New Light-Weight Crypto Algorithms for RFID

Axel Poschmann, Gregor Leander, Kai Schramm, and Christof Paar
Horst Görtz Institute for IT-Security, Ruhr-University Bochum, Germany
Email: {poschmann, schramm, cpaar}@crypto.rub.de, leander@itsc.rub.de

***Abstract*— We propose a new block cipher, DESL (DES Lightweight extension), which is strong, compact and efficient[1]. Due to its low area constraints DESL is especially suited for RFID (Radio Frequency Identification) devices. DESL is based on the classical DES (Data Encryption Standard) design, however, unlike DES it uses a single S-box repeated eight times. This approach makes it possible to considerably decrease chip size requirements. The S-box has been highly optimized in such a way that DESL resists common attacks, i.e., linear and differential cryptanalysis, and the Davies-Murphy-attack. Therefore DESL achieves a security level which is appropriate for many applications. Furthermore, we propose a light-weight implementation of DESL which requires 45% less chip size and 86% less clock cycles than the best AES implementations with regard to RFID applications. Compared to the smallest DES implementation published, our DESL design requires 38% less transistors. Our 0.18μm DESL implementation requires a chip size of 7392 transistors (1848 gate equivalences) and is capable to encrypt a 64-bit plaintext in 144 clock cycles. When clocked at 100 kHz, it draws an average current of only 0.89μA. These hardware figures are in the range of the best eSTREAM streamcipher candidates, comprising DESL as a new alternative for ultra low-cost encryption.**

I. Introduction

RFID (Radio Frequency Identification) tags have numerous applications, ranging from supply chain optimization to ehealth. Basically, RFID tags consist of a transponder and an antenna and are able to remotely send and receive data from an RFID host or reader device. In general, RFID tags can be divided into passive and active devices: active tags provide their own power supply (i.e. in form of a battery), whereas passive tags solely rely on the energy of the carrier signal transmitted by the reader device. Passive RFID devices are not only much less expensive, but also require less chip size and have a longer life cycle [1].

Our proposed DESL algorithm and its low-power, size-optimized implementation aims at very constrained devices such as passive RFID tags. Providing cryptographic primitives (esp. encryption) at extremely low cost is of paramount importance for securing RFID applications. Thus far, there have been two approaches for providing cryptographic primitives for such situations:

- Optimized low-cost implementations for standardized and trusted algorithms, which means in practice in essence block ciphers such as AES [2], see e.g., [3].
- Design new ciphers with the goal of having low hardware implementation costs[2]

The best known light-weight AES implementation [3] requires 3400 gates and draws a maximum current of $3.0\mu A$ @ $100kHz$. This AES design is based on a byte-per-byte serialization, which only requires the implementation of a single S-box [4] and achieves one encryption within 1032 clock cycles ($= 10.32ms$ @ $100kHz$). Unfortunately, the ISO/IEC 18000 standard requires that the latency of a response of an RFID tag does not exceed $320\mu s$, which is why [3] proposes a slightly modified challenge-response protocol based on interleaving. The problem with this approach is that AES like most modern block ciphers was primarily designed with good software implementation properties in mind, and not necessarily with hardware-friendly properties. The only known cipher that was designed with a strong focus on low hardware costs is the Data Encryption Standard, DES [5]. If we compare a standard, one-round implementation of AES and DES, the latter consumes about 6% (!) of the logic resources of AES, while having a shorter critical path [6], [7]. Therefore, we decided to follow the second approach and modify the hardware efficient and very well investigated cipher DES.

The main design ideas of the new cipher family, which are either original DES efficiently implemented or a variant of DES, are:

1) Use of a serial hardware architecture which reduces the gate complexity.
2) Optionally apply key-whitening in order to render brute-force attacks impossible.
3) Optionally replace the 8 original S-Boxes by a single one which further reduces the gate complexity.

If we make use of the first idea, we obtain a light-weight implementation of the original DES algorithm which consumes about 35% less gates than the best known AES implementation [3]. To our knowledge, this is the smallest reported DES implementation, trading area for throughput. The implementation requires also about 86% fewer clock cycles for encrypting of one block than the serialized AES implementation in [3] (1032 cycles vs. 144) which makes it easier to use in standardized RFID protocols. However, the security provided is limited by the 56 bit key. Brute forcing this key space takes a few months and hundreds of PCs in software, and only a few days with a special-purpose machine such as COPACOBANA [8]. Hence, this implementation is only relevant for application where short-term security is needed, or where the values protected are relatively low. However, we can imagine that in certain low cost applications such a security level is adequate.

In situation where a higher security level is needed key whitening, which we define here as follows:

[1]Part of this work has already been presented at RFIDSec '06, a non-proceeding workshop.

[2]see, e.g., the profile 2 algorithms of the eStream project at http://www.ecrypt.eu.org/stream/

$DESX_{k.k1.k2}(x) = k2 \oplus DES_k(k1 \oplus x)$, can be added to standard DES, yielding DESX. The bank of XOR gates increase the gate count by about 14%[3]. The best known key search attack uses a time-memory trade-off and requires 2^{120} time steps and 2^{64} memory locations, which renders this attack entirely out of reach. The best known mathematical attack is linear cryptanalysis [9]. LC requires about 2^{43} chosen ciphertext blocks together with the corresponding plainttexts. At a clock speed of 500 kHz, our DESX implementation will take more than 80 years, so that analytical attacks do not pose a realistic threat. Please note that parallelization is only an option if devices with identical keys are available.

In situations where extremely light-weight cryptography is needed, we can further decrease the gate complexity of DES by replacing the eight original S-Boxes by a single new one. This light-weight variant of DES is named DESL and has a brute-force resistance of 2^{56}. In order to strengthen the cipher, key whitening can be applied yielding the ciphers DESXL. The crucial question is what the strength of DESL and DESXL is with respect to analytical attacks. We are fully aware that any changes to a cipher might open the door to new attacks, even if the changes have been done very carefully and checked against known attacks. Hence, we believe that DESL (or DESXL) should primarily not be viewed as competitors to AES, but should be used in applications where established algorithms are too costly. In such applications which have to trade security (really: trust in an algorithm) for cost, we argue that it is a cryptographically sounder approach to modestly modify a well studied cipher (in fact, the world's best studied crypto algorithm), rather than designing a new algorithm altogether.

II. Lightweight Implementation of DES and DESL

In this section we present a size-optimized design of DES, which is smaller than any previous implementations of DES to our knowledge.

A. Design Considerations and Implementation of DES

The overall architecture of our size-optimized DES implementation is depicted in Figure 1.

Our design basically consists of five core modules: *mem_left*, *mem_right*, *keyschedule*, *controller*, and *sbox*. The *controller* module manages all control signals in the ASIC based on the finite state machine depicted in Figure 3. In the *keyschedule* module all DES round keys are generated. It is composed of a 56-bit register, an input multiplexor, and an output multiplexor to select the right fraction of the roundkey. The *mem_left* module consists of eight 4-bit wide registers, each composed of D-flip-flops[4]. The *mem_right* module is similar to the *mem_left* module with slight differences. It also consists of eight 4-bit wide registers, but it has different input and

[3]This number only includes additonal XOR gates, because we assume that all keys have to be stored at different memory locations anyway.

[4]Note that the memory modules were designed in a shift register manner, such that the output of a 4-bit block is fed as the new input into the following block. At the end of the chain the current 4-bit block is provided and can be processed without an additional output multiplexor, which results in a saving of 192 transistors (48 GE).

