROM and RAM requirements. Ultra-lightweight implementations require up to 4KB ROM and 256 bytes RAM, low-cost implementations require up to 4KB ROM and 8KB RAM and lightweight implementations require up to 32KB ROM and 8KB RAM. Finally we record the implementation characteristics of the latest proposals in hardware/software and create benchmarks based on metrics.

The remainder of this article is structured as follows. In section 2, we briefly introduce LWC and denote the evaluation metrics we applied for categorizing the different block cipher proposals. Section 3 presents the evolution of LWC in symmetric-key block ciphers. Section 4 proposes the more suitable implementations in the three proposed platform and section 5 concludes this work. Finally, appendix A summarizes the general features and cryptanalysis results of each cipher and presents the benchmarks for the hardware and software implementations.

2 LWC implementations

2.1 Brief overview

Embedded devices have inherent limitations in terms of processing power, memory, storage, connectivity and energy consumption [45, 95]. While ordinary cryptography focuses in providing high levels of security, lightweight cryptography also needs to consider the aforementioned restrictions, which can affect design and implementation choices.

Hardware LWC implementations try to achieve the required functionality with the minimum amount of hardware real-estate. These designs are suitable for ultra-constrained devices that perform specific tasks. The efficiency depends on a number of aspects analyzed below, like, design complexity, CMOS technology, power and energy consumption, and throughput.

The implementation's complexity is one of the most dominant factors that affect the design approach. It is determined by the logic gates that are required to implement a cipher. The relevant metric is called Gate Equivalent (GE). GE is a unit of measurement that specifies the manufacturing-technology-independent complexity of digital electronic circuits. In CMOS technologies, the silicon area of a NAND2 gate usually constitutes the technology-dependent unit area commonly referred to as gate equivalent. Except for expressing complexity, the number of the logic gates or the GE metric reflect to a portion of the chip area that is occupied by the hardware implementation. Thus, they are usually utilized as an intuitive way to express the chip area of a design. A

specification in GE for a certain circuit represents a complexity measure, for which the corresponding silicon area can be deduced for a dedicated manufacturing technology. The authors in [121] and [145] suggest that approximately 250-4000 logic gates, out of the total 1000-10000 logic gates commonly present on RFID tag spaces, can be used for security related tasks. For lightweight devices, up to 3000GE can be acceptable while for even smaller devices, like low-cost RFIDs and 4-bit microcontrollers, 2000GE and 1000GE are proposed respectively (Table 4).

CMOS technology is another factor that affects the implementation characteristics. Different technologies and standard cell libraries produce different results. For example, the same implementation of PRESENT that is presented in [117] produces 1075GE on $0.18\mu m$, 1169GE on $0.25\mu m$ and 1000GE on $0.35\mu m$ CMOS technology.

Software implementations on the other hand target less constrained devices. One of the main goals of software implementations is to keep memory and CPU needs as low as possible in order to minimize power consumption and the compatible devices' cost.

Software implementations are optimized for throughput and energy savings by requiring fewer cycles, since voltage and clock frequency are usually fixed for microcontrollers. These implementations are included in cryptographic libraries for embedded systems [47, 56].

Memory restrictions, however, are bound to negatively affect performance. As small memory elements are utilized, more cycles are needed to execute an operation. While in hardware implementations the 4-bit S-boxes are a popular option [7, 21, 39, 48, 51, 111, 124, 135, 148, 150], in software, the use of 8-bit S-boxes has also been proposed [7, 24, 28, 37, 99, 103]. The reduction in cycles count is considered significant despite the use of more space.

For active devices, like wireless sensor nodes that have their own power supply, energy consumption is a significant aspect. For passive devices, like contactless smart cards and RFID tags that do not have their own power supply and must adapt to the host device's constraints, power consumption is the main concern.

Power is an important as it is related to two main issues: the power consumption of a device and attacks related to power analysis. When frequency is fixed at a low value, like 100 kHz that is usually examined in LWC research (Table 4), the power consumption is directly connected to chip area [110]. A small area predisposes that the circuit will consume low power. Table 1 summarizes the GE and power characteristics of standard-cell libraries. Power deviates by a factor of two to three across different technologies. Table 2 summarizes the general frequency, RAM, ROM and power characteristics of different microcontroller platforms in the sensor network market.

Researchers aim to make the power usage profile of their implementations independent of the secret key, in order to withstand simple and differential power analysis [46, 54, 102]. As a result, the power consumption is increased. In applications where Differential Power Analysis (DPA) [80] is a threat, researchers attempt to counter DPA while trying to respect the power con-

straints. In less critical applications, power consumption is considered a priority while the countermeasures for DPA are less important.

Timing requirements are also imperative for deploying several special domain applications [82,150] and ISO/IEC standard protocols [39]. The ideal lightweight cipher should occupy a small circuit area, consume the least amount of resources and power, and perform as fast as possible.

Table 1 Characteristics of different CMOS technology nodes for commercial standard-cell libraries [138]

CMOS Technology Node	**Gate Density (kGEs/mm^2)**	**Power (nW/MHz/GE)**
$0.35\mu m$	6	18
$0.18\mu m$	125	15
$0.13\mu m$	206	10
$0.09\mu m$	403	7
$0.065\mu m$	800	5.68

Table 2 Characteristics of different microcontroller platforms [118]

Microcontroller platform	**Frequency (MHz)**	**RAM (KB)**	**ROM (KB)**	**Power (mA)**
8-bit	4 - 8	0.064 - 4	1.4 - 128	2.2 - 8
16-bit	4 - 8	2 - 10	48 - 60	1.5 - 2
32-bit	13 - 180	256 - 512	4000 - 32000	31 - 100

2.2 Fair comparison.

In order to fairly compare different ciphers several factors should uphold:

- The compared implementations need to provide the same security level.
- For hardware, the benchmarking set-up, including the CMOS library, the synthesis scripts, the synthesis tools need to be fixed and not to favour any of the competing ciphers
- Similarly, the same benchmarking machines need to be applied for software.
- There are plenty of implementation choices (e.g. serial and round-based) and the designs are optimized for specific evaluation metrics. This may lead to different architectures for different metrics.
- The reported results should ideally be reproducible by the wider community.

In order to fairly compare different ciphers, the aforementioned factors should all be considered, as it is difficult to implement all the proposals on the same platform.

On the same issue, the authors in [52] studied the impact of technology and standard cell libraries in the hardware implementation of hash functions. They conclude that accurate comparisons between different hardware implementations can be achieved when the same technology has been used, even if different standard cell libraries are used. On the other hand, significant inaccuracy could be noticed when using different technologies. Similar observations are mentioned for all other reported features, like performance and power consumption. The authors of said work also note that only latency is independent of technology, while power and area depend on technology. Throughput may depend on technology when the maximum throughput is needed, as the maximum frequency depends on the technology being used. Similar remarks can be deduced for software implementations.

A comparative study is conducted in Section 5. The ciphers with low security level are initially indicated to draw the reader's attention. In order to provide a fair comparison, the different implementations are initially grouped, based on the deployed technology in hardware and software. Also, the implementations with the worst features (e.g. high energy consumption or low throughput) are excluded. Among the efficient ciphers, the implementations of the same technology with the same key and data-path size are retrieved and compared.

Then, inter-technology evaluation is performed based on a subset of ciphers that are implemented in more than one technologies. For example, in hardware TWINE is only implemented in $0.09\mu m$ and RECTANGLE only in $0.13\mu m$. PRESENT is implemented in both platforms. Based on the aforementioned analysis, PRESENT is more efficient than TWINE in $0.09\mu m$ and PRESENT is less efficient than RECTANGLE in $0.13\mu m$. Thus, we derive that in general RECTANGLE is also more efficient than TWINE.

2.3 Evaluation metrics

In section 3, we compare several lightweight ciphers based on a number of metrics presented in this section. We summarize the details of 127 hardware and 233 software implementations. Specifically, we investigate security, cost and performance features. The best ciphers are those who provide the appropriate level of security and balance the tradeoffs between cost and performance. Some metrics are generic while others are tied to the hardware implementations (e.g. hardware technology and GE) and some to software implementations (e.g. RAM/ROM size). The metrics used for the comparison presented in this paper are the following:

– Security level: It is a logarithmic measure of the fastest known computational attack on an algorithm and is measured in bits. In most cases the level is identified by the key length in bits. The level of security cannot

exceed the key length but it can be smaller. Initially, a cipher's developers estimate its security level, which can be updated accordingly, based on reported attacks.
- Hardware technology: It is the CMOS technology used to implement the cipher with the corresponding occupied area being measured in μm. The technologies most commonly used in LWC research papers are $0.13\mu m$ and $0.18\mu m$ (Appendix A). The complexity and the area occupied by the hardware implementation are described by the Gate Equivalent (GE) metric and depend on the technology used. The GE metric is calculated by dividing the layout area of an implementation in μm^2 by the corresponding area of a NAND2 gate.
- Throughput: It stands for the Kb/s the cipher's encryption/decryption operation achieves at a specific frequency. Most of the research papers in LWC utilize frequencies of 100 KHz for hardware and 4 MHz for software. We record the throughput of each evaluated implementation. In cases where an implementation uses a different frequency, the reported throughput is projected at these two frequencies.
- Latency: It defines the number of clock cycles required to compute a single block's plaintext/ciphertext.
- Power and energy consumption: We evaluate the power of hardware implementations in μW. The power can be roughly estimated based on the GE and the hardware technology details referred to in Table 1. For software implementations, we consider an average power of $0.004\mu W$ and $0.00135\mu W$ for 8- and 16-bit microcontrollers in 4MHz frequency with 0.9V voltage respectively, based on Table 2. Energy consumption per bit for both hardware and software implementations is calculated by the formula:

$$Energy\,[\mu J] \;=\; (Latency\,[cycles/block] \times Power\,[\mu W])\,/\,block\,size\,[bits] \tag{1}$$

  Where *latency* is the number of clock cycles that are required to encrypt a block, *power* is the μW that are consumed by the hardware or software implementation and *block size* is the size of data in bits that each cipher can process in one encryption/decryption operation.
- RAM/ROM memory: The amount of RAM and ROM (in bytes) the algorithm requires. RAM bytes represent the resources required to store the intermediate state. ROM bytes represent the code size which is actually the size of the implementation.
- Efficiency: Indicates the trade-off between performance and implementation size. The higher the value, the better. For hardware implementations, the metric is calculated by the formula:

$$Hardware\,Efficiency \;=\; Throughput\,[Kbps]\,/\,Complexity\,[KGE] \tag{2}$$

  Where *throughput* is the Kb/s the cipher's encryption operation achieves at 100 KHz frequency and the *complexity* is the value of the chip area and complexity in KGE.

For software implementations, the metric is calculated by the formula:

$$Software\,Efficiency = Throughput\,[Kbps] \,/\, Code\,size[KB] \quad (3)$$

Where *throughput* is the Kb/s the cipher's encryption operation achieves at 4 MHz frequency and *code size* is the size of the executable code in KB.

3 Block ciphers

3.1 General information

There are five basic types of block ciphers based on their inner structure: **Substitution Permutation Networks (SPNs)**, **Feistel networks**, **Add-Rotate-XOR (ARX)**, **NLFSR-based** and **hybrid**. AES is the best known cipher that adopts the SPN structure, DES is the best known Feistel type cipher, while IDEA the best known ARX cipher, KeeLoq the best known NLFSR-based cipher and the best known hybrid ciphers are those of the Hummingbird family.

SPNs process plaintext through a series of sequential substitution and permutation boxes that transform the data and prepare them for the next round. SPN ciphers include AES [132], NOEKEON [32], ICEBERG [130], mCrypton [91], PRESENT [21], PUFFIN-2 [142], PRINTcipher [82], Klein [48], LED [51], EPCBC [150], PICARO [107], PRINCE [22], Zorro [46], RECTANGLE [155], I-PRESENT^{TM} [154] and PRIDE [4].

Feistel networks perform a diffusion function on half of the data of each block, resulting in a smaller round function. Additional logic is needed to apply the diffusion function to the untransformed state, such as a bitwise XOR operation, which requires 2.5-3 GE per bit. SPNs do not have this extra overhead and, as a result, serialized SPN ciphers are likely to achieve smaller datapaths. Feistel ciphers include DES [131] and its variants [87], GOST [111], TEA [146] and its variants [101,147], Camellia [9], SEA [129], CLEFIA [125], KASUMI [42], MIBS [66], TWIS [103], TWINE [135], LBlock [148], Piccolo [124], SIMON [15], ITUbee [74], FeW [83], Robin [49], Fantomas [49] and HISEC [5].

SPN and Feistel are the most widely-used types (Appendix A). As will be made clear in the following sections, many of the proposed lightweight Feistel-type ciphers suffer from security problems, in contrast to SPN-type ones. Still, Feistel networks offer both encryption and decryption with little cost, but in many tag-based applications decryption functionality is rarely required. An encryption-only SPN cipher still remains a strong competitor, if not the choice of preference, for the LWC field.

ARXs use addition, rotation and XORs with no S-boxes. The produce compact and fast implementations but their security properties are not well-studied, especially when compared to SPN and Feistel ciphers. ARX designs are IDEA [84], HIGHT [60], SPECK [15], LEA [61] and BEST-1 [67].

NLFSR-based ciphers utilize the building blocks of stream ciphers. They are mostly used for hardware implementations. The security of their inner components is based on stream cipher analysis. KeeLoq [63], KATAN and KTANTAN family [25] and Halka [34] have a stream cipher structure.

Hybrid ciphers combine the three aforementioned types to improve specific parameters, like throughput. Their analysis is determined by the selected cipher types. The Hummingbird family [38], [39] is a special case with hybrid structure of stream and block cipher. PRESENT-GRP [12] is another hybrid design that combines the PRESENT SPN with the bit permutation of a grouping permutation (GRP) structure.

Table 3 indicates the general characteristics of each examined cipher (key size, block size, rounds of operation and cipher's type) with references to the recorded cryptanalysis and attacks. Table 4 - Table 10 summarize the implementation characteristics of lightweight block ciphers in hardware and software respectively. In Table 4 - Table 7, the hardware implementations are categorized according to the supporting key size, the type of the block cipher, and the block size. We also indicate the parameters of latency, throughput, technology, GE, hardware efficiency, power, energy and cipher's type. In Table 8 - Table 10, the categorization of software is only based on the key and block size as the cipher's type column has been omitted. We further examine the parameters of ROM size, RAM size, latency, energy, throughput and software efficiency

The field 'Type' indicates the inner structure of the cipher. The following classes are used: SPN, Feistel, GFN (Generalized Feistel Network), ARX, NLFSR and hybrid.

The implementations presented are encryption-only, unless stated otherwise. Furthermore, the following notations are used:

- (S) for serialized implementations
- (D) for implementations that offer both encryption and decryption
- (H) when the key is hard-wired on the device

4 Chronicles

We categorize the lightweight block cipher proposals in three time periods based on the general LWC features and design goals. The ciphers of each period are grouped based on their type and are referred to in chronological order.

4.1 Period 1: early lightweight applications

In 80s and 90s cryptography on mainstream computers was investigated and the first cryptographic standards were established. Embedded systems usage was limited. Lightweight security was provided by compact implementations of mainstream ciphers and some early lightweight proposals. These solutions

target specific applications, like GSM security and remote keyless systems. The first attacks exposed the vulnerabilities of the lower security settings and established the cryptanalysis principles in this domain.

4.1.1 SPN ciphers.

AES is considered a landmark in traditional cryptography and, thus, cannot be ignored in the context of LWC. It uses 128-bit blocks with 128-, 192- and 256-bit keys through 10, 12 and 14 rounds respectively. Latest achievements in AES hardware implementations based on a serialized AES S-box [99] use 2400GE and 226 cycles per block. The S-box is based on Canright's research [26], who thoroughly investigated the hardware requirements for the AES S-box. Canright proposes a very compact S-box that is composed of smaller fields. The S-box that is used in the lightweight version is further minimized in area by the replacement of D flip-flops and MUX with scan flip-flops. The scan flip-flops are mainly utilized in the construction of the State and the Key array. Also the area requirements of the control logic are reduced by the replacement of the FSM with a LFSR. The authors also show that row-wise processing is more hardware-friendly than column-wise. The cipher remains safe with biclique cryptanalysis achieving slightly better results than exhaustive search [20].

In [37], a compact software implementation of AES is proposed. Registers are used to store the internal state and the mix column step while the key is stored in RAM. The S-box and the round constants are implemented as look-up tables. Also shift and XOR operations are used for the multiplications performed for mix column. The implementation requires 1659 bytes of ROM and needs 4557 cycles to encrypt a 128-bit block.

NOEKEON [32] uses 128-bit keys and blocks through 16 identical rounds. A related-key cryptanalysis was presented in [79] and as a result the cipher was rejected by the NESSIE project. Later, the designers of NOEKEON argued that the presented attacks were not practical and that the cipher remains safe [33]. The first hardware implementation [108] occupies 2880GE and is suitable for lightweight devices. In software [37], it requires 364 bytes of code for 21.7 Kbps throughput and is suitable for 32-bit processors. It is an early involutive cipher (uses the same datapath for encryption and decryption, allowing the same circuity to be reused for the inverse operation) whose design is explored by newer proposals.

ICEBERG [130] is a fast involutive cipher. It uses 128-bit keys with 64-bit blocks through 16 rounds. ICEBERG is optimized for reconfigurable hardware implementations as it allows changing the key at every clock cycle without any loss in performance and the deriving of the round keys on-the-fly. It produces very efficient combinations of encryption/decryption and requires 5800 gates for 400 Kbps of throughput [28]. Differential cryptanalysis on the 8-round version is the best known attack [134]. The overall design is investigated by newer involutive ciphers to provide low-cost encryption/decryption functionality.

4.1.2 Feistel ciphers.

DES is one of the first ciphers to be investigated for LWC. It uses 56-bit keys with 64-bit blocks through 16 rounds. The disadvantage of DES compared to AES is the smaller key size (i.e. 56-bits), yielding a lower security level (also broken under linear cryptanalysis [72]). On the other hand, smaller key sizes allow for smaller inner structures and low area footprint. The **DESX** variant [87] uses key whitening to increase the security level and render brute force attacks impossible; it uses 184-bit keys for the same block size and rounds. Hardware implementations of DES and DESX are presented in [87] and their cost is 2309GE and 1848GE respectively.

Older software implementations are presented in [36]. Newer, more compact software implementations, are presented in [37] and [115].

TEA (Tiny Encryption Algorithm) [146, 153] (2355GE) uses 128-bit keys with 64-bit blocks and 64 rounds. It is notable for its efficiency in power-energy-memory, its simplicity and ease of implementation. TEA needs 648 bytes of code [37], requires 7408 cycles of encryption and does not use S-boxes. The weak points of TEA, identified in [75] and [113], are the equivalent-keys attack and its bad performance as a hash function. **XTEA** (eXtended TEA or Block TEA) [73, 101] was proposed to overcome these weaknesses. Among others, XTEA operates on arbitrary size blocks and utilizes a more complex key scheduling procedure. A related-key rectangle attack on 36 rounds of XTEA is presented in [93]. At a later stage, XXTEA (Corrected Block TEA) [147] was proposed, but a chosen-plaintext attack based on differential analysis against the full-round cipher was later presented [151]. Even though these ciphers are designed for software implementations, hardware implementations are reported in [73] for XTEA with 3490GE, which well exceed the 3000GE limit.

Camellia [9] is developed by Nippon Telegraph and Telephone Corporation and Mitsubishi Electric Corporation and is approved by ISO/IEC, IETF, the projects NESSIE and CRYPTREC, and is adopted in Japan's new e-Government Recommended Ciphers List. It became popular as it achieves similar security level and processing capabilities with AES. It uses the same block and key sizes as AES through 18 or 24 rounds. In LWC, it is mainly studied for the fast software implementations as the hardware implementation [123] exceeds the 3000GE bound (at 6511GE). In software [24], Camellia requires 1262 bytes of code and 64 cycles for encryption. Cache timing attacks in software implementations were presented in [158].

4.1.3 ARX ciphers.

IDEA (International Data Encryption Algorithm) [84] uses 128-bit keys with 64-bit blocks through 8.5 rounds and all data operations are performed in 16-bit unsigned integers. To reduce the memory overhead, IDEA does not contain S- and P-boxes. It is based on XOR, addition and modular multiplication operations. It has been implemented in hardware for encryption in high-speed

networks [137]. However, these designs are not suitable for embedded devices. Yet, IDEA performs well in embedded software and is used in PGP v2.0. Its most compact implementation occupies 596 bytes of code for 94.8 Kbps throughput [100]. Similarly to AES, biclique attacks are slightly faster than exhaustive search [76].

4.1.4 NLFSR ciphers.

KeeLoq [63] is widely used in remote keyless entry systems. It was created by Gideon Kuhn in the 80's. The cipher is dedicated to hardware and utilizes a non-linear feedback shift register (NLFSR). It uses 64-bit keys with 32-bit blocks, a 32-bit initialization vector (IV) and process the data for 528 rounds. Practical key recovery based on a slide attack and a novel meet-in-the-middle attack are presented in [63].

4.2 Period 2: 1^{st} LWC generation

The first LWC generation covers the years 2005-2012. In this era, embedded systems are gaining ground and adequate general purpose security becomes imperative. The foundations of LWC are set and a high variety of ciphers are designed specifically for this domain. An arm race is held where the area reduction is one of the most targeted design goals. Mostly encryption-only implementations are evaluated, with speed and power optimizations also taking place. PRESENT is the new milestone as the first lightweight cipher that reached the bound of around 1000GE area. Speed and power optimized ciphers are also proposed. Cryptanalysis is evolved as new attacks are efficiently applied in lightweight primitives. The period ends by the formal representation of LWC and the establishment of the ISO/IEC standard for LWC that includes the block ciphers PRESENT and CLEFIA.

4.2.1 SPN ciphers.

mCrypton (miniature of Crypton) [91] is a compact edition of Crypton [90]. It uses smaller block size (64-bit) through 13 rounds and variable key sizes (64-, 96- and 128-bit) to comply with the new constraints in ubiquitous computing devices. It is designed for low-cost RFID tags and sensors and exhibits low-power and compact implementations in both hardware and software. A related key rectangle attack was reported in [106] for the 8-round mCrypton.

PRESENT [21] meets lightweight and ultra-lightweight requirements. It is one of the first ciphers implemented on ultra-constrained devices with almost 1000GE (encryption-only) [117]. PRESENT is a milestone in the evolution of lightweight block ciphers and is used along with AES as a benchmark for newer proposals. As with CLEFIA, PRESENT is also standardized in ISO/IEC 29192. It uses 80- and 128-bit keys with 64-bit blocks through 31 rounds. It has an SPN structure and needs 1030GE for 80-bit keys of both encryption. The

main feature is the replacement of the ordinary eight S-boxes with a carefully selected single S-box. PRESENT is the first cipher that applied this serialized architecture and its design is extremely hardware efficient, since it uses a fully wired diffusion layer without any algebraic unit.

PRESENT software implementations on different platforms have also been studied [110]. The newest software proposal [37] decreases the code size, retains the throughput and performs both encryption and decryption. The code size is 1000 bytes and the algorithm requires 11342/13599 cycles for encryption/decryption on an 8-bit microcontroller. The implementation stores the round key and the state in registers. Two 256-byte tables are used for the encryption and decryption S-boxes to permit parallel lookups, while the code for permutation is utilized in both operations.

On the downside, side-channel attacks [114,149] and a related-key attack [104] have been reported on the 17 round of PRESENT. Improved differential fault analysis was presented in [71], which recovered the key by inducing two or three 2-byte random faults. Biclique cryptanalysis on the full round cipher are slightly better than exhaustive search [69]. A truncated differential attack on the reduced 26-round cipher was presented in [19].

PUFFIN-2 [142] is based on its predecessor cipher **PUFFIN** (2303GE) [29] (the latter is broken under statistical saturation attacks [85]). It is designed to be implemented exclusively with a serialized architecture and supports the same key and block size as PRESENT through 34 rounds. PUFFIN-2 is faster and has smaller footprint (1083GE), while providing both encryption and decryption functionality; it is an involutive cipher using the same datapath for encryption and decryption. The cipher is resistant to related-key attacks, since at key scheduling the relevant permutation layers are not regularly distributed among rounds. Differential cryptanalysis on the full cipher is slightly better than exhaustive search [18].

Klein [48] uses 64-bit blocks with 64-, 80- and 96-bit keys through 12, 16, and 20 rounds respectively. It targets legacy sensor platforms. The main goal was to achieve high software-based performance while keeping a compact hardware footprint. As, in general, sensors are more powerful than RFID tags, software implementations are more practical. For its inner structure, several choices are made to balance the small area in hardware and software performance. Thus, bit-shifting operations, used by many hardware compact ciphers, are avoided. Byte-oriented matrix multiplication operations are opted for because of their software efficiency on 8-bit processors.

Practical chosen-plaintext key-recovery attacks have been successful for up to 8 rounds of Klein-64 [10]. They exploit the existence of differentials of high probability deriving from the combination of the 4-bit S-box and the byte-oriented MixColumns operation. An asymmetric biclique attack on the full version cipher exploits weaknesses of the diffusion layer and key schedule [3]. The worst case computations are $2^{62.84}$ with 2^{39} data complexity. A modified version of the attack, with higher data requirements, is slightly faster.

LED (Lightweight Encryption Device) [51] is designed to achieve small hardware footprint while keeping reasonable software performance. It uses 64-

, 80-, 96- and 128-bit keys with 64-bit blocks through 32 and 48 rounds. It is an AES-like cipher, and the authors apply some newer trends from the field of lightweight hash functions based on block ciphers. LED uses no key scheduling process, the PRESENT S-box, the row-wise processing as described for lightweight AES [99] and the mix column approach of the hash function PHOTON [50]. The absence of key scheduling is also proposed in CGEN [116]. This approach offers significant chip area reduction for hardware implementations but may give rise to serious security issues, like related key attacks [16]. The authors study and address these vulnerabilities. The research for the absence of key scheduling is the main contribution of the cipher, though, as reported, further independent cryptanalysis must be performed. However, the reduced round cipher is vulnerable to biclique cryptanalysis [69]. Differential fault analysis based on Super-Sbox techniques obtain significant improvements for fault attacks [156]. The search space for the 128-bit key exhaustive search is reduced on an average of $2^{21.96}$.

4.2.2 Domain specific SPNs.

Domain specific ciphers, like PRINTcipher [82] and EPCBC [150] are implemented to meet the cryptographic requirements of specific applications, such as Integrated Circuit (IC) printing and Electronic Product Code (EPC) encryption respectively. IC-printing is used for the production and personalization of circuits at very low cost. EPC is an industry standard by EPCglobal [41] and is considered as a replacement for barcodes using low-cost passive RFID. In its smallest form, it uses 96-bit keys.

PRINTcipher uses 80- and 160-bit keys with 48- and 96-bit blocks through 48 and 96 rounds respectively. It is developed for both domains, i.e. PRINTcipher-48 (402GE) is designed for IC-printing applications while PRINTcipher-96 (726GE) for EPC encryption. It uses 3-bit operations. As computer architectures do not use odd number of bits, a software implementation would use redundant resources. It is noted that PRINTcipher was released in order to investigate this application domain and is not ready for actual deployment yet. The authors are not concerned about related-key attacks as they are considered unrealistic for IC printing applications. Such attacks on the full round cipher were presented in [89].

EPCBC is a PRESENT-like cipher that uses 96-bit keys with 48- or 96-bit block size through 32 rounds. The main contribution is the adjustment of an improved PRESENT version with the 96-bit keys aiming to be used for EPC encryption. The authors took the security analysis of PRESENT into consideration. Thus, EPCBC uses an optimized key scheduling procedure that is more secure against related key differential attacks. The most lightweight version of the cipher costs 1008GE and, to our knowledge, it is the most suitable cipher for EPC encryption. Reduced-round versions are vulnerable to practical algebraic cryptanalysis [141].

Ciphers like AES and the original PRESENT support smaller or larger key sizes than 96-bit. Assuming that 96-bit keys are adequate, smaller keys do

not offer the required level of security while larger keys produce larger than necessary implementations due to redundant components. Other ciphers that support 96-bit keys are SEA (3758GE), mCrypton (2420GE), Klein (1528GE) and LED (1116GE). The older ciphers, i.e. SEA and mCrypton, produce larger implementations and are not suitable for EPC encryption, although mCrypton is the most efficient.

4.2.3 Feistel ciphers.

SEA (Scalable Encryption Algorithm) [129] is designed for scalable software implementations on constrained devices, being parameterized in text, key and processor size. It uses low-cost encryption routines that are suitable for processors with a limited instruction set. The design goals were to meet low memory requirements, small code size and a limited instruction set. Other features are the efficient combination of encryption-decryption, the ability to derive keys on-the-fly and the straightforward implementation in assembly code. Its most compact software implementation on 8-bit microcontrollers [37], requires 426 bytes of code and 41604 cycles for encryption. In [94], the authors propose a hardware implementation where they demonstrate a fully generic VHDL design to achieve the algorithm's scalability. The most lightweight version requires 3758GE.

CLEFIA [7,125] is another lightweight block cipher, known for its highly efficient hardware and software implementations. It was designed by SONY and is standardized in ISO/IEC 29192. CLEFIA uses 128-bit blocks with 128-, 192- and 256-bit keys through 18, 22, and 26 rounds respectively. The most compact implementation occupies 2488GE (encryption only) for a 128-bit key. It follows a serialized architecture without requiring any additional registers. Decryption can be implemented with an 116GE overhead. The most lightweight encryption/decryption version occupies 2604GE, which is 23% smaller than the corresponding AES-128 version. The authors apply clock gating techniques to reduce the number of multiplexers and increase the cycle counts. Furthermore, they adopt some older ideas utilized in compact AES implementations. For a compact matrix multiplier, operations are computed column by column. Scan flip-flops replace D flip-flops and MUX to further reduce area. Improbable differential attacks on reduced round versions achieve the best cryptanalysis results and are slightly better than exhaustive search [136].

KASUMI [42] is implemented for cryptography in GSM, UMTS and GPRS. It uses 128-bit keys with 64-bit blocks and processes the data in 8 rounds. A differential-based related-key attack was presented in [17].

Two newer DES and DESX variants, called **DESL** (DES Lightweight) and **DESXL** [87], are also presented for the same key and block sizes and rounds. DESL reduces the gate complexity by replacing the 8 original S-boxes with a single one, thus removing 7 S-boxes and a multiplexer. The single S-box is repeated 8 times and is designed to resist against differential, linear and the Davis-Murphy attacks. It achieves a size reduction of 20%. Also, it utilizes

serial hardware techniques to reduce gate complexity. DESXL is the combination of DESL and DESX and features an 18% size reduction compared to DESL. Hardware implementations of DESL and DESXL are presented in [87] and their cost is 2629GE and 2168GE respectively. Software implementations, are presented in [37,115]. For a compact implementation of DESXL they propose the use of a function which can compute all permutations and expansions depending on the calling parameters, while producing 6-bits outputs as direct input to S-boxes. The permutation and expansion tables are stored and processed from flash memory.

MIBS [66] supports 64- and 80-bit keys with 64-bit blocks through 32 rounds. It has a Feistel structure with an SPN round function, utilizing the S-box of mCrypton [91]. The permutation is operated in nibbles and the F-function operates on half a block. The key scheduling is based on PRESENT. The authors take the side-channel and related key attacks reported for PRESENT into consideration. They secure MIBS by using a round-dependent counter to mix the contents of the key register. However, many types of attacks [14] have come close to a complete compromise, namely linear attacks up to 18 rounds, first ciphertext-only attacks on 13 rounds, differential analysis up to 14 rounds and impossible-differential attacks up to 12 rounds. Although the full cipher is not broken, these attacks reduce its security level by more than 50%.

TWIS [103] is based on the CLEFIA cipher. A differential distinguisher with probability 1 for its full-round version was reported in [133].

Attempting to create a lightweight version of the Soviet **GOST** cipher, authors in [111] managed to produce a 651GE implementation. It uses 256-bit key, 64-bit block size and is a Feistel network with 32 rounds. The main contribution of this proposal is the adaptation of the PRESENT S-box that decreases the GE metric. The authors chose not to use simple wiring for permutation to reduce area, since that would result in weaker differential and linear properties. Recently, vulnerabilities were published and the cipher is theoretically broken by an improved three-subset meet-in-the-middle attack [64]. In [30], the authors further analyze differential attacks on GOST and present a single-key attack which is the fastest attack to our knowledge.

LBlock [148] uses 80-bit keys and 64-bit blocks through 32 rounds. It produces ultra-lightweight implementations in both hardware and software. In hardware, it needs 1320GE for 200 Kbps throughput. In software, assuming an 8-bit microcontroller, it takes 3955 clock cycles to encrypt a single block. For diffusion, the authors chose to use only half of the data in each round, and a simple rotation for the other half. This is applied efficiently on hardware and software. Permutation operations are implemented with no cost of GE in hardware. A 4-bit word-wise permutation is a bitwise permutation, as it can be implemented efficiently in software. Therefore, diffusions and permutations are efficient in both hardware and software. Key scheduling was designed in a stream cipher way.

Biclique cryptanalysis has been performed against the full round LBlock [143]. The complexity of the attack is slightly lower than exhaustive key search. The modification of the key scheduling operation is proposed as a counter-

measure to prevent the attack [143]. Improved differential fault attacks were presented in [70] that recover the key in a few seconds to one hour on a general PC. A round addition differential fault analysis reconstructs the key by utilizing one correct and two faulty ciphertexts [152].

TWINE [135] is a 64-bit block cipher with 80- and 128-bit keys through 36 rounds. It produces a compact hardware implementation of 1866GE. It is a GFN and implements a unified encryption and decryption functionality. The authors aim to balance performance on both hardware and software. For this reason, similar choices to LBlock have been made. Still, TWINE is an independent work and presents several concrete design advantages. One of their differences is that LBlock uses ten S-boxes while TWINE uses a single S-box. TWINE uses nibble permutation instead of bit permutation, while LBlock utilizes bit permutation for key scheduling purposes. The authors of TWINE published a saturation attack against a 22-round version of LBlock [135], lowering the security level originally claimed. While TWINE is not the most compact cipher, its performance is well balanced on both hardware and software. However, the simplified key-scheduling process can be exploited by meet-in-the-middle attacks [62].

Piccolo [124] is one of the most lightweight block ciphers. It is a variant of a Generalized Feistel Network (GFN) and supports 64-bit blocks with 80- and 128-bit keys through 25 and 31 rounds respectively. It features very compact implementations which achieve low energy consumption. The 80-bit key version is the most lightweight, requiring 432GE. Piccolo needs an additional 60GE for the decryption functionality. As a result, Piccolo remains more compact even when compared to encryption-only ciphers.

The key scheduling process is implemented by multiplexers without flip-flops (like GOST and KTANTAN) that require a large area. These values (key components) are stored in the key inputs segment, which is not hard-wired, and no registers are used for storing keys. As with other newer ciphers, Piccolo uses scan flip-flops for the data state. Also, the cipher uses AND-NOR and OR-NAND gates, which require 2GE, to perform the same functionality as XOR and XNOR respectively, which require 2.25GE. Moreover, it adopts a half-word based round permutation.

In software [27], it requires 2434 bytes of code and consumes 79 bytes of RAM. It has poor performance and achieves 7.8 Kbps of throughput.

Biclique cryptanalysis has been achieved on full round versions of Piccolo-80, [69,128]. For Piccolo-128, they perform slightly lower than exhaustive key search. Impossible differential cryptanalysis is also performed on reduced round versions [11].

4.2.4 ARX ciphers.

HIGHT [60] uses a compact round function, without the use of S-boxes, and all operations are simple computations. It uses 128-bit keys with 64-bit blocks through 32 rounds. The most compact hardware implementation

requires 2608GE for 188Kbps throughput [92]. An impossible differential attack on 26-round HIGHT was presented in [104], a related-key attack on full-round HIGHT was presented in ICISC2010 [81], biclique cryptanalysis has been achieved on full round version [128], and zero-correlation attacks on the 26- and 27-round cipher were presented in [144].

4.2.5 NLFSR ciphers.

The **KATAN** and **KTANTAN** [25] cipher family supports 80-bit keys and 32-, 48- and 64-bit blocks through 254 rounds. The ciphers follow the design of KeeLoq but they apply an LFSR (instead of NLFSR) and process the data for less rounds (254 rounds). They use a very simple key scheduling mechanism. KATAN achieves a small footprint of 802GE. KTANTAN authors propose to hard-wire the key on the device to further reduce the GE, due to the absence of key generation operations. KTANTAN is only suitable for devices where the key is initialized once and does not change. The most compact version of KTANTAN achieves 462GE. The authors propose that the version of KTANTAN-48 (588GE) is more suitable for RFID devices. The basic cipher resembles the structure of the stream cipher Trivium [98].

Their software implementations are not efficient since they use bit manipulations extensively. This fact is noted in [37], where a compact software implementation of KATAN is presented. The authors mainly try to decrease the code size. Although the proposal achieves a quite low code size of 338 bytes, it produces low throughput and consumes too much energy. The encryption takes 72963 cycles and the decryption 88525 cycles. Multidimensional meet-in-the-middle attacks on reduced round KATAN that are faster than exhaustive search were presented in [159]. Several related-key attacks which recover the full 80-bit key of KTANTAN with a probability of one are presented in [2].

4.2.6 Hybrid ciphers.

Hummingbird (HB) [38] is another promising ultra-lightweight cipher, which introduces a hybrid block and stream cipher structure, with 256-bit key and 16-bit block size through 20 rounds. It features better performance than PRESENT on 4-bit microcontrollers. The first version was found vulnerable to a number of attacks [119].

Its successor, **Hummingbird-2** (HB-2) [39], optionally produces a message authentication code (MAC) for each message processed and is developed with both lightweight software and hardware implementation constraints in mind. It can actually be considered a combination of a cipher and a protocol; its design fulfills the requirements of ISO 18000-6C protocol. HB-2 uses 128-bit secret keys and 64-bit Initialization Vectors (IVs) and, as its predecessor, it has been targeted for low end microcontrollers and hardware implementations in resource-constrained devices such as RFID tags and wireless sensors. HB-2 is broken under a related key attack [120].

4.3 Period 3: 2^{nd} LWC generation

This is the current period of LWC's lifetime. Pervasive computing is here and general purpose constrained devices are utilized in everyday activities. The competition for area optimization continues but in a less binding manner. The 1^{st} generation of ciphers achieved a high degree of reduction, where almost 80-90% of the implementation is occupied by memory elements (e.g. the cases of PRESENT and KATAN). New challenges are now considered to meet the real application requirements. Over and above the speed-optimization, low latency is now the objective. Decryption functionality becomes essential and involutive designs gain ground. Cryptanalysis reveals that many lightweight ciphers are vulnerable to side-channel attacks. Masking techniques and ciphers that are easy to mask are proposed. Strategies for designing secure and efficient ciphers are applied (e.g. wide-trail), along with further analysis on the linear and non-linear layers of the inner structures.

4.3.1 SPN ciphers.

The authors in [99] try to apply countermeasures against first-order side-channel attacks on AES. They implement the features proposed in [102] and increase the level of resistance. Their implementation occupies 11031GE ($0.33\mu m$). A newer implementation [31] requires 6340GE but it is deployed in larger CMOS technology ($0.45\mu m$). **PICARO** [107] is a new cipher which is specifically designed to counter such attacks. The authors implement a cipher that fits the masking constraints well, instead of integrating masking schemes to existing ciphers, like the AES proposal. They consider 4 different masking levels. PICARO's masked hardware implementation is faster than the corresponding AES. **Zorro** [46] is a newer proposal that is based on AES and provides efficient masking. It is suitable for embedded systems and it is implemented in 8-bit microcontrollers. Zorro is more efficient than PICARO as it requires less cycles to operate. However, practical invariant subspace attacks are presented in [86].

PRINCE [22] accomplishes low-latency in hardware. It is a notable proposal that applies the wide-trail strategy and uses 128-bit keys with 64-bit blocks through 12 rounds. Its lightweight implementation [13] requires 2953GE for 533.3 Kbps of throughput and has low energy consumption. Efficient software implementations are also presented [4]. Cryptanalysis in [68] and [127] reduced the security level, as attacks were performed on the reduced 6-round and the full 12-round cipher. A key recovery attack on the 7-round version based on truncated differential cryptanalysis is presented in [157].

RECTANGLE [155] is a recent lightweight SPN proposal. It uses 64-bit blocks with 80- and 128-bit keys through 24, 28 and 32 rounds respectively. The substitution is performed by 16 4×4 S-boxes in parallel setting and the permutation is executed in 3 rotations. Its authors propose a new type of S-box along with an asymmetric design of the permutation layer. These new designed criteria are motivated by the cryptanalysis of PRESENT. RECTAN-

GLE adapts bit-slice techniques to LWC and allows lightweight hardware and fast software implementations. In hardware, its most lightweight version with 80-bit keys requires 1467GE for 246 Kbps throughput and consumes the lowest energy/bit. In software, it requires 5.38 cycles/byte for encryption of 1000 byte messages using Intel 128-bit SSE instructions (such implementations are outside the LWC scope).

I-PRESENT^{TM} [154] is an involutive version of PRESENT, using the same key and block sizes through 30 rounds. The S-box layer uses two additional 4×4 S-boxes 16 times. The involutive part is inspired by PRINCE. A ciphertext block is produced after a 15-round function, followed by a 15-round involutive function (30 rounds in total the original cipher takes 31 rounds). The S-box from NOEKEON is used as the S-box of the involutive function. The key schedule procedure generates 30 64-bit round subkeys. Decryption is identical to encryption, except the fact that the round subkeys are inputted in the reverse order. The most compact hardware implementation requires about 2769GE, providing both encryption and decryption. The relevant encryption-only implementation of PRESENT requires 1570GE.

PRIDE [4] is a software cipher that uses 128-bit keys with 64-bit blocks through 31 rounds. The wide-trail design strategy is adopted and the linear properties of the S-box are further studied. A bit-sliced implementation is presented, with an extremely efficient linear layer. The implementation outperforms many other proposals in 8-bit microcontrollers and exhibits low latency and energy consumption.

4.3.2 Feistel ciphers.

SIMON [15] was designed by the NSA. A performance evaluation was presented in [15]. It performs well in software and hardware. It supports several sizes for key (64, 72, 96, 128, 144, 192, 256) and block (32, 48, 64, 96, 128) and round numbers (32, 36, 42, 44, 52, 54, 68, 69, 72). Independent cryptanalysis efforts on SIMON [6] present a series of observations on the cipher's construction and attacks on reduced round versions. Differential fault attacks on SIMON are presented in [139]. Cube and dynamic cube attacks on SIMON, with 64-bit key and 32-bit block, recover the full key in a practical time complexity [112].

ITUbee [74] is a newly proposed cipher, designed for lightweight software and suitable for 8-bit software based platforms. It is based on a Feistel structure and has no key schedule, featuring 80-bit keys with 80-bit blocks. Round-dependent constants are used, instead of strong key scheduling. In its most compact form [74], it requires 586 bytes of code and 2937 cycles for encryption. Self-similarity cryptanalysis is performed in [126] that exploits the round constants and builds relations between them. The reduced round cipher is distinguishable from an ideal random permutation. A deterministic related-key differential distinguisher is presented for the 8-round version in the single-key model, reducing the security by one bit.

FeW [83] is a software oriented design for LWC. It uses 80- and 128-bit keys with 64-bit blocks through 32 rounds. The S-box of Humminbird-2 is utilized in encryption, decryption and key schedule. The key expansion process is similar to PRESENT. FeW combines a Feistel and a generalized Feistel structure to enhance security against basic cryptographic attacks. It is reported as safe against present day cryptanalytic attacks [83].

Robin and **Fantomas** [49] are LS-designs (i.e. a combination of look-up table-based L-boxes and bitslice S-boxes). Both ciphers use 128-bit keys and blocks and are based on the Feistel-type block cipher MISTY [97] (ancestor of KASUMI). They integrate LS-design techniques to provide efficient masking and protection against side-channel attacks. The ciphers use the same constants and target single-key security, excluding related or chosen key attacks. They target efficient and secure software implementations on 8-bit microcontrollers, but are considered to perform well in hardware as well.

Robin is an involutive instance of LS-design, with 16 rounds that utilize the MISTY Class13 S-box [140]. The S-box and L-box are involutive, allowing the same circuity to be reused for the inverse operation (i.e. encryption and decryption). It aims to have similar security margins and performance to NOEKEON. These two bitslice ciphers optimized for Boolean masking are more efficient than AES, Zorro and PICARO. However, practical invariant subspace attacks are presented in [86], similarly with Zorro. A careful tweak of the cipher is proposed to prevent the attacks in case of Robin and the LS-design security is not questioned.

Fantomas is a non-involutive LS-design with 12 rounds that utilizes the MISTY 3/5-bit S-boxes [97]. It illustrates the impact of choosing involutive components with respect to inherent efficiency limits imposed by the LS-designs. Analysis indicates that involutive LS-designs require about 4/3 rounds of non-involutive LS-designs to achieve a similar level of security. Fantomas is safe against the aforementioned attacks on Robin and is also faster.

HISEC [5] is a GFN and exhibits similar characteristics as PRESENT with a different bit-permutation procedure. It uses 80-bit keys with 64-bit blocks and processes the data for 15 rounds. HISEC provides protection against the differential, integral and boomerang attacks that have been presented for other ciphers, like PRESENT, LBlock, Klein and TWINE. The hardware requirements are estimated to 1695GE. No performance details are provided.

4.3.3 ARX ciphers.

SPECK [15] is designed by the NSA along with SIMON. As is the case with SIMON, SPECK also performs well in software and hardware. It uses the same key and block sizes through 22, 23, 26, 27, 28, 29, 32, 33, and 34 rounds. SIMON is better in hardware and SPECK is better in software. Independent cryptanalysis efforts on SPECK [1] present a series of observations on the cipher's construction and attacks on reduced versions of the cipher. Differential fault attacks are presented in [139].

BEST-1 (Better Encryption Security Technique-1) [67] is designed for ultra-lightweight implementations with low cost and power on WSN and RFID. It uses 128-bit keys with 64-bit block through 12 rounds and fits in 8-bit processors. It is an ARX proposal and the main operations are mod2^8 addition and subtraction, bitwise shift and XOR. Decryption is performed at little cost as a sequence of steps after inverting the encryption process. In hardware, BEST-1 requires around 2200GE and achieves 265.7Mbps at 80MHz frequency.

LEA (Lightweight block Encryption Algorithm) [61] is a software-oriented ARX designed by the Electronics and Telecommunication Research Institute of Korea. It targets fast encryption on common processors and all operations are 32-bit wide. It uses 128-bit blocks with 128-, 192- and 256-bit keys through 24, 28, and 32 rounds respectively. LEA has small code size and consumes low power. On ARM platforms, it requires 590 bytes of ROM and 32 bytes of RAM for 326.94 cycles per byte speed. The cipher exhibits high hardware demand with the most compact implementation requiring 3826GE for 76.19 Mbps throughput [88]. The security analysis is theoretical. Power analysis on hardware is presented in [77]. The 128-bit key is retrieved by attacking the first round. Implementing LEA with countermeasures becomes essential.

4.3.4 NLFSR ciphers.

Halka [34] is a recently proposed cipher. It is designed for lightweight hardware but also remains software friendly. It uses 80-bit keys with 64-bit blocks through 24 rounds. The main feature is the implementation of a novel multiplicative inverse for 8-bit S-boxes using LFSR, which requires 138GE. The relevant feature of other existing standards demand more gates, like the Canright's AES S-box [26] that occupies 253GE. Thus, the total gate count of the Halka should be low. The use of 8- instead of 4-bit S-boxes is considered more secure in this design. Then, eight 8x256 bytes S-boxes are utilized to achieve super-fast software performance. The key schedule is similar to PRESENT but it is materialized with 8-bit S-box. In hardware, Halka is estimated to require 7% less GE than PRESENT but is considered more secure [34]. In software, it could be three times faster than PRESENT [34].

4.3.5 Hybrid ciphers.

PRESENT-GRP [12] is a hybrid cipher that combines the SPN of PRESENT with a GRP (group operation) structure for bit permutation. It uses 128-bit keys with 64-bit blocks through 31 rounds. The security properties of GRP are well-studied and offer higher confusion properties than bit permutation. The P-box of the original PRESENT is replaced by GRP. The hybrid cipher with the PRESENT S-box and GRP enhances security and results in lower memory and area consumption. The hardware design requires 2125GE, which is a little larger than the corresponding PRESENT (1884GE). In software, the implementation size is slightly larger than PRESENT for the same amount of RAM.

5 Results and discussion

5.1 Security

Table 3 indicates the features of each cipher and the best publicly known cryptanalysis results. Security issues should be carefully considered before using a cipher. For example KeeLoq, GOST, TEA, XTEA, mCrypton, HIGHT, KASUMI, PUFFIN, Hummingbird, MIBS, KTANTAN, TWIS, PRINTcipher, Humminbird-2, LBlock, Zorro and LEA all suffer from security vulnerabilities and should be avoided. DES 56-bit key is not adequate for nowadays applications.

Newer 2^{nd} generation proposals, namely PICARO, SPECK, ITUbee, RECTANGLE, FeW, Halka, BEST-1, Robin, Fantomas, HISEC, I-PRESENT^{TM}, PRESENT-GRP and PRIDE have to be further analyzed for vulnerabilities and their claimed level of security. AES, Camellia, PRESENT, CLEFIA, and DES variants are the most studied solutions and as a consequence the most acceptable ciphers. PRESENT and CLEFIA are the standardized block ciphers for lightweight cryptography.

5.2 Comparison

We examine 127 hardware and 233 software implementations. We evaluate all implementations based on the metrics and the fair comparison approach detailed in Section 2. In hardware, 0.09, 0.13, 0.18 and 0.35μm technologies are used. The software implementations are deployed in 8-, 16- and 32-bit microcontrollers.

5.2.1 Hardware implementations.

Regarding hardware implementations for constrained devices, Figure 1 to Figure 4 illustrate the best hardware implementations according to individual metrics. Implementations are summarized per technology in terms of GE, energy, power and hardware efficiency respectively. Implementations on 0.09μm technology are coloured with purple, on 0.13μm technology with deep blue, on 0.18μm technology with blue, and on 0.35μm technology with green. The standardized ciphers that are utilized as comparison units (AES, PRESENT and CLEFIA) have a yellow colour and implementations with excess demands in each metric are outlined with red colour. All hardware implementations are detailed in Table 4 - Table 7.

All proposals for the ciphers IDEA, Camellia, ICEBERG, KASUMI and LEA exceed the boundary of 3000GE and are, therefore, not considered efficient. NOEKEON, PUFFIN-2, LED and EPCBC consume excessive energy per bit and produce low hardware efficiency. KTANTAN, HIGHT, HB-2, TEA, Klein, SEA, KATAN, AES, DES, DESX, EPCBC, LED, PUFFIN-2 and NOEKEON consume high energy per bit. DESX, DES, AES, KATAN,

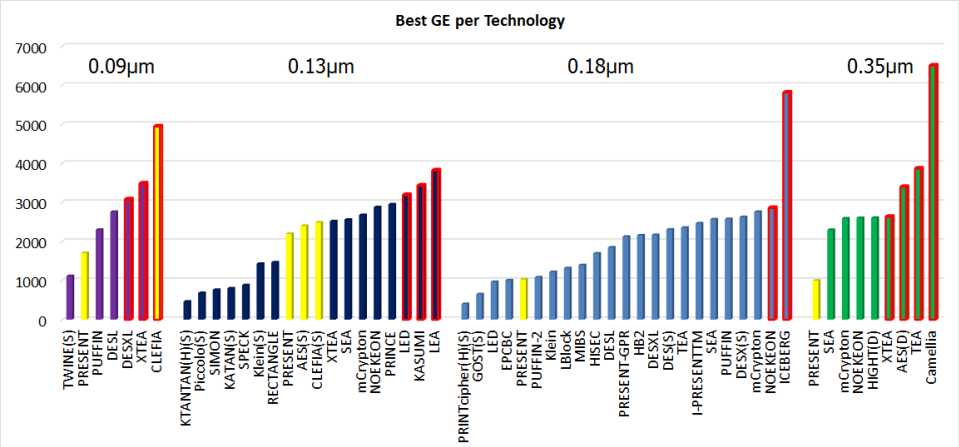

Fig. 1 Best hardware implementations in terms of GE.

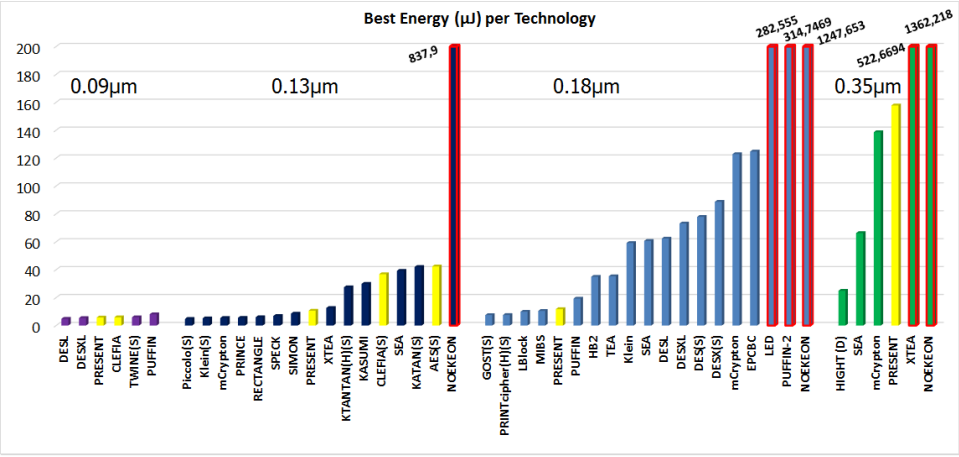

Fig. 2 Best hardware implementations in terms of energy consumption (axis y is limited at 200μJ).

KTANTAN, EPCBC, PUFFIN-2 and NOEKEON produce high latency (144 3720 cycles per block). NOEKEON, PUFFIN-2 and EPCBC have also low throughput. DES, HB-2, I-PRESENT^{TM}, DESX and HIGHT have higher power requirements than other ciphers.

KTANTAN, PRINTcipher, Piccolo, SIMON, TWINE, KATAN, SPECK and GOST produce compact implementations with the lowest power requirements (less than 1μW). Klein, LED, RECTANGLE, EPCBC, PRESENT, PUFFIN, PUFFIN-2, DESL, LBlock, MIBS, AES, CLEFIA, XTEA, HISEC and SEA consume also low power (around 2.5μW). Piccolo, DESL, Klein, DESXL, mCrypton, PRINCE, PRESENT, CLEFIA, TWINE, RECTANGLE,

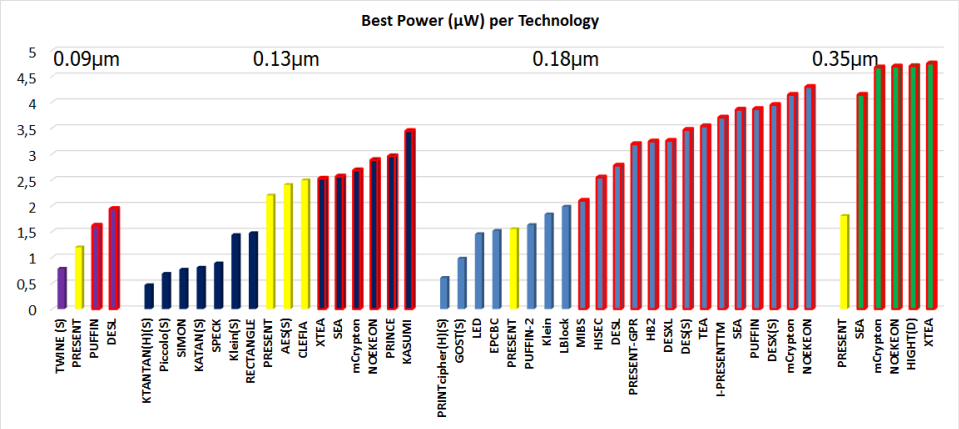

Fig. 3 Best hardware implementations in terms of power requirements.

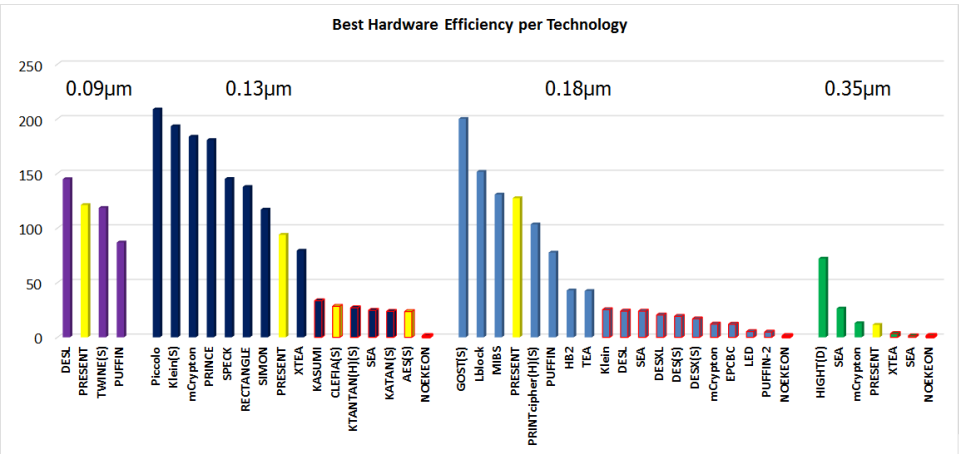

Fig. 4 Best hardware implementations in terms of hardware efficiency (Hardware Efficiency = Throughput [Kbps]/Complexity [KGE]).

SPECK, GOST, PRINTcipher, PUFFIN, SIMON, LBlock and MIBS consume low energy per bit (less than 10 μJ per bit).

Figure 5 illustrates the best overall hardware implementations. Based on the criteria adopted in this work, Piccolo evaluates best compared to all the other proposals for many key sizes. PRESENT and TWINE perform similarly and achieve a good overall status. TWINE, as a Feistel network, can offer the decryption functionality at low cost. RECTANGLE, PRINCE, SPECK and mCrypton exhibit low-latency and efficient implementations. SEA also offers low latency but it is quite slower. For lower security, DESL performs well.

Low-cost RFID devices require about 2000GE and 80-bit keys. From the ciphers that support 80-bit keys, LED, PUFFIN-2, KATAN, Klein, EPCBC and LBlock produce low hardware efficiency or consume a lot of energy per

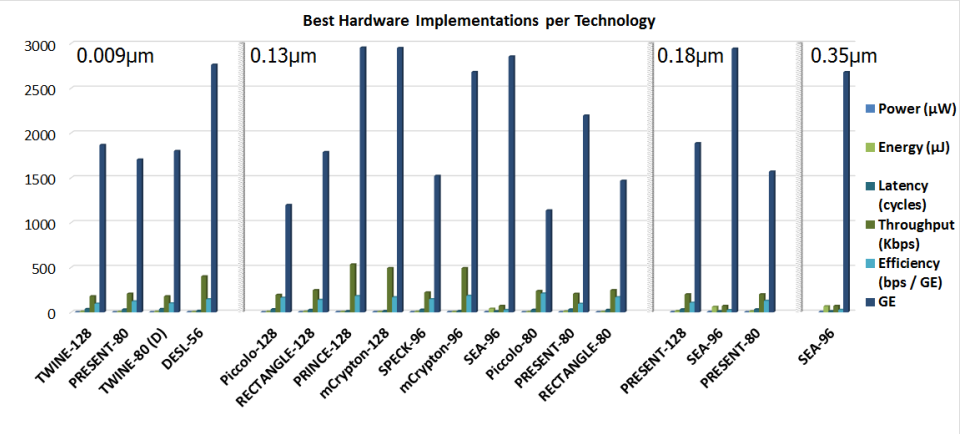

Fig. 5 Best overall hardware implementations per technology.

bit. Klein typically targets wireless sensors which are more powerful devices and PUFFIN-2 requires more than 2000GE. Piccolo achieves the best overall status compared to all the other ciphers. It occupies a small chip area even when both encryption and decryption are implemented. Piccolo has higher throughput and hardware efficiency, while it consumes the least energy. SIMON, SPECK and RECTANGLE also achieve a good overall status and consume low energy but are newly proposed ciphers. GOST and PRINTcipher and perform well in this category with higher key sizes. KTANTAN, MIBS, TWINE and PRESENT perform reasonably on such devices.

Suitable implementations for ultra-constrained devices require less than 1000GE and 64-bit keys. Klein, LED and MIBS provide exactly 64-bit keys. Among these three ciphers, only LED can be implemented within the GE limit but with high energy cost. The most compact ciphers that require around 1000GE and provide key sizes between 80- and 256-bits are the GOST, PRINTcipher, KTANTAN, Piccolo, SIMON, KATAN, SPECK, PRESENT and LED. GOST, PRINTcipher and KTANTAN perform the best.

Some serial implementations have achieved very small chip areas. The ciphers PRINTcipher, GOST, Piccolo, KTANTAN and SIMON can be implemented with less than 800 GEs.

Among the novelties and well-known practices proposed to enhance hardware implementations are the use of a scan flip-flop instead of a D flip-flop and a 2-to-1 MUX, AND-NOR gates instead of XOR gates and OR-NAND gates instead of XNOR gates. Furthermore, the storage of memory elements in registers instead of flip-flops consumes less energy and occupies less chip area. Bit-slice techniques are also utilized to reduce the area and design complexity requirements. Key scheduling is simplified to enhance latency and throughput. Involutive designs are suggested for low cost encryption/decryption functionality.

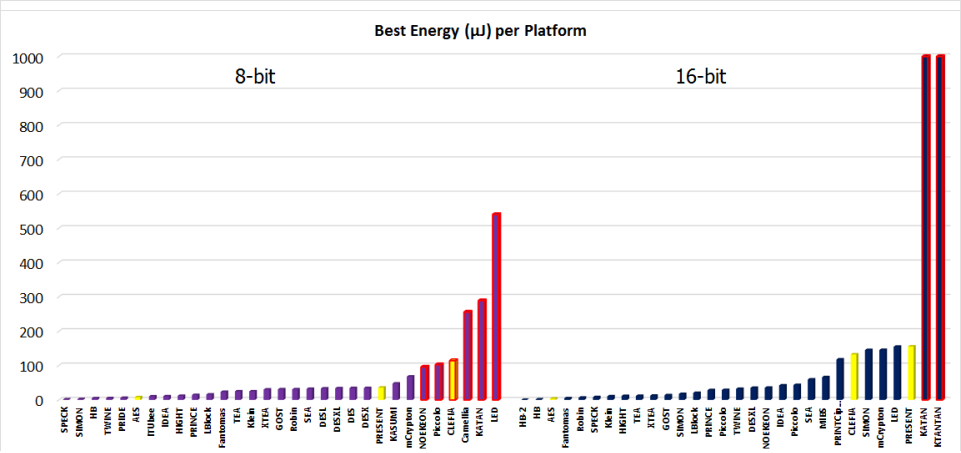

Fig. 6 Best software implementations in terms of energy consumption.

5.2.2 Software implementations.

For software implementations, Figure 6 to Figure 8 illustrate the best software implementations according to individual metrics. The ciphers AES, Camellia, IDEA, NOEKEON, KASUMI, GOST, HIGHT, PRESENT, PRESENT-GRP, CLEFIA, HB, HB-2, Piccolo, TEA, XTEA, SEA, mCrypton, MIBS, TWINE, LED, Klein, KATAN, KTANTAN, ITUbee, SIMON, SPECK, PRINTcipher, LBlock, PRINCE, PRIDE, Zorro, Robin, Fantomas, LEA and DES and its variants are examined.

KTANTAN, LED, KATAN and Camellia consume high energy. KTANTAN, KATAN, PRINTcipher, LED and Piccolo are too slow. The standardized PRESENT and CLEFIA has also low throughput as the lightweight proposals for mCrypton, PRINCE, KASUMI, MIBS, SEA, DESX, DES, DESXL and DESL. KASUMI, mCrypton, TWINE, NOEKEON, Piccolo, Camellia, IDEA, MIBS, PRINTcipher, CLEFIA, LED, KATAN and KTANTAN exhibit high latency in software (more than 10000 cycles). KTANTAN, KATAN, PRINTcipher, GOST, LED, MIBS, Camellia, DESX and DES achieve the worst overall performance based on the software efficiency metric in all three microcontroller technologies.

A speed optimized implementation of HB exceeds the boundary of 8KB RAM for both low-cost and lightweight devices and is not considered efficient for LWC. For ultra-constrained devices only, KTANTAN, PRESENT-GRP and Zorro exceed the boundary of 256 byte RAM and can not be applied. For ultra-constrained and low-cost devices, some of the implementations for GOST, PRINTcipher, KTANTAN, PRINCE, DES and DESX exceed the boundary of 4KB ROM with HB-2, DESL, DESXL, HB, LBlock and KATAN coming close.

The most compact implementations (in terms of code size) are reported for SPECK, SIMON, PRIDE, SEA, KATAN, NOEKEON, AES, ITUbee, HIGHT

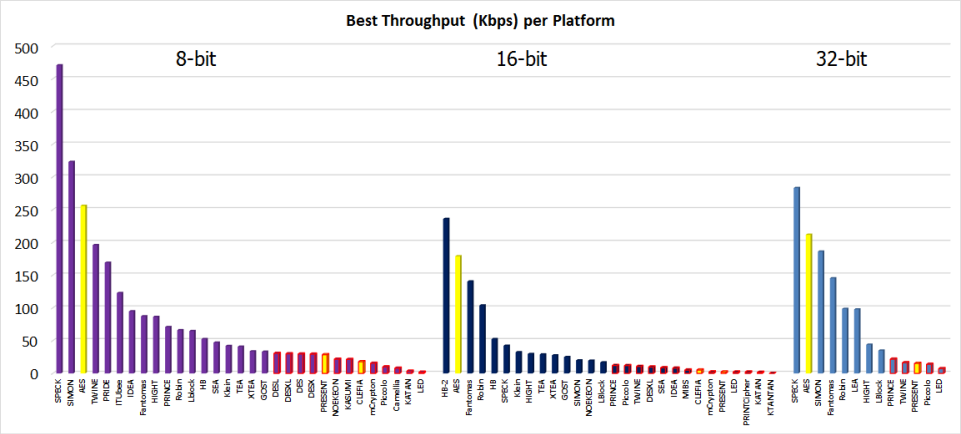

Fig. 7 Best software implementations in terms of throughput.

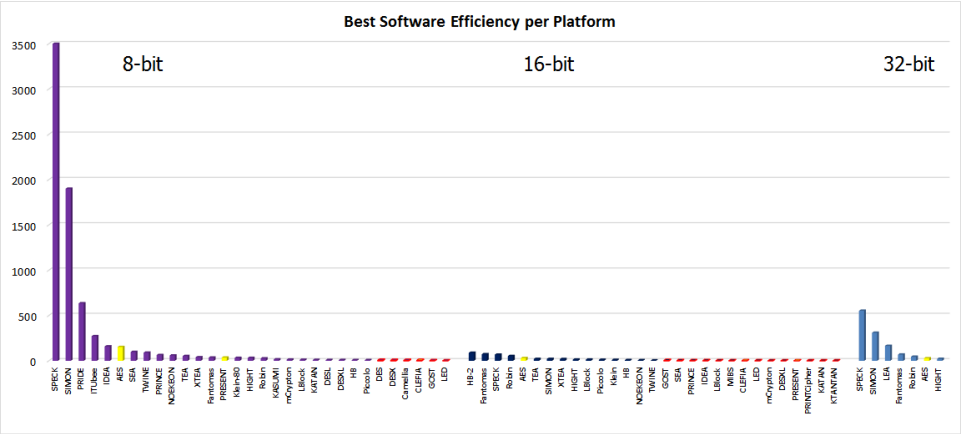

Fig. 8 Best software implementations in terms of software efficiency (Software Efficiency = Throughput [Kbps] / Code size [KB]).

and XTEA (less than 0.5KB). The implementations with the higher throughput are those of SPECK, SIMON, AES, HB-2, TWINE, PRIDE, Fantomas, ITUbee, Robin, LEA, IDEA and HIGHT (more than 85 Kbps). The lowest latency is achieved by HB-2, SPECK, SIMON, HB, TWINE, PRIDE, AES, ITUbee, IDEA and HIGHT (less than 3000 cycles). HB-2, SPECK, HB, SIMON, AES, Fantomas, TWINE, PRIDE and Robin have low energy consumption (less than 10 μJ per bit).

Figure 9 illustrates the best overall software implementations for the three microcontroller platforms and different keys. Based on the criteria adopted in this work, SPECK and PRIDE evaluate best compared to all the other proposals for many key sizes. Fantomas and Robin perform similarly well, providing additional security against side-channel attacks. AES, SEA, and ITUbee

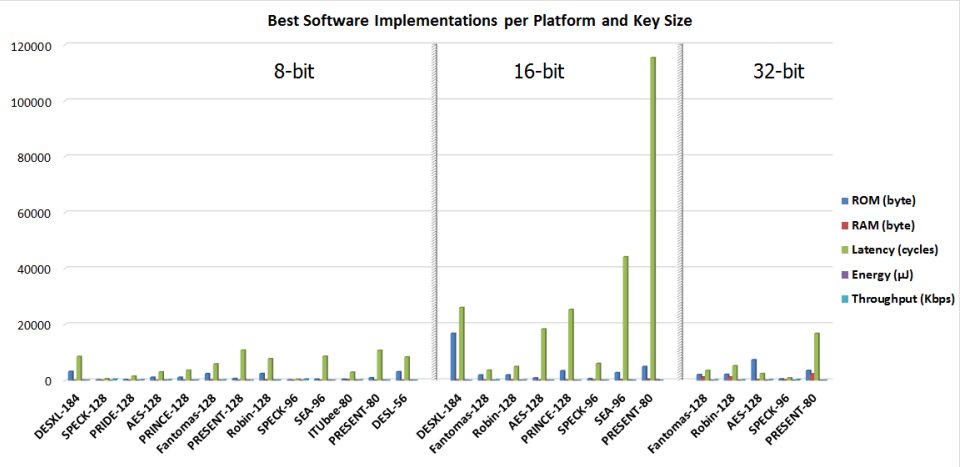

Fig. 9 Best overall software implementations per platform.

achieve a good overall status. PRESENT is slower with high latency and energy consumption. SPECK and SEA also offer efficient encryption/decryption implementations with low cost. DESXL provides higher key size, while for lower security DESL performs well.

In software implementations, code size reduction can be achieved by using low-level languages, such as the device's assembly language, and by developing segments of code that will facilitate code reuse. To reduce the memory requirements it is suggested to reuse storage space for keeping different elements, whenever it is possible. To achieve better performance it is preferred to use the device's word size as a processing unit (usually byte words). Word-wise permutations (4- or 8-bit) are preferred to bitwise permutations. Moreover, word-wise matrix multiplication operations are utilized.

The ciphers SIMON, SPECK, TWINE, Klein, LED and LBlock were designed both for lightweight hardware and compact software implementations. SIMON and SPECK shine in both domains and TWINE in hardware, while the rest of the ciphers perform reasonably well in both domains and do not excel in any of them.

5.3 Design directions

The maintenance of the internal state is the most challenging aspect of managing space requirements. The state has to be saved at each round which typically requires at least half of the total chip area and considerable power. As memory is limited in low-cost devices and read/write operations are power-demanding, the intermediate state is often kept in registers. The registers are materialized by flip-flops that have high area and power needs. Thus, lightweight ciphers try to minimize the storage requirements.

Space reduction can be achieved by lowering the block and key size. A small block or key size preserves small memory complexity. On the other hand, various security issues, such as birthday attacks, are introduced when we use block sizes that are less than 32 bits. Popular parameters for lightweight ciphers are 64-bit blocks and 80-bit keys size.

Absence of decryption functionality and hard-wired keys are techniques that can also improve the performance of applications that utilize cryptography. Some proposals tend to apply simple wiring while others avoid this approach to harden differential analysis. For applications where the implementation cost is more important than the security level, Feistel networks and simple key schedule mechanisms are a good choice. For applications were moderate security is needed, SPNs with more robust key schedule mechanisms are preferred.

5.4 S-box

S-boxes are a critical component of block ciphers, typically used to introduce nonlinearity. In software, they are implemented using look-up tables (LUT). However, these LUTs, when implemented in hardware, may have a large footprint or introduce other technical complications. Combinatorial solutions [36] are usually more efficient, where the input bits affect the complexity of the equations and the output bits affect the number of the Boolean equations required. If a combinatorial solution does not deal with the inner structure of the S-box, the occupied area will grow rapidly with the number of input and output bits. High nonlinearity demands more area, but also offers higher resistance to differential and linear analysis.

The size of the S-box is another important parameter. Large S-boxes can achieve high levels of security but consume many resources for both their hardware and software implementations. On hardware, the gate count increases exponentially with the size. However, very small S-boxes cannot provide an acceptable level of security. A good choice that balances efficiency and security, and is adopted by the majority of the lightweight ciphers, is the 4×4 S-box (as an alternative to typical 8×8 S-boxes). If higher levels of security are required, it is preferable to increase the number of rounds.

6 Conclusions

Lightweight cryptography is one of the important building blocks of secure embedded systems. This paper presented the latest developments in the field and the state-of-the-art implementations in lightweight symmetric-key cryptography. To this end, hardware and software implementations of block ciphers were examined. Moreover, the security, cost and performance properties of all the different proposals were considered and a comparative analysis was presented, with the information aggregated in the corresponding tables.

Research is ongoing, as new developments in the field constantly emerge, with novel algorithms and cryptanalysis techniques being proposed in the literature. Crypto-enabled resource-constrained devices are high in demand, a trend that is expected to continue to grow as embedded systems permeate our lives, leading to the realization of ubiquitous computing. Consequently, the authors hope that the analysis presented here will contribute to the design of robust systems and architectures, enabling secure the transition to this new reality and the *Internet of Things*.

Acknowledgement

This work was funded by the General Secretarial Research and Technology (G.S.R.T.), Hellas under the Artemis JU research program nSHIELD (new embedded Systems arcHItecturE for multi-Layer Dependable solutions) project. Call: ARTEMIS-2010-1, Grand Agreement No: 269317.

Appendix A

In this appendix, we evaluate block cipher implementations as they are reported in the literature. Table 3 indicates the features of the examined block cipher and the best publicly known cryptanalysis results. Table 4 to Table 7 and Table 8 to Table 10 summarize hardware and software implementations respectively.

Cipher	**Year**	**Key size (bits)**	**Block size (bits)**	**Rounds**	**Type**	**Analysis / Attacks**
			1st Period			
DES [131]	1977	56	64	16	Feistel	Broken under linear cryptanalysis [72]
DESX [87]	1984	184	64	16	Feistel	-
KeeLoq [63]	1985	64 (32 IV)	32	528	NLFSR	Slide and meet-in-the-middle attack [63]
GOST [111]	1989	256	64	32	Feistel	Theoretically broken under single-key setting [30]
IDEA [84]	1991	128	64	8.5	ARX	Narrow-bicliques [76]
TEA [146]	1994	128	64	Variable (64 recommended)	Feistel	Equivalent keys attack [75, 113]
XTEA [101]	1997	128	64	Variable (64 recommended)	Feistel	Related-key rectangle attack on 36 rounds [93]
AES [132]	1998	128, 192, 256	128	10, 12, 14	SPN	Biclique cryptanalysis [20]
XXTEA [147]	1998	128	Arbitrary (at least 2 words)	Depends on the block size	Feistel	Chosen-plaintext attack based on differential analysis [151]
Camellia [9]	2000	128, 192, 256	128	18, 24	Feistel	Cache timing attacks [158]
NOEKEON [32]	2000	128	128	16	SPN	Related-key cryptanalysis [79]
ICEBERG [130]	2004	128	64	16	SPN	Differential cryptanalysis on 8-round [134]
			2nd Period			
mCrypton [91]	2005	64, 96, 128	64	13	SPN	Rectangle attack for the 8-round [106]
HIGHT [60]	2006	128	64	32	ARX + GFN	Theoretically broken under related-key setting [81], biclique cryptanalysis [128], impossible differential attack on 26-round [104], zero-correlation attacks on 26- and 27-round [62]

SEA [94]	2006	Variable (96 recommended)	Variable (96, 8 recommended)	Variable	Feistel	-
PRESENT [21]	2007	80, 128	64	31	SPN	Side-channel attacks [114,149], related key attack on the 17-round [104], improved differential fault analysis [71], Biclique attacks on full round [69], truncated differential attack on 26-round [19]
CLEFIA [125]	2007	128, 192, 256	128	18, 22, 26	GFN	Improbable differential attack on reduced round [136]
KASUMI [42]	2007	128	64	8	Feistel	Broken under a related-key attack [17]
DESL [87]	2007	56	64	16	Feistel	-
DESXL [87]	2007	184	64	16	Feistel	-
PUFFIN [29]	2008	128	64	32	SPN	Broken under linear hulls [85]
PUFFIN-2 [142]	2009	80	64	34	SPN	Differential Cryptanalysis [18]
HB [38]	2009	256	16	20	Hybrid	Several attacks [119]
MIBS [66]	2009	64, 80	64	32	Feistel	Many types of attacks [14]
KATAN [25]	2009	80	32, 48, 64	254	NLFSR	Multidimensional meet-in-the-middle attacks on reduced round [159]
KTANTAN [25]	2009	80	32, 48, 64	254	NLFSR	Theoretically broken under single-key setting [2]
TWIS [103]	2009	128	128	10	GFN	Broken under differential distinguisher with probability 1 [133]
PRINTcipher [82]	2010	80, 160	48, 96	48, 96	SPN	Related-key cryptanalysis [89]
HB-2 [39]	2011	128 (64 IV)	16	-	Hybrid	Related key attack on the full cipher [120]
Klein [48]	2011	64, 80, 96	64	12, 16, 20	SPN	Practical chosen-plaintext key-recovery attacks up to 8 rounds of Klein-64 [10], biclique cryptanalysis of the full round Klein-64 [3]
LED [51]	2011	64, 80, 96, 128	64	32, 48	SPN	Biclique attacks on reduced round [69], differential fault analysis based on Super-Sbox techniques [156]
TWINE [135]	2011	80, 128	64	36	GFN	Meet-in-the-middle attacks [62]
LBlock [148]	2011	80	64	32	Feistel	A saturation attack against 22-round [135], biclique cryptanalysis on full round [143], improved differential fault attacks [70], round addition differential fault analysis [152]
Piccolo [124]	2011	80, 128	64	25, 31	GFN	Biclique attacks on full round [69,128], impossible differential cryptanalysis on reduced round versions [11]
EPCBC [150]	2011	96	48, 96	32	SPN	Algebraic cryptanalysis on reduced rounds [141]
3rd Period						
PICARO [107]	2012	128	-	12	SPN	-
PRINCE [22]	2012	128	64	12	SPN	Attacks on the reduced 6-round and the full 12-round [68,127], truncated differential cryptanalysis on 7-round [157]
SIMON [15]	2013	64, 72, 96, 128, 144, 192, 256	32, 48, 64, 96, 128	32, 36, 42, 44, 52, 54, 68, 69, 72	Feistel	Attacks on reduced versions [6], differential fault attacks [139], cube and dynamic cube attacks on SIMON32/64 [112]
SPECK [15]	2013	64, 72, 96, 128, 144, 192, 256	32, 48, 64, 96, 128	22, 23, 26, 27, 28, 29, 32, 33, 34	ARX	Attacks on reduced versions [1], differential fault attacks [139]
Zorro [46]	2013	128	128	24	SPN	Practical invariant subspace attacks [86]
ITUbee [74]	2013	80	80	-	Feistel	Self-similarity cryptanalysis on 8-round [126]
LEA [61]	2013	128, 192, 256	128	24, 28, 32	ARX	Power analysis [77]
RECTANGLE [155]	2014	80, 128	64	25	SPN	-
FeW [83]	2014	80, 128	64	32	Feistel-M	-
Halka [34]	2014	80	64	24	NLFSR	-
BEST-1 [67]	2014	128	64	12	ARX	-

Robin [49]	2014	128	128	16	Feistel + LS-designs	Practical invariant subspace attacks [86]
Fantomas [49]	2014	128	128	12	Feistel + LS-designs	-
HISEC [5]	2014	80	64	15	GFN	-
I-PRESENT^{TM} [154]	2014	80, 128	64	30	SPN	-
PRESENT-GRP [12]	2014	128	64	31	Hybrid	-
PRIDE [4]	2014	128	64	20	SPN	-

Table 3: The general characteristics of each examined cipher

Cipher **Better is**	**Key size (bits)**	**Block size (bits)**	**Latency (Cycles / block)** **Lower**	**Throughput at 100 KHz (Kbps)** **Higher**	**Tech (μm)**	**Area (GE)** **Lower**	**Efficiency (Kbps/KGE)** **Higher**	**Power (μW)** **Lower**	**Energy (μJ / bit)** **Lower**
128-bits key									
SPNs									
NOEKEON [108]	128	128	3720	3.44	0.18	2862	1.20	4.30	1247.65
ICEBERG [28]	128	64	16	400	0.18	5817	68.76	8.72	21.81
PRESENT [150]	128	64	528	12.12	0.18	1339	9.05	2.00	165.24
PRESENT [110]	128	64	559	11.45	0.18	1391	8.23	2.08	182.24
PRESENT [21]	128	64	32	200	0.18	1886	106.04	2.82	14.14
LED [51]	128	64	1872	3.4	0.18	1265	2.68	1.89	555.02
mCrypton [108]	128	64	190	33.51	0.18	2760	12.14	4.14	122.90
PUFFIN [29]	128	64	32	200	0.18	2577	77.60	3.86	19.32
I-PRESENT^{TM} (D) [154]	128	64	-	-	0.18	2783	-	4.17	-
Feistel									
Camellia [123]	128	128	44	290.1	-	6511	44.55	9.76	33.57
TEA [153]	128	64	64	100	0.18	2355	42.46	3.53	35.32
Hybrid									
PRESENT-GPR [12]	128	64	-	-	0.18	2125	-	3.18	-
HB2 [39]	128	16	4	400	0.18	3220	124.22	4.83	12.07
-//-	128	16	16	100	0.18	2332	42.88	3.50	34.98
-//- [39]	128	16	20	80	0.18	2159	37.05	3.23	40.48
96-bits key									
SPNs									
EPCBC [150]	96	96	792	12.12	0.18	1333	9.09	2.00	164.95
-//-	96	48	396	12.12	0.18	1008	12.02	1.51	124.74
Klein [48]	96	64	335	19.1	0.18	1528	12.50	2.30	119.97
LED [51]	96	64	1872	3.4	0.18	1116	3.04	1.67	489.64
Feistel									
SEA [108]	96	8	243	3.29	0.18	2569	1.28	3.85	1170.50
-//-	96	8	11	70.59	0.18	2941	24.00	4.41	60.65
80-bits key									
SPNs									
Klein [48]	80	64	271	23.62	0.18	1478	15.98	2.21	93.87
LED [51]	80	64	1872	3.4	0.18	1040	3.26	1.56	456.3

PRESENT [117]	80	64	547	11.7	0.18	1075	10.88	1.61	137.81
PRESENT [150]	80	64	516	12.4	0.18	1030	12.03	1.54	124.56
PRESENT [21]	80	64	32	200	0.18	1570	127.38	2.35	11.77
I-PRESENT^{TM} (D) [154]	80	64	-	-	0.18	2467	-	370	-
PRINTcipher(H)(S) [82]	80	48	768	6.25	0.18	402	15.54	0.60	96.48
PRINTcipher [82]	80	48	48	100	0.18	503	198.80	0.75	7.54
PUFFIN-2 [142]	80	64	1240	5.2	0.18	1083	4.80	1.62	314.74
Feistel									
LBlock [148]	80	64	32	200	0.18	1320	151.51	2.00	9.9
MIBS [66]	80	64	32	200	0.18	1530	130.71	2.30	11.47
GFN									
HISEC [5]	80	64	-	-	0.18	1695	-	2.54	-
64-bits key									
SPNs									
Klein [48]	64	64	207	30.9	0.18	1220	25.32	1.83	59.18
LED [51]	64	64	1248	5.1	0.18	966	5.27	1.45	282.55
Feistel									
MIBS [66]	64	64	32	200	0.18	1396	143.26	2.09	10.47
56-bits key									
Feistel									
DES(S) [87]	56	64	144	44.4	0.18	2309	19.22	3.46	77.92
DESL [87]	56	64	144	44.4	0.18	1848	24.02	2.77	62.37
160-bits key									
SPNs									
PRINTcipher(H)(S) [82]	160	96	3072	3.13	0.18	726	4.31	1.09	348.48
PRINTcipher [82]	160	96	96	100	0.18	967	103.41	1.45	14.50
184-bits key									
Feistel									
DESX(S) [87]	184	64	144	44.4	0.18	2629	16.88	3.94	88.72
DESXL(S) [87]	184	64	144	44.4	0.18	2168	20.47	3.25	73.17
256-bits key									
Feistel									
GOST [111]	256	64	32	200	0.18	1017	196.656834	1.52	7.62
-//-	256	64	32	200	0.18	1000	200.00	1.50	7.5
GOST(S) [111]	256	64	264	24.24	0.18	651	37.23	0.97	40.28
-//-	256	64	264	24.24	0.18	800	30.30	1.20	49.5

Table 4: Hardware implementations of block ciphers on $0.18\mu m$ technology

Cipher **Better is**	**Key size (bits)**	**Block size (bits)**	**Latency (Cycles / block)** **Lower**	**Throughput at 100 KHz (Kbps)** **Higher**	**Tech (μm)**	**Area (GE)** **Lower**	**Efficiency (Kbps/KGE)** **Higher**	**Power (μW)** **Lower**	**Energy (μJ / bit)** **Lower**
128-bits key									
SPNs									
AES(S) [99]	128	128	226	56.64	0.13	2400	23.60	2.4	42.38
AES(M) [99]	128	128	226	48	0.13	11031	4.35	11.03	194.78
NOEKEON [108]	128	128	3724	3.4	0.13	2880	1.18	2.88	837.90
SIMON [15]	128	128	559	22.9	0.13	1317	17.38	1.32	57.52
-//-	128	64	368	17.4	0.13	1000	17.40	1	57.50
SPECK [15]	128	128	1058	12.1	0.13	1396	8.66	1.40	115.39
-//-	128	64	464	13.8	0.13	1127	12.24	1.12	81.70

PRINCE [13]	128	64	12	533.3	0.13	2953	180.59	2.95	5.53
PRINCE [22]	128	64	12	533.3	0.13	3491	152.76	3.50	6.54
LED [13]	128	64	48	133.33	0.13	3194	41.74	3.20	23.95
RECTANGLE [155]	128	64	26	246	0.13	1787	137.66	1.78	7.25
mCrypton [91]	128	64	13	492.3	0.13	2949	166.93	3.00	6.00
mCrypton(D) [91]	128	64	13	492.3	0.13	4108	119.83	4.10	8.34
mCrypton [108]	128	64	191	33.5	0.13	2709	12.36	2.71	80.85
Feistel									
XTEA [73]	128	64	32	200	0.13	2521	79.33	2.52	12.60
GFNs									
CLEFIA(S) [7]	128	128	328	39	0.13	2488	15.67	2.48	63.76
CLEFIA(S) (D) [7]	128	128	328 / 320	39 / 40	0.13	2604	14.97 / 15.36	2.60	66.72
CLEFIA [7]	128	128	176	76	0.13	2678	28.37	2.67	36.82
CLEFIA (D) [7]	128	128	176	76	0.13	2781	27.32	2.78	38.23
KASUMI (D) [122]	128	64	56	115.14	0.13	3437	33.50	3.44	29.90
Piccolo-128 [124]	128	64	33	193.94	0.13	1197	162.02	1.20	6.18
Piccolo-128(D) [124]	128	64	33	193.85	0.13	1362	142.32	1.36	7.03
Piccolo-128(S) [124]	128	64	528	12.12	0.13	758	15.98	0.75	62.54
Piccolo-128 (D)(S) [124]	128	64	528	12.12	0.13	818	14.81	0.81	67.49
ARX									
LEA [88]	128	128	168	76.19	0.13	3826	19.91	3.82	50.22
-//-	128	128	96	133.3	0.13	4296	31.02	4.30	32.22
-//-	128	128	24	533.3	0.13	5426	98.28	5.42	10.17
96-bits key									
SPNs									
mCrypton-96 [91]	96	64	13	492.3	0.13	2681	183.62	2.68	5.45
SIMON [15]	96	48	304	15.8	0.13	763	20.70	0.76	48.32
-//-	96	64	352	18.2	0.13	838	21.71	0.83	46.09
-//-	96	64	45	142.2	0.13	1216	116.94	1.21	8.55
-//-	96	96	649	14.8	0.13	984	15.04	0.98	66.52
SPECK [15]	96	48	400	12	0.13	884	13.57	0.88	73.67
-//-	96	64	441	14.5	0.13	984	14.73	0.98	67.80
-//-	96	64	29	220.7	0.13	1522	145.00	1.52	6.89
-//-	96	96	696	13.8	0.13	1134	12.16	1.13	82.22
Feistel									
SEA(D) [94]	96	96	93	103	0.13	3758	27.40	3.75	36.41
SEA [94]	96	8	93	258	0.13	3758	68.65	3.75	436.87
-//-	96	8	93	258	0.13	4313	59.81	4.31	501.39
SEA [108]	96	8	349	2.29	0.13	2562	0.89	2.56	1117.67
-//-	96	8	11	70.59	0.13	2854	24.73	2.85	39.24
80-bits key									
SPNs									
PRESENT [13]	80	64	31	206	0.13	2195	93.84	2.20	10.63
RECTANGLE [155]	80	64	26	246	0.13	1467	167.68	1.46	5.96
GFNs									
Piccolo-80 [124]	80	64	27	237.04	0.13	1136	208.66	1.13	4.80
Piccolo-80(D) [124]	80	64	27	237	0.13	1274	186.02	1.27	5.38
Piccolo-80(S) [124]	80	64	432	14.81	0.13	683	21.68	0.68	46.10
Piccolo-80(D)(S) [124]	80	64	432	14.81	0.13	743	19.93	0.74	50.15
NLFSR									
KATAN(S) [25]	80	32	256	12.5	0.13	802	15.58	0.80	64.16
-//-	80	48	256	18.8	0.13	927	20.28	0.92	49.44
-//-	80	64	255	25.1	0.13	1054	23.81	1.05	42.00
KTANTAN(H)(S) [25]	80	32	256	12.5	0.13	462	27.05	0.46	36.96
-//-	80	48	256	18.8	0.13	588	31.97	0.58	31.36

-//-	80	64	255	25.1	0.13	688	36.48	0.68	27.41
64-bits key									
SPNs									
Klein [13]	64	64	12	533.33	0.13	2760	193.23	2.76	5.17
Klein(S) [13]	64	64	98	65.3	0.13	1432	45.60	1.43	21.92

Table 5: Hardware implementations of block ciphers on $0.13\mu m$ technology

Cipher **Better is**	**Key size (bits)**	**Block size (bits)**	**Latency (Cycles / block)** **Lower**	**Throughput at 100 KHz (Kbps)** **Higher**	**Tech (μm)**	**Area (GE)** **Lower**	**Efficiency (Kbps/KGE)** **Higher**	**Power (μW)** **Lower**	**Energy (μJ / bit)** **Lower**
128-bits key									
SPNs									
PUFFIN [78]	128	64	32	200	0.09	2303	86.848	1.61	8.06
Feistel									
CLEFIA [36]	128	128	36	355.56	0.09	4993	71.21	3.50	9.83
CLEFIA [125]	128	128	36	355.56	0.09	4950	71.83	3.45	9.74
CLEFIA(D) [125]	128	128	18	711	0.09	5979	118.91	4.18	5.88
XTEA [78]	128	64	32	200	0.09	3490	57.30	2.44	12.21
GFNs									
TWINE [135]	128	64	36	178	0.09	1866	95.39	1.30	7.34
TWINE(D) [135]	128	64	36	178	0.09	2285	77.89	1.60	9.00
80-bits key									
SPNs									
PRESENT [78]	80	64	31	206.4	0.09	1704	121.12	1.19	5.77
GFNs									
TWINE(S) [135]	80	64	540	11.8	0.09	1116	10.57	0.78	65.91
TWINE [135]	80	64	36	178	0.09	1503	118.42	1.05	5.91
TWINE(D) [135]	80	64	36	178	0.09	1799	98.94	1.25	7.08
56-bits key									
Feistel									
DESL [78]	56	64	16	400	0.09	2762	144.82	1.93	4.83
184-bits key									
Feistel									
DESXL [78]	184	64	16	400	0.09	3082	129.78	2.15	5.39

Table 6: Hardware implementations of block ciphers on $0.09\mu m$ technology

Cipher **Better is**	**Key size (bits)**	**Block size (bits)**	**Latency (Cycles / block)** **Lower**	**Throughput at 100 KHz (Kbps)** **Higher**	**Tech (μm)**	**Area (GE)** **Lower**	**Efficiency (Kbps/KGE)** **Higher**	**Power (μW)** **Lower**	**Energy (μJ / bit)** **Lower**
128-bits key									
SPNs									
AES(D) [36]	128	128	1032	12.4	0.35	3400	3.64	6.12	493.42
NOEKEON [108]	128	128	3720	3.44	0.35	2604	1.32	4.68	1362.21
mCrypton [108]	128	64	190	33.51	0.35	2594	12.91	4.66	138.61
PRESENT [117]	80	64	561	11.4	0.35	1000	11.40	1.80	157.78
GFNs									

HIGHT [60]	128	64	34	188	0.25	3048	61.67	5.48	29.14
HIGHT(D) [92]	128	64	34	188	0.35	2608	72.08	4.70	24.93
TWIS [103]	128	-	-	-	-	-	-	-	-
Feistel									
Camellia [123]	128	128	44	290	-	6511	44.54	11.71	40.28
TEA [65]	128	64	512	6.25	0.35	3872	1.61	7.00	557.56
XTEA [108]	128	64	705	9.08	0.35	2636	3.44	4.74	522.66
TinyXTEA-1 [73]	128	64	240	26.7	0.35	3500	7.62	6.30	236.25
TinyXTEA-3 [73]	128	64	112	57	0.35	3490	16.33	6.28	109.93
96-bits key									
Feistel									
SEA [108]	96	8	243	3.29	0.35	2300	1.43	4.14	1257.52
-//-	96	8	11	70.59	0.35	2679	26.34	4.82	66.30

Table 7: Hardware implementations of block ciphers on 0.25 and 0.35μm technology

Cipher **Better is**	**Key size (bits)**	**Block size (bits)**	**ROM (byte)** **Lower**	**RAM (byte)** **Lower**	**Latency (Cycles / block)** **Lower**	**Energy (μJ / bit)** **Lower**	**Throughput at 4 MHz (Kbps)** **Higher**	**Efficiency (Kbps/KB)** **Higher**
128-bits key								
SPNs								
AES(D) [37]	128	128	1659	33	4557 / 7015	19.2	112 / 72	67.51
AES(D) [23]	128	128	1912	432	2000 / 2896	8.0	256 / 176	133.89
AES(D) [36]	128	128	2606	0	6637 / 7429	26.5	77.1 / 68	29.58
AES(D) [109]	128	128	1704	-	5064 / 8226	20.2	101 / 62	59.27
-//-	128	128	1940	-	3304 / 5037	13.2	154 / 101	79.38
-//-	128	128	2158	-	3084 / 4505	12.3	166 / 113	76.92
AES [109]	128	128	918	-	4192 / -	16.7	122 / -	132.89
-//-	128	128	1110	-	3004 / -	12.0	170 / -	153.15
AES(D) [24]	128	128	399	10	23011 / 33000	92.0	22.25 / 15.50	55.76
AES(D) [35]	128	128	26800	551	58984 / 60319	235.9	8.68 / 8.48	0.32
AES [35]	128	128	2742	127	22603 / -	90.4	22.65 / -	8.26
-//-	128	128	12320	359	6651 / -	76.6	76.98 / -	6.24
AES [40]	128	128	1570	-	3159 / -	12.6	162 / -	103.18
NOEKEON(D) [37]	128	128	364	32	23517 / 23502	95.9	21.7 / 21.7	59.61
PRESENT [40]	128	64	660	-	10792 / -	43.1	23.7 / -	35.90
PRINCE(D) [35]	128	64	5650	241	139794 / 140764	559.1	1.83 / 1.81	0.32
PRINCE [35]	128	64	4300	63	17207 / -	68.8	14.87 / -	3.45
-//-	128	64	11262	89	13405 / -	53.6	19.09 / -	1.69
PRINCE [4]	128	64	1108	-	3614 / -	14.4	70.8 / -	63.89
PRIDE [4]	128	64	266	0	1514 / -	6.0	169 / -	635.33
Feistel								
Camellia(D) [24]	128	128	1262	12	64000 / 68260	256.0	8 / 7.5	6.33
Robin(D) [35]	128	128	4950	266	69199 / 76974	276.7	7.39 / 6.65	1.49
Robin [35]	128	128	2434	103	7760 / -	31.0	65.97 / -	27.10
Fantomas(D) [35]	128	128	5898	262	51471 / 60230	205.8	9.94 / 8.50	1.68
Fantomas [35]	128	128	2400	103	5866 / -	23.4	87.28 / -	36.36
SIMON [15]	128	64	246	0	901 / -	3.6	284 / -	1154.47
-//-	128	128	686	0	2723 / -	10.8	188 / -	274.05
KASUMI(D) [37]	128	64	1264	24	11939 / 11939	47.6	21.4 / 21.4	16.93
TEA(D) [37]	128	64	648	24	7408 / 7539	30.3	34.5 / 33.9	53.24
TEA(D) [36]	128	64	1140	0	6271 / 6299	34.3	40.8 / 40.6	35.78

XTEA(D) [109]	128	64	504	-	17514 / 19936	70.0	14.6 / 12.8	28.96
-//-	128	64	820	-	7786 / 8928	31.1	32.8 / 28.6	40.00
-//-	128	64	1246	-	7595 / 8735	30.3	33.7 / 29.3	27.04
GFN								
CLEFIA [40]	128	128	3046	-	28648 / -	114.5	17.8 / -	5.84
ARX								
SPECK [15]	128	128	396	0	1333 / -	5.3	384 / -	969.69
-//-	128	64	186	0	599 / -	2.3	427.5 / -	2298.38
HIGHT(D) [37]	128	64	402	32	19503 / 20159	79.8	13.1 / 12.6	32.58
HIGHT(D) [36]	128	64	5672	0	2964 / 2964	11.8	86.3 / 86.3	15.21
HIGHT(D) [48]	128	64	2510	117	7377 / 5844	29.5	34.7 / 43.8	13.82
HIGHT(D) [35]	128	64	2608	342	87694 / 83464	350.7	2.91 / 2.44	1.11
HIGHT [35]	128	64	1084	54	11399 / -	45.5	22.45 / -	20.71
-//-	128	64	5718	47	6377 / -	25.5	40.14 / -	7.01
IDEA(D) [37]	128	64	836	232	8250 / 22792	34.3	31 / 11	37.08
IDEA(D) [36]	128	64	596	0	2700 / 15393	10.8	94.8 / 16	159.06
Hybrid								
HB(D) [48]	128	16	2646	159	4324 / 9142	17.2	14.8 / 7	5.59
96-bits key								
SPNs								
mCrypton(D) [37]	96	64	1076	28	16457 / 22656	68	15.5 / 11.2	14.40
Feistel								
SEA(D) [37]	96	96	426	24	41604 / 40860	173.7	9.2 / 9.3	21.59
SEA(D) [36]	96	96	2132	0	9654 / 9654	38.6	39 / 39	18.29
SEA(D) [109]	96	96	332	-	14723 / 14723	58.8	26 / 26	78.31
-//-	96	96	448	-	8597 / 8597	34.3	44 / 44	98.21
-//-	96	96	786	-	8053 / 8053	32.2	47 / 47	59.79
SIMON [15]	96	96	416	0	1533 / -	6.1	250.5 / -	602.16
-//-	96	64	198	168	1741 / -	6.9	147 / -	742.42
-//-	96	64	238	0	861 / -	3.4	297.5 / -	1250.00
SIMON(D) [35]	96	64	2476	375	199948 / 200950	799.7	1.28 / 1.27	0.51
SIMON [35]	96	64	878	65	24305 / -	97.2	10.53 / -	11.99
SIMON [15]	96	48	170	0	594 / -	2.3	323 / -	1900.00
ARX								
SPECK [15]	96	96	276	0	887 / -	3.5	433 / -	1568.84
-//-	96	64	152	108	1232 / -	4.9	207.8 / -	1367.10
-//-	96	64	182	0	577 / -	2.3	444 / -	2439.56
SPECK(D) [35]	96	64	1692	300	121953 / 224635	487.8	2.09 / 1.13	1.23
SPECK [35]	96	64	572	49	14003 / -	56.0	18.28 / -	31.95
SPECK [15]	96	48	134	0	408 / -	1.6	470.5 / -	3511.19
80-bits key								
SPNs								
Klein-80(D) [37]	80	64	1268	18	6095 / 7658	25.1	42 / 33	33.12
PRESENT(D) [37]	80	64	1000	18	11342 / 13599	45.3	22.5 / 18.8	22.50
PRESENT(D) [36]	80	64	936	0	10723 / 11239	42.8	23.8 / 22.7	25.42
PRESENT(D) [110]	80	64	2398	528	9592 / 9824	38.3	26.6 / 26	11.09
PRESENT(D) [109]	80	64	920	-	28062 / 60427	112.2	9.1 / 4.2	9.89
-//-	80	64	1148	-	15042 / 17677	60.1	17 / 14.4	14.80
-//-	80	64	2146	-	8958 / 11592	35.8	28.5 / 22	13.28
PRESENT(D) [35]	80	64	11208	591	1616360 / 3261346	6465.4	0.15 / 0.07	0.01
PRESENT [35]	80	64	1562	83	1937461 / -	7749.8	0.13 / -	0.08
-//-	80	64	4504	2157	160631 / -	642.5	1.59 / -	0.35
LED(D) [35]	80	64	4538	274	1141377 / 1493451	4565.5	0.22 / 0.17	0.04
LED [35]	80	64	2482	86	143253 / -	573.0	1.78 / -	0.71
-//-	80	64	2428	262	134997 / -	539.9	1.89 / -	0.77
PRINTCipher-48 [48]	80	48	6210	128	- / -	-	- / -	-

Feistel								
ITUbee [74]	80	80	716	0	2607 / -	10.4	122.7 / -	171.36
-//-	80	80	400	186	2937 / -	11.7	109 / -	272.50
LBlock [148]	80	64	-	-	3955 / -	15.8	64.7 / -	-
LBlock(D) [35]	80	64	3086	331	108148 / 104766	432.5	2.36 / 2.44	0.76
LBlock [35]	80	64	1268	46	16473 / -	65.8	15.54 / -	12.25
-//-	80	64	8336	50	10263 / -	41.0	24.94 / -	2.99
GFNs								
TWINE(D) [135]	80	64	1304	414	2168 / 2168	8.6	118 / 118	90.49
-//-	80	64	728	335	18794 / 18689	75.1	13.6 / 13.6	18.68
-//-	80	64	792	191	18794 / 18688	75.1	13.6 / 13.6	17.17
-//-	80	64	2294	386	1301 / 1302	5.2	196 / 196	85.44
TWINE(D) [35]	80	64	3610	402	199574 / 190304	798.2	1.28 / 1.34	0.35
TWINE [35]	80	64	1408	59	21637 / -	86.5	11.83 / -	8.40
Piccolo(D) [35]	80	64	2654	319	203616 / 205898	814.4	1.25 / 1.24	0.47
Piccolo [35]	80	64	1178	65	25681 / -	102.7	9.96 / -	8.45
NLFSR								
KATAN(D) [37]	80	64	338	18	72063 / 88525	289.2	3.5 / 2.8	10.35
56-bits key								
Feistel								
DES(D) [36]	56	64	4314	0	8633 / 8154	34.5	29.6 / 31.3	6.86
DESL(D) [115]	56	64	3098	0	8365 / 7885	33.4	30.6 / 32	9.87
184-bits key								
Feistel								
DESXL(D) [37]	184	64	820	48	84602 / 84602	348.2	3 / 3	3.65
DESXL(D) [36]	184	64	3192	0	8531 / 7961	34.1	30 / 32	9.39
DESX(D) [115]	184	64	4406	0	8699 / 8220	34.7	29.4 / 31	6.67
256-bits key								
Feistel								
GOST(D) [48]	256	64	14342	233	7734 / 7573	30.9	33.1 / 33.8	2.30
Hybrid								
HB(D) [38]	256	16	3680	1308	3664 / 3868	14.6	17.4 / 16.5	4.72
-//-	256	16	2950	1064	2414 / 2650	9.6	26.5 / 24.1	8.98
-//-	256	16	30500	10918	1399 / 1635	5.5	45.7 / 39.1	1.49
-//-	256	16	3760	1360	1220 / 1461	4.8	52.4 / 43.8	13.93

Table 8: Software implementations of block ciphers on 8-bit microcontrollers

Cipher **Better is**	**Key size (bits)**	**Block size (bits)**	**ROM (byte) Lower**	**RAM (byte) Lower**	**Latency (Cycles / block) Lower**	**Energy (μJ / bit) Lower**	**Throughput at 4 MHz (Kbps) Higher**	**Efficiency (Kbps/KB) Higher**
128-bits key								
SPNs								
AES(D) [27]	128	128	4460	19	30257 / 38508	40.8	17 / 13.3	3.81
AES [48]	128	128	900	60	18351 / -	24.7	27.9 / -	31.00
AES(D) [35]	128	128	20726	574	28979 / 31309	39.1	17.66 / 16.35	0.85
AES [35]	128	128	3124	124	33386 / -	45.0	15.33 / -	4.90
-//-	128	128	8844	92	2862 / -	3.8	178.89 / -	20.22
NOEKEON(D) [27]	128	128	2710	34	26291 / 27129	35.4	19.4 / 18.8	7.15
-//-	128	128	2784	34	52564 / 53435	70.9	9.7 / 9.5	3.48
PRINCE(D) [35]	128	64	4174	240	202414 / 203273	273.2	1.26 / 1.25	0.30
PRINCE [35]	128	64	3418	70	25340 / -	34.2	10.10 / -	2.95
-//-	128	64	15728	76	21124 / -	28.5	12.11 / -	0.76

mCrypton(D) [27]	128	64	3108	24	108415 / 220568	146.3	2.3 / 1.1	0.74
LED(D) [27]	128	64	2648	41	1341488 / 1345152	1811.0	0.2 / 0.2	0.07
-//-	128	64	2948	41	268721 / 274963	362.7	0.9 / 0.9	0.30
-//-	128	64	2264	41	171056 / 173832	230.9	1.4 / 1.4	0.61
Feistel								
Robin(D) [35]	128	128	3170	238	39350 / 37528	53.1	13.01 / 13.64	4.10
Robin [35]	128	128	1942	80	4935 / -	6.6	103.74 / -	53.41
Fantomas(D) [35]	128	128	4164	234	29038 / 28392	39.2	17.63 / 18.03	4.23
Fantomas [35]	128	128	1920	78	3646 / -	4.9	140.42 / -	73.13
TEA(D) [27]	128	64	1354	13	8785 / 9129	11.8	29 / 28	21.41
XTEA(D) [27]	128	64	1394	11	9287 / 9631	12.5	27.5 / 26.5	19.72
GFNs								
CLEFIA(D) [27]	128	128	4780	180	98145 / 101855	132.4	5.2 / 5.0	1.08
Piccolo(D) [27]	128	64	2510	91	36497 / 39600	49.2	7.0 / 6.4	2.78
TWINE(D) [27]	128	64	2216	23	82003 / 60932	110.7	3.1 / 4.2	1.39
ARX								
IDEA(D) [27]	128	64	3140	82	31402 / 163380	42.3	8.1 / 1.5	2.57
HIGHT(D) [27]	128	64	3130	18	32372 / 32623	43.7	7.9 / 7.8	2.52
HIGHT(D) [48]	128	64	2050	40	8620 / 8620	11.6	29.7 / 29.7	14.48
HIGHT(D) [35]	128	64	2368	342	215107 / 209838	290.3	1.19 / 1.21	0.50
HIGHT [35]	128	64	980	62	26728 / -	36.0	9.57 / -	9.76
-//-	128	64	13780	64	21882 / -	29.5	11.69 / -	0.84
Hybrid								
HB(D) [48]	128	16	1822	82	4637 / 9697	6.2	13.8 / 6.6	7.57
HB-2(D) [39]	128	16	770	50	1520 / 1544	2.0	42.1 / 41.4	54.67
-//-	128	16	2518	114	576 / 729	0.7	111 / 87.7	44.08
-//-	128	16	3648	114	359 / 560	0.4	178 / 114	48.79
-//-	128	16	3600	114	745 / 930	1.0	85.9 / 68.8	23.86
-//-	128	16	4178	114	574 / 770	0.7	111 / 83.1	26.56
-//-	128	16	3200	1500	495 / 652	0.6	129 / 98.1	40.31
-//-	128	16	2227	114	319 / 371	0.4	200 / 172	89.80
-//-	128	16	4959	114	271 / 362	0.3	236 / 176	47.59
-//-	128	16	2200	116	332 / 336	0.4	192 / 190	87.27
96-bits key								
SPNs								
mCrypton(D) [27]	96	64	2834	20	108499 / 220320	146.4	2.3 / 1.1	0.81
Klein-96(D) [27]	96	64	5862	39	51502 / 170789	69.5	5.0 / 1.5	0.85
Feistel								
SEA(D) [27]	96	96	2804	24	119455 / 120158	161.2	3.2 / 3.1	1.14
SEA(D) [48]	96	96	2754	204	44138 / 41739	59.5	8.7 / 9.2	3.15
SIMON(D) [35]	96	64	8158	392	107671 / 111970	145.3	2.37 / 2.28	0.29
SIMON [35]	96	64	940	82	12902	17.4	19.84 / -	21.10
ARX								
SPECK(D) [35]	96	64	1342	300	51621 / 45248	69.6	4.95 / 5.65	3.68
SPECK [35]	96	64	618	58	6054 / -	8.1	42.28 / -	68.41
80-bits key								
SPNs								
PRESENT(D) [35]	80	64	12928	1042	939518 / 1941769	1268.3	0.27 / 0.13	0.02
PRESENT [35]	80	64	3650	352	3497578 / -	4721.7	0.07 / -	0.01
-//-	80	64	4960	396	115338 / -	155.7	2.21 / -	0.44
LED(D) [35]	80	64	7004	252	1186982 / 1319409	1602.4	0.21 / 0.19	0.02
LED [35]	80	64	4042	96	694812 / -	937.9	0.35 / -	0.08
-//-	80	64	4422	104	148334 / -	200.2	1.72 / -	0.38
Klein-80(D) [27]	80	64	5676	38	40278 / 135369	54.3	6.3 / 1.9	1.10

PRINTCipher-48(D) [48]	80	48	6210	48	87272 / 64000	117.8	2.2 / 3.0	0.35
Feistel								
LBlock(D) [27]	80	64	3568	13	42954 / 22005	57.9	6.0 / 11.6	1.68
LBlock(D) [35]	80	64	2024	328	157923 / 161886	213.1	1.62 / 1.58	0.80
LBlock [35]	80	64	976	58	18988 / -	25.6	13.48 / -	13.81
-//-	80	64	10556	64	15212 / -	20.5	16.82 / -	1.59
MIBS(D) [27]	80	64	3138	16	58688 / 39842	79.2	4.3 / 6.4	1.37
GFNs								
Piccolo(D) [27]	80	64	2434	79	32106 / 34630	43.3	7.8 / 7.5	3.20
Piccolo(D) [35]	80	64	1824	318	172477 / 178280	232.8	1.48 / 1.43	0.81
Piccolo [35]	80	64	966	70	21448 / -	28.9	11.93 / -	12.34
TWINE(D) [35]	80	64	3452	352	285493 / 287712	385.4	0.89 / 0.88	0.25
TWINE [35]	80	64	1570	72	34778 / -	46.9	7.36 / -	4.68
-//-	80	64	1922	136	23938 / -	32.3	10.69 / -	5.56
NLFSR								
KATAN(D) [27]	80	32	5816	1881	744279 / 717056	1004.7	1.7 / 1.7	0.29
-//-	80	48	7076	1969	1127271 / 1053680	1521.8	1.7 / 1.8	0.24
-//-	80	64	8348	1953	1518391 / 1397924	2049.8	1.6 / 1.8	0.19
KTANTAN(H)(D) [27]	80	32	10516	614	10233211 / 10193489	13814.8	0.012 / 0.012	0.001
-//-	80	48	11764	702	10614933 / 10525067	14330.1	0.018 / 0.018	0.001
-//-	80	64	16252	790	11004783 / 10864265	14856.4	0.023 / 0.023	0.001
64-bits key								
SPNs								
mCrypton(D) [27]	64	64	2726	18	107803 / 219870	145.5	2.3 / 1.1	0.84
Klein-64(D) [27]	64	64	5486	36	29514 / 100600	39.8	8.6 / 2.5	1.56
Klein-64(D) [48]	64	64	2980	50	7901 / 10000	10.6	32.4 / 25.6	10.87
LED(D) [27]	64	64	2670	41	894680 / 897352	1207.8	0.2 / 0.2	0.07
-//-	64	64	2498	41	212409 / 217401	286.7	1.2 / 1.1	0.48
-//-	64	64	2264	41	114872 / 116280	155.0	2.2 / 2.2	0.97
Feistel								
MIBS(D) [27]	64	64	3184	29	49056 / 52890	66.2	5.2 / 4.8	1.63
184-bits key								
Feistel								
DESXL(D) [27]	184	64	16816	112	26055 / 66913	35.1	9.8 / 3.8	0.58
256-bits key								
Feistel								
GOST(D) [48]	256	64	4748	190	10240 / 10118	13.8	25 / 25.3	5.26
Hybrid								
HB(D) [38]	256	16	3680	1308	3664 / 3868	4.9	17.4 / 16.5	4.72
-//-	256	16	2950	1064	2414 / 2650	3.2	26.5 / 24.1	8.98
-//-	256	16	30500	10918	1399 / 1635	1.8	45.7 / 39.1	1.49
-//-	256	16	3760	1360	1220 / 1461	1.6	52.4 / 43.8	13.93

Table 9: Software implementations of block ciphers on 16-bit microcontrollers

Cipher **Better is**	**Key size (bits)**	**Block size (bits)**	**ROM (byte)** **Lower**	**RAM (byte)** **Lower**	**Latency (Cycles / block)** **Lower**	**Energy (μJ / bit)** **Lower**	**Throughput at 4 MHz (Kbps)** **Higher**	**Efficiency (Kbps/KB)** **Higher**
128-bits key								

SPNs								
AES [12]	128	128	3716	2016	- / -	-	- / -	-
AES(D) [35]	128	128	15256	576	21781 / 22583	-	23.50 / 22.67	1.54
AES [35]	128	128	2444	232	19735 / -	-	25.94 / -	10.61
-//-	128	128	7360	184	2418 / -	-	211.74 / -	28.76
Zorro [12]	128	128	3024	1528	- / -	-	- / -	-
PRESENT [12]	128	64	3200	1320	- / -	-	- / -	-
PRESENT(D) [12]	128	64	3252	1384	- / -	-	- / -	-
PRESENT(D) [35]	80	64	7372	790	210571 / 399882	-	1.21 / 0.64	0.16
PRESENT [35]	80	64	1760	280	221471 / -	-	1.15 / -	0.65
-//-	80	64	3520	2308	16761 / -	-	15.27 / -	4.33
PRINCE(D) [35]	128	64	4660	392	112809 / 113561	-	2.26 / 2.25	0.48
PRINCE [35]	128	64	4076	224	14344 / -	-	17.84 / -	4.37
-//-	128	64	13344	328	11775 / -	-	21.74 / -	1.62
LED [12]	128	64	3876	1264	- / -	-	- / -	-
LED(D) [35]	80	64	3732	334	328387 / 364271	-	0.77 / 0.70	0.20
LED [35]	80	64	2212	192	41728 / -	-	6.13 / -	2.77
-//-	80	64	2164	368	35161 / -	-	7.28 / -	3.36
Klein [12]	96	64	2472	1256	- / -	-	- / -	-
Feistel								
Robin(D) [35]	128	128	3668	304	45926 / 46224	-	11.14 / 11.07	3.03
Robin [35]	128	128	2188	184	6187 / -	-	82.75 / -	37.81
-//-	28	128	2156	1180	5194 / -	-	98.57 / -	45.71
Fantomas(D) [35]	128	128	4604	308	35724 / 34318	-	14.33 / 14.91	3.11
Fantomas [35]	128	128	2184	184	4552 / -	-	112.47 / -	51.49
-//-	128	128	2088	1176	3524 / -	-	145.28 / -	69.57
CLEFIA [12]	128	128	4708	1256	- / -	-	- / -	-
CLEFIA(D) [12]	128	128	4880	1256	- / -	-	- / -	-
SIMON(D) [35]	96	64	892	400	13690 / 12860	-	18.69 / 19.90	20.95
SIMON [35]	96	64	600	104	1376 / -	-	186.04 / -	310.06
LBlock(D) [35]	80	64	2136	406	81324 / 83615	-	3.14 / 3.06	1.47
LBlock [35]	80	64	1192	148	10215 / -	-	25.06 / -	21.02
-//-	80	64	7664	280	7349 / -	-	34.83 / -	4.54
DES [12]	56	64	10628	4680	- / -	-	- / -	-
GFN								
Piccolo(D) [35]	80	64	1604	406	141579 / 150871	-	1.80 / 1.69	1.12
Piccolo [35]	80	64	940	160	18388 / -	-	13.92 / -	14.80
TWINE(D) [35]	80	64	2464	418	131198 / 130265	-	1.95 / 1.96	0.79
TWINE [35]	80	64	1180	140	20505 / -	-	12.48 / -	10.57
-//-	80	64	1180	156	15677 / -	-	16.32 / -	13.83
ARX								
LEA [61]	128	128	590	32	5231 / -	-	97.8 / -	165.76
HIGHT(D) [35]	128	64	2196	392	83157 / 91929	-	3.07 / 2.78	1.39
HIGHT [35]	128	64	1008	128	11602 / -	-	22.06 / -	21.88
-//-	128	64	6920	128	5836 / -	-	43.86 / -	6.33
SPECK(D) [35]	96	64	792	332	7665 / 12513	-	33.39 / 20.45	42.15
SPECK [35]	96	64	512	96	904 / -	-	283.18 / -	553.08
Hybrid								
HB [12]	128	16	3372	1256	- / -	-	- / -	-
PRESENT-GRP [12]	128	64	2980	1384	- / -	-	- / -	-

Table 10: Software implementations of block ciphers on 32-bit microcontrollers

References

1. F. Abed, E. List, S. Lucks, and J. Wenzel, Cryptanalysis of the SPECK family of block ciphers, IACR Cryptology ePrint Archive, 2013.
2. M. Agren, Some instant- and practical-time related-key attacks on KTANTAN32/48/64, 18^{th} International Conference on Selected Areas in Cryptography (SAC'11), Miri, A. and Vaudenay, S, (Eds.), Springer, 2011, pp. 213-229.
3. Z. Ahmaadian, M. Salmasizadeh, and M. R. Aref, Biclique cryptanalysis of the full-round KLEIN block cipher, IET Information Security, IET, 2015, pp. 8.
4. M. R. Albrecht, B. Driessen, E. B. Kavun, G. Leander, C. Paar, and T. Yalcin, Block Ciphers Focus On The Linear Layer (feat. PRIDE), Advances in Cryptology - CRYPTO, Springer, LNCS, 8616, 2014, pp.5776.
5. S. S. M. Aldabbagh, I. F. T. A. Shaikhli, and M. A. Alahmad, HISEC: A New Lightweight Block Cipher Algorithm, International Conference on Security of Information and Networks (SIN'14), Glasgow, Scotland, UK, 2014, pp. 151-157.
6. H.A. Alkhzaimi and M.M. Lauridsen, Cryptanalysis of the SIMON family of block ciphers, IACR Cryptology ePrint Archive, 2013.
7. T. Akishita and H. Hiwatari, Very compact hardware implementations of the blockcipher CLEFIA, Selected Areas in Cryptography (SAC'12), Springer, LNCS, 7118, 2012, pp.278292.
8. A. Anjali, Priyanka and S.K. Pal, A Survey of Cryptanalytic Attacks on Lightweight Block Ciphers, International Journal of Computer Science and Information & Security (IJCSITS), 2012, 2(2).
9. K. Aoki, T. Ichikawa, M. Kanda, M. Matsui, S. Moriai, J. Nakajima, and T. Tokita, Camellia: a 128-bit block cipher suitable for multiple platforms design and analysis, Selected Areas in Cryptography (SAC'01), Springer, LNCS, 2001, pp.3956.
10. J.-P. Aumasson, M. Naya-Plasencia, and M.-J. O. Saarinen, Practical attack on 8 rounds of the lightweight block cipher klein, Progress in Cryptology INDOCRYPT 2011, Springer, LNCS, 7107, 2011, pp. 134145.
11. S. A. Azimi, Z. Ahmadian, J. Mohajeri, and M. R. Aref, Impossible differential cryptanalysis of Piccolo lightweight block cipher, International ISC Conference on Information Security and Cryptology (ISCISC), Tehran, September, 2014, pp. 8994.
12. G. Bansod, N. Raval, and N. Pisharoty, Implementation of a New Lightweight Encryption Design for Embedded Security, IEEE Transactions on Information Forensics and Security, IEEE, vol. 10, issue 1, 2014, pp. 142-151.
13. L. Batina, A. Das, B. Ege, E. B. Kavun, N. Mentens, C. Paar, I. Verbauwhede, and T. Yalcin, Dietary recommendations for lightweight block ciphers power, energy and area analysis of recently developed architectures, IACR Cryptology ePrint Archive, 2013.
14. A. Bay, J.N. Jr, and S. Vaudenay, Cryptanalysis of reduced-round MIBS Block Cipher, Cryptology and Network Security (CANS), Springer, LNCS, 6467(5005), 2010, pp.119.
15. R. Beaulieu, S. Douglas, J. Smith, S. Treatman-Clark, B. Weeks, and L. Wingers, The SIMON and SPECK Families of Lightweight Block Ciphers, IACR Cryptology ePrint Archive, 2013, 404.
16. E. Biham, New types of cryptanalytic attacks using related keys, Journal of Cryptology, 7(4), 1994.
17. E. Biham, O. Dunkelman, and N. Keller, A Related-Key Rectangle Attack on the Full KASUMI, Advances in Cryptology ASIACRYPT 2005, Springer, LNCS, 3788, 2005, pp.443461.
18. C. Blondeau and B. Gerard, Differential Cryptanalysis of PUFFIN and PUFFIN2, Workshop on Lightweight Cryptography, ECRYPT, 2011.
19. C. Blondeau and K. Nyberg, Links between truncated differential and multidimensional linear properties of block ciphers and underlying attack complexities, EUROCRYPT 2014, Springer, LNCS, 8441, 2014, pp. 165-182.
20. A. Bogdanov, D. Khovratovich, and C. Rechbergerm, Biclique Cryptanalysis of the full AES, ASIACRYPT 2011, Springer, LNCS, 7073, 2011, pp.344371.
21. A. Bogdanov, L.R. Knudsen, G. Leander, C. Paar, and A. Poschmann, PRESENT: An Ultra-Lightweight Block Cipher, Cryptographic Hardware and Embedded Systems, CHES 2007, Springer, LNCS, 4727, 2007, pp.450466.

22. J. Borghoff et al., PRINCE A Low-latency Block Cipher for Pervasive Computing Applications, Advances in Cryptology ASIACRYPT 2012, Springer, LNCS, 7658, 2012, pp.208225.
23. J.W. Bos, D.A. Osvik, and D. Stefan, Fast Implementations of AES on Various Platforms, IACR Cryptology ePrint Archive, 2009.
24. M. akiroglu, Software implementation and performance comparison of popular block ciphers on 8-bit low-cost microcontroller, International Journal of the Physical Sciences, 5(9), 2010, pp.13381343.
25. C. DE Canniere, O. Dunkelman, and M. Knezevic, KATAN & KTANTAN - A Family of Small and Efficient Hardware-Oriented Block Ciphers, Cryptographic Hardware and Embedded Systems, CHES 2009, Springer, LNCS, 5747, 2009, pp.272288.
26. D. Canright, A very compact S-Box for AES, Cryptographic Hardware and Embedded Systems, CHES 2005, Springer, LNCS, 3659, 2005, pp.441455.
27. M., Cazorla, K. Marquet, and M. Minier, Survey and benchmark of lightweight block ciphers for wireless sensor networks, IACR Cryptology ePrint Archive, 2013, 295.
28. H. Cheng and H.M. Heys, Compact ASIC implementation of the ICEBERG block cipher with concurrent error detection, IEEE International Symposium on Circuits and Systems - ISCAS 2008, Seattle, Wash, 2008, pp. 29212924.
29. H. Cheng, H.M. Heys, and C. Wang, PUFFIN: A Novel Compact Block Cipher Targeted to Embedded Digital Systems, 11^{th} EUROMICRO Conference on Digital System Design Architectures - DSD 2008, Methods and Tools, Parma, Italy, 2008, pp. 383390.
30. N.T. Courtois, An Improved Differential Attack on Full GOST, IACR Cryptology ePrint Archive, 2012, 138.
31. T. De Cnudde, O. Reparaz, B. Bilgin, S. Nikova, V. Nikov, V. Rijmen, Masking AES with d+1 Shares in Hardware, Cryptographic Hardware and Embedded Systems (CHES 2016), Springer, LNCS, 9813, 2016, pp. 192-212.
32. J. Daemen, M. Peeters, G. Van Assche, and V. Rijmen, The NOEKEON Block Cipher, 2000, pp.1-30, http://gro.noekeon.org/ .
33. J. Daemen, M. Peeters, G. Van Assche, and V. Rijmen, On Noekeon, no!, 2001, http://gro.noekeon.org/ .
34. S. Das, Halka: a lightweight, software friendly block cipher using ultra-lightweight 8-bit S-box, IACR Cryptology ePrint Archive, 2014.
35. D. Dinu, Y. L. Corre, D. Khovratovich, L. Perrin, J. Grobshadl, and A. Biryukov, Triathlon of Lightweight Block Ciphers for the Internet of Things, IACR Cryptology ePrint Archive, 2015, 209.
36. T. Eisenbarth, S. Kumar, C. Paar, A. Poschmann, and L. Uhsadel, A Survey of Lightweight-Cryptography Implementations, IEEE Design & Test of Computers, 24(6), 2007, pp.522533.
37. T. Eisenbarth et al., Compact implementation and performance evaluation of block ciphers in ATtiny devices, Progress in Cryptology - AFRICACRYPT 2012, Springer, LNCS, 7374, 2012, pp.172187.
38. D. Engels, X. Fan, G. Gong, H. Hu, and E.M. Smith, Hummingbird: Ultra-Lightweight Cryptography for Resource-Constrained Devices, Financial Cryptography and Data Security - FC 2010, Springer, LNCS, 6054, 2010, pp.318.
39. D. Engels, M.O. Saarinen, P. Schweitzer, and E.M. Smith, The Hummingbird-2 Lightweight Authenticated Encryption Algorithm, RFID Security and Privacy, Springer, LNCS, 7055, 2011, pp.1931.
40. S. Engels, E. B. Kavun, H. Mihajloska, C. Paar, and T. Yalcin, A Non-Linear/Linear Instruction Set Extension for Lightweight Block Ciphers, 21^{st} IEEE Symposium on Computer Arithmetics (ARITH'21), IEEE Computer Society, Austin, TX, 2014, pp. 76-75.
41. EPCGLOBAL, EPC Tag Data Standard Version 1.5 EPCglobal Specification, 2010.
42. ETSI'S Security Algorithms Group Of Experts (SAGE), Specification of the 3GPP confidentiality and integrity algorithms, 2007, Document 2: Kasumi specification.
43. K. Fysarakis, G. Hatzivasilis, I. G. Askoxylakis, and C. Manifavas, RT-SPDM: Real-time Security, Privacy & Dependability Management of Heterogeneous Systems, Human Aspects of Information Security, Privacy and Trust (HCI International 2015), Springer, LNCS, 9190, 2015, pp. 619-630.

44. K. Fysarakis, G. Hatzivasilis, I. Papaefstathiou, and C. Manifavas, RtVMF – A secure Real-time Vehicle Management Framework with critical incident response, IEEE Pervasive Computing Magazine (PVC) – Special Issue on Smart Vehicle Spaces, IEEE, vol. 15, issue 1, 2016, pp. 22-30.
45. K. Fysarakis, G. Hatzivasilis, K. Rantos, A. Papanikolaou, and C. Manifavas, Embedded systems security challenges, Measurable security for Embedded Computing and Communication Systems – MeSeCCs 2014, 7-9 January, 2014, Lisbon, Portugal, pp. 1-10.
46. B. Gerard, V. Grosso, M. Naya-Plasencia, and F.-X. Standaert, Block Ciphers that are Easier to Mask: How Far Can we Go?, IACR Cryptology ePrint Archive, 2013.
47. D. Gligoroski, Edon-library of Reconfigurable Cryptographic Primitives Suitable for Embedded Systems, Workshop on Cryptographic Hardware and Embedded Systems, 2003.
48. Z. Gong, S. Nikova, and Y.W. Law, KLEIN: A New Family of Lightweight Block Ciphers, RFID Security and Privacy, Springer, LNCS, 7055, 2012, pp.118.
49. V. Grosso, G. Laurent, F.-X. Standaert, and K. Varici, LS-Designs: Bitslice Encryption for efficient Masked Software Implementations, Fast Software Encryption, FSE 2014, Springer, LNCS, 8540, 2014.
50. J. Guo, T. Peyrin, and A. Poschmann, The PHOTON family of lightweight hash functions, Advances in Cryptology CRYPTO 2011, Springer, LNCS, 6841, 2011, pp. 222239.
51. J. Guo, T. Peyrin, A. Poschmann, and M. J. B. Robshaw, The LED Block Cipher, Cryptographic Hardware and Embedded Systems, CHES 2011, Springer, LNCS, 6917, 2011, pp.326341.
52. X. Guo and P. Schaumont, The technology dependence of lightweight hash implementation cost, ECRYPT Workshop on Lightweight Cryptography (LC '11), 2011.
53. X. Guo, Secure and Efficient Implementations of Cryptographic Primitives, Virginia Polytechnic Institute and State University, 2012.
54. P. Hamalainen et al., Design and implementation of low-area and low-power AES encryption hardware core, Digital System Design: Architectures, Methods and Tools, 2006. DSD 2006, 9^{th} IEEE EUROMICRO Conference, 2006, pp.577583.
55. G. Hatzivasilis, G. Floros, I. Papaefstathiou, and C. Manifavas, Lightweight Authenticated Encryption for Embedded On-Chip Systems, Information Security Journal: A Global Perspective, Taylor & Francis, 2016, pp. 1-11.
56. G. Hatzivasilis, E. Gasparis, A. Theodoridis, and C. Manifavas, ULCL: an Ultra-Lightweight Cryptographic Library for Embedded Systems, Measurable Security for Embedded Computing and Communication Systems – MeSeCCs 2014, 7-9 Junuary, 2014, Lisbon, Portugal, pp. 11-18.
57. G. Hatzivasilis and C. Manifavas, Building trust in ad hoc distributed resource-sharing networks using reputation-based systems, 16^{th} Panhellenic Conference on Informatics (PCI 2012), IEEE, 5-7 October, 2012, Piraeus, Greece, pp. 416-421.
58. G. Hatzivasilis, I. Papaefstathiou, and C. Manifavas, ModConTR: a Modular and Configurable Trust and Reputation-based system for secure routing, 11^{th} ACS/IEEE International Conference on Computer Systems and Applications (AICCSA'2014), IEEE, Doha, Qatar, 10-13 November, 2014, pp. 56-63.
59. G. Hatzivasilis, I. Papaefstathiou, C. Manifavas, and I. Askoxylakis, Lightweight password hashing scheme for embedded systems, 9^{th} WG 11.2 International Conference on Information Security Theory and Practice (WISTP), IFIP, Springer, LNCS, 9311, 2015, pp. 249-259.
60. D. Hong et al., HIGHT: A New Block Cipher Suitable for Low-Resource Device. Cryptographic Hardware and Embedded Systems, CHES 2006, Springer, LNCS, 4249, 2006, pp.4659.
61. D. Hong, J.-K. Lee, D.-C. Kim, D. Kwon, K. H. Ryu, and D.-G. Lee, LEA: A 128-Bit Block Cipher for Fast Encryption on Common Processors, International Workshop on Information Security Applications (WISA 2013), Springer, LNCS, 8267, 2014, pp. 3-27.
62. J. Huand, S. Vaudenay, and X. Lai, On the Key Schedule of Lightweight Block Ciphers, Progress in Cryptology INDOCRYPT 2014, Springer, LNCS, 8885, 2014, pp. 124-142.
63. S. Indesteege, N. Keller, O. Dunkelman, E. Biham, and B. Preneel, A practical attack on Keeloq, Advances in Cryptology - EUROCRYPT 2008, Springer, LNCS, 4965, 2008, pp.118.

64. T. Isobe, A single-key attack on the full GOST block cipher, Fast Software Encryption, FSE 2011, Springer, LNCS, 6733, 2011, pp.290305.
65. P. Israsena and S. Wongnamkum, Hardware Implementation of a TEA-Based Lightweight Encryption for RFID Security, RFID Security, RFIDsec 2009, 3, 2009, pp.417433.
66. M. Izadi, B. Sadeghiyan, S.S. Sadeghian, and H.A. Khanooki, MIBS: a new lightweight block cipher, Cryptology and Network Security (CANS), Springer, LNCS, 5888, 2009, pp.334348.
67. J. Jacob, BEST-1: A Light Weight Block Cipher, IOSR Journal of Computer Engineering (IOSR-JCE), vol. 16, issue 2, ver. XII, March-April, 2014, pp. 91-95.
68. J. Jean, I. Nikoli, T. Peyrin, L. Wang, and S. Wu, Security Analysis of PRINCE. Fast Software Encryption, FSE 2013, Springer, LNCS, 8424, 2014, pp. 92-111.
69. K. Jeong, H. C. Kang, C. Lee, J. Sung, and S. Hong, Biclique cryptanalysis of lightweight block ciphers present, piccolo and led, IACR Cryptology ePrint Archive, 2012.
70. K. Jeong, C. Lee, and J. I. Lim, Improved differential fault analysis on lightweight block cipher LBlock for wireless sensor networks, EURASIP Journal on Wireless Communications and Networking (JWCN), EURASIP, 2013, 2013/1/151.
71. K. Jeong, Y. Lee, J. Sung, and S. Hong, Improved differential fault analysis on PRESENT-80/128, International Journal of Computer Mathematics, Taylor & Francis, vol. 90, issue 12, 2013, pp.25532563.
72. P. Junod, On the Complexity of Matsui's Attack, Selected Areas in Cryptography (SAC'01), Springer, LNCS, 2259, 2001, pp.199211.
73. J.-P. Kaps, Chai-tea, cryptographic hardware implementations of xtea, Progress in Cryptology INDOCRYPT 2008, Springer, LNCS, 5365, 2008, pp.363375.
74. F. Karakoc, H. Demirci, and A.E. Harmanci, ITUbee: A Software Oriented Lightweight Block Cipher, Lightweight Cryptography for Security and Privacy, Springer, LNCS, 8162, 2013, pp.1627.
75. J. Kelsey, B. Schneier, and D. Wagner, Related-key cryptanalysis of 3-WAY, Biham-DES, CAST, DES-X, newDES, RC2, and TEA, ICICS'97. Springer-Verlag, 1997, pp. 233246.
76. D. Khovratovich, G. Leurent, and C. Rechberger, Narrow-Bicliques: Cryptanalysis of Full IDEA, EUROCRYPT 2012, Springer, LNCS, 7237, 2012, pp.392410.
77. Y. Kim and H. Yoon, First Experimental Result of Power Analysis Attacks on a FPGA Implementation of LEA, IACR Cryptology ePrint Archive, 2014, 999.
78. P. Kitsos, N. Sklavos, M. Parousi, and A.N. Skodras, A comparative study of hardware architectures for lightweight block ciphers, Computers & Electrical Engineering, 38(1), 2012, pp.148160.
79. L.R. Knudsen and Raddum, H., On Noekeon, Public reports of the NESSIE project, 2001, Report: NES/DOC/UIB/WP3/009/1.
80. P. Kocher, J. Jaffe, and B. Jun, Differential power analysis, Advances in Cryptology, CRYPTO'99, Springer, 1999, pp.388397.
81. B. Koo, D. Hong, and D. Kwon, Related-key attack on the full HIGHT, Information Security and Cryptology, ICISC 2010, Springer, LNCS, 6829, 2011, pp.4967.
82. L. Knudsen, G. Leander, A. Poschmann, and M. J. B. Robshaw, PRINTcipher: a block cipher for IC-printing, Cryptographic Hardware and Embedded Systems, CHES 2010, Springer, LNCS, 6225, 2010, pp.1632.
83. M. Kumar, S. K. Pal, and A. Panigrahi, FeW: a lightweight block cipher, IACR Cryptology ePrint Archive, 2014.
84. X. Lai and J. L. Massey, A proposal for a new block encryption standard, Advances in Cryptology EUROCRYPT '90, Springer, LNCS, 473, 1991, pp.389404.
85. G. Leander, On linear hulls, statistical saturation attacks, PRESENT and a cryptanalysis of PUFFIN, EUROCRYPT 2011, Springer, LNCS, 6632, 2011, pp. 303-322.
86. G. Leander, B. Minaud, and S. Ronjom, A Generic Approach to Invariant Subspace Attacks: Cryptanalysis of Robin, iSCREAM and Zorro, EUROCRYPT 2015, IACR, Sofia, Bulgaria, 26-30 April, 2015.
87. G. Leander, C. Paar, A. Poschmann, and K. Schramm, New Lightweight DES Variants, Fast Software Encryption, FSE 2007, Springer, LNCS, 4593, 2007, pp.196210.

88. D. Lee, D.-C. Kim, D. Kwon, and H. Kim, Efficient Hardware Implementation of the Lightweight Block Encryption Algorithm LEA, Sensors Open Access Journal, MDPI, vol. 14, 2014, pp. 975-994.
89. Y. Lee, K. Jeong, C. Lee, J. Sung, and S. Hong, Related-key Cryptanalysis on the Full PRINTcipher Suitable for IC-Printing, International Journal of Distributed Sensor Networks, Hindawi, vol. 2014 (2014), article ID 389476, pages 10.
90. C.H. Lim, A Revised Version of CRYPTON: CRYPTON V1.0, Fast Software Encryption, FSE 1999, Springer, LNCS, 1636, 1999, pp.3145.
91. C.H. Lim and T. Korkishko, mCrypton - A lightweight block cipher for security of low-cost RFID tags and Sensors, Information Security Applications, Springer, LNCS, 3786, 2006, pp.243258.
92. Y.-I. Lim, J.-H. Lee, Y. You, and K.-R. Cho, Implementation of HIGHT cryptic circuit for RFID tag, IEICE Electronics Express, 6(4), 2009, pp.180186.
93. J. Lu, Related-key rectangle attack on 36 rounds of the XTEA block cipher, International Journal of Information Security, 8(1), 2008, pp.111.
94. F. Mace, F.-X. Standaert, and J. Quisquater, ASIC implementations of the block cipher sea for constrained applications, RFID Security (RFIDsec 2007), Malaga, Spain, 2007, pp. 103114.
95. C. Manifavas, G. Hatzivasilis, K. Fysarakis, I. Papaefstathiou, A survey of lightweight stream ciphers for embedded systems, Security and Communication Networks, Wiley, 21 December, 2015, issue 9, pp. 1226-1246.
96. C. Manifavas, G. Hatzivasilis, K. Fysarakis, and K. Rantos, Lightweight Cryptography for Embedded Systems a Comparative Analysis, 6^{th} International Workshop on Autonomous and Spontaneous Security SETOP 2012, Springer, LNCS, 8247, 2012, pp.333349.
97. M. Matsui, New Block Encryption Algorithm MISTY, Fast Software Encryption (FSE 1997) Springer, LNCS, 1267, 1997, pp. 54-68.
98. N. Mentens, J. Genoe, B. Preneel, and I. Verbauwhede, A low-cost implementation of Trivium, SASC, 2008, pp.197204.
99. A. Moradi, A. Poschmann, S. Ling, C. Paar, and H. Wang, Pushing the limits: a very compact and a threshold implementation of AES. Advances in Cryptology EUROCRYPT 2011, Springer, LNCS, 6632, 2011, pp. 6988.
100. S. Mukherjee and B. Sahoo, A Survey on Hardware Implementation of IDEA Cryptosystem, Information Security Journal: A Global Perspective, 20(4-5), 2011, pp.210218.
101. R. Needham and D. Wheeler, TEA extensions, Technical report, Computer Laboratory, University of Cambridge, October, 1997.
102. S. Nikova, V. Rijmen, and M. Schlaffer, Secure Hardware Implementation of Nonlinear Functions in the Presence of Glitches, Journal of Cryptology, 24(2), 2011, pp.292321.
103. S.K. Ojha, N. Kumar, K. Jain, and Sangeeta, TWIS A Lightweight Block Cipher, Information Systems Security, Springer, LNCS, 5905, 2009, pp.280291.
104. O. Ozen, K. Varici, C. Tezcan, and C. Kocair, Lightweight Block Ciphers Revisited: Cryptanalysis of Reduced Round PRESENT and HIGHT, Information Security and Privacy, Springer, LNCS, 5594, 2009, pp.90107.
105. C. Paar, A. Poschmann, and M. J. B. Robshaw, New designs in lightweight symmetric encryption, RFID Security, vol. 3, 2009, pp.349371.
106. J.H. Park, Security analysis of mCrypton proper to low-cost ubiquitous computing devices and applications, International Journal of Communication Systems, 22(8), 2009, pp.959969.
107. G. Piret, T. Roche, and C. Carlet, PICARO A Block Cipher Allowing Efficient Higher-Order Side-Channel Resistance, Applied Cryptography and Network Security, Springer, LNCS, 7341, 2012, pp.311328.
108. T. Plos, C. Dobraunig, M. Hofinger, A. Oprisnik, C. Wiesmeier, and J. Wiesmeier, Compact hardware implementation of the block ciphers mCrypton, NOEKEON, and SEA, Progress in Cryptology INDOCRYPT 2012, Springer, LNCS, 7668, 2012, pp.358377.
109. T. Plos, H. Grob, and M. Feldhofer, Implementation of Symmetric Algorithms on a Synthesizable 8-Bit Microcontroller Targeting Passive RFID Tags, Selected Areas in Cryptography (SAC'11), Springer, LNCS, 6544, 2011, pp.114129.

110. A. Poschmann, Lightweight Cryptography: Cryptographic Engineering for a Pervasive World, Ruhr-University Bochum, Germany, 2009.
111. A. Poschmann, S. Ling, and H. Wang, 256 Bit Standardized Crypto for 650 GE GOST Revisited, Cryptographic Hardware and Embedded Systems, CHES 2010, Springer, LNCS, 6225, 2010, pp.219233.
112. R. Rabbaninejad, Z. Ahmadian, M. Salmasizadeh, and M. R. Aref, Cube and dynamic cube attacks on SIMON32/64, International ISC Conference on Information Security and Cryptology (ISCISC), Tehran, 2014, pp. 98-103.
113. V.A. Reddy, A Cryptanalysis of the Tiny Encryption Algorithm. University of Alabama, 2003.
114. M. Renauld and F.-X. Standaert, Algebraic side-channel attacks, IACR Cryptology ePrint Archive, 2009, 279.
115. S. Rinne, T. Eisenbarth, and C. Paar, Performance Analysis of Contemporary Light-Weight Block Ciphers on 8-bit Microcontrollers, Software Performance Enhancement for Encryption and Decryption (SPEED 2007), Amsterdam, NL, 2007.
116. M. J. B. Robshaw, Searching for Compact Algorithms: CGEN, Progress in Cryptology - VIETCRYPT 2006, Springer, LNCS, 4341, 2006, pp.3749.
117. C. Rolfes, A. Poschmann, G. Leander, and C. Paar, Ultra-lightweight implementations for smart devicessecurity for 1000 gate equivalents, Smart Card Research and Advanced Applications, Springer, LNCS, 5189, 2008, pp.89103.
118. R. Roman, C. Alcaraz, and J. Lopez, A Survey of Cryptographic Primitives and Implementations for Hardware-Constrained Sensor Network Nodes, Mobile Networks and Applications, 12(4), 2007, pp.231244.
119. M-J. O. Saarinen, Cryptanalysis of Hummingbird-1, Fast Software Encryption (FSE 2011), Springer, LNCS, 6733, 2011, pp.328341.
120. M-J. O. Saarinen, Related-key Attacks Against Full Hummingbird-2, Fast Software Encryption (FSE 2014), Springer, LNCS, 8424, 2014, pp.467482.
121. S.E. Sarma, Towards the five-cent tag - MIT-AUTOID-WH-006, 2001.
122. A. Satoh and S. Morioka, Small and High-Speed Hardware Architectures for the 3GPP Standard Cipher KASUMI, International Conference on Information Security (ISC 2002), Springer, LNCS, 2433, 2002, pp.48-62.
123. A. Satoh and S. Morioka, Hardware-focused performance comparison for the standard block ciphers AES, Camellia, and Triple-DES, Information Security, Springer, LNCS, 2851, 2003, pp.252266.
124. K. Shibutani, T. Isobe, H. Hiwatari, A. Mitsuda, T. Akishita, and T. Shirai, Piccolo: An Ultra-Lightweight Blockcipher, Cryptographic Hardware and Embedded Systems (CHES 2011), Springer, LNCS, 6917, 2011, pp.342357.
125. T. Shirai, K. Shibutani, T. Akishita, S. Moriai, and T. Iwata, The 128-bit blockcipher CLEFIA (extended abstract), Fast Software Encryption (FSE 2007), Springer, LNCS, 4593, 2007, pp. 181-195.
126. H. Soleimany, Self-similarity cryptanalysis of the block cipher ITUbee, IET Information Security, IET, 2014, pp. 6.
127. H. Soleimany et al., Reflection cryptanalysis of PRINCE-like ciphers, Journal of Cryptology, Springer, 13 December, 2013.
128. J. Song, K. Lee, and H. Lee, Biclique cryptanalysis on lightweight block cipher: HIGHT and Piccolo, International Journal of Computer Mathematics, Taylor & Francis, vol. 90, issue 12, 2013, pp.25642580.
129. F.-X. Standaert, G. Piret, N. Gershenfeld, and J. Quisquater, SEA: A Scalable Encryption Algorithm for small embedded applications, Smart Card Research and Advanced Applications, Springer, LNCS, 3928, 2006, pp.222236.
130. F.-X. Standaert, G. Piret, G. Rouvroy, J. Quisquater, and J.-D. Legat, ICEBERG: An Involutional Cipher Efficient for Block Encryption in Reconfigurable Hardware, Fast Software Encryption (FSE 2004), Springer, LNCS, 3017, 2004, pp.279298.
131. Standard, NIST FIPS, Data Encryption Standard (DES). Federal Information Processing Standards Publication, 1999, 46-3.
132. Standard, NIST FIPS, Advanced Encryption Standard (AES). Federal Information Processing Standards Publication, 2001, 197.
133. B. Su, W. Wu, L. Zhang, and Y. Li, Full-round differential attack on TWIS block cipher, Information Security Applications, Springer, LNCS, 6513, 2010, pp.234242.

134. Y. Sun, M. Wang, S. Jiang, and Q. Sun, Differential Cryptanalysis of Reduced-Round ICEBERG, AFRICACRYPT 2012, Springer, LNCS, 7374, 2012, pp. 155-171.
135. T. Suzaki, K. Minematsu, S. Morioka, and E. Kobayashi, Twine: A lightweight, versatile block cipher, ECRYPT Workshop on Lightweight Cryptography (LC11), 2011, pp. 146169.
136. C. Texcan, The Improbable Differential Attack: Cryptanalysis of Reduced Round CLEFIA, INDOCRYPT 2010, Springer, LNCS, 6498, 2010, pp. 197209.
137. O. Tigli, Area efficient ASIC implementation of IDEA (International Data Encryption Standard), Best design for ASIC implementation of IDEA, GMU 2003.
138. TOSHIBA, Toshiba CMOS Technology Roadmap for ASIC, 2015. http://www.toshiba-components.com/ASIC/Technology.html .
139. H. Tupsamudre, S. Bisht, and D. Mukhopadhyay, Differential Fault Analysis on the families of SIMON and SPECK ciphers, IACR Cryptology ePrint Archive, 2014.
140. M. Ullrich, C. D. Canniere, S. Indesteege, O. Kucuk, N. Mouha, and B. Preneel, Finding Optimal Bitsliced Implementations of 4 × 4-bit S-boxes, Symmetric Key Encryption Workshop (SKEW), Copenhagen, DK, 2011.
141. M. Walter, S. Bulygin, and J. Buchmann, Optimizing Guessing Strategies for Algebraic Cryptanalysis with Applications to EPCBC, Information Security and Cryptology, Springer, LNCS, 7763, 2013, pp. 175-197.
142. C. Wang and H.M. Heys, An ultra compact block cipher for serialized architecture implementations, Canadian Conference on Electrical and Computer Engineering (CCECE '09), St. John's, Newfoundland, IEEE, 2009, pp. 10851090.
143. Y. Wang, W. Wu, X. Yu, and L. Zhang, Security on lblock against biclique cryptanalysis, Information Security Applications (WISA 2012), Springer, LNCS, 7690, 2012, pp.114.
144. L. Wen, M. Wang, A. Bogdanov, and H. Chen, Multidimensional zero-correlation attacks on lightweight block cipher HIGHT: Improved cryptanalysis of an ISO standard, Information Processing Letters, Elsevier, 114, 2014, pp.322330.
145. S. Weis, Security and privacy in radio-frequency identification devices. Faculty of the Massachusetts Institute of Technology (M.I.T.), 2003.
146. D. Wheeler and R. Needham, TEA, a Tiny Encryption Algorithm, Fast Software Encryption (FSE 1994), Springer, LNCS, 1008, 1994, pp.363366.
147. D. Wheeler and R. Needham, Correction to XTEA, Technical report, Computer Laboratory, University of Cambridge, October, 1998.
148. W. Wu and L. Zhang, LBlock: a lightweight block cipher, Applied Cryptography and Network Security, Springer, LNCS, 6715, 2011, pp.327344.
149. L. Yang, M. Wang, and S. Qiao, Side channel cube attack on PRESENT, Cryptology and Network Security (CANS), Springer, LNCS, 5888, 2009, pp.379391.
150. H. Yap, K. Khoo, A. Poschmann, and M. Henricksen, EPCBC-A Block Cipher Suitable for Electronic Product Code Encryption, Cryptology and Network Security (CANS), Springer, LNCS, 7092, 2011, pp.7697.
151. E. Yarrkov, Cryptanalysis of XXTEA, IACR Cryptology ePrint Archive, 2010, 254.
152. H. Yoshikawa, M. Kaminaga, A. Shikoda, and T. Suzuki, Secret key reconstruction method using round addition DFA on lightweight block cipher LBlock, International Symposium on Information Theory and its Applications (ISITA), Melbourne, VIC, 2014, pp. 493-496.
153. Y. Yu, Y. Yang, Y. Fan, and H. Min, Security scheme for RFID tags, Auto-ID Labs, Fudan University, White paper, 2006.
154. M. R. Z'aba, N. Jamil, M. E. Rusli, M. Z. Jamaludinm, and A. A. M. Yasir, I-PRESENT^{TM}: An Involutive Lightweight Block Cipher, Journal of Information Security, Scientific Research, vol. 5, 2014, pp. 114-122.
155. W. Zhang, Z. Bao, D. Lin, V. Rijmen, B. Yang, and I. Verbauwhede, RECTANGLE: a bit-slice ultra-lightweight block cipher suitable for multiple platforms, IACR Cryptology ePrint Archive, 2014.
156. G. Zhao, R. Li, L. Cheng, C. Li, and B. Sun, Differential fault analysis on LED using Super-Sbox, IET Information Security, IET, 2014, pp. 10.
157. G. Zhao, B. Sun, C. Li, and J. Su, Truncated differential cryptanalysis of PRINCE, Security and Communication Networks, Wiley, 2015.

158. X. Zhao, T. Wang, and Y. Zheng, Cache Timing Attacks on Camellia Block Cipher, IACR Cryptology ePrint Archive, 2009.
159. B. Zhu and G. Gong, Multidimensional Meet-in-the-Middle Attack and Its Applications to KATAN32/48/64, Cryptography and Communications, Springer, vol. 6, issue 4, 2014, pp. 313-333.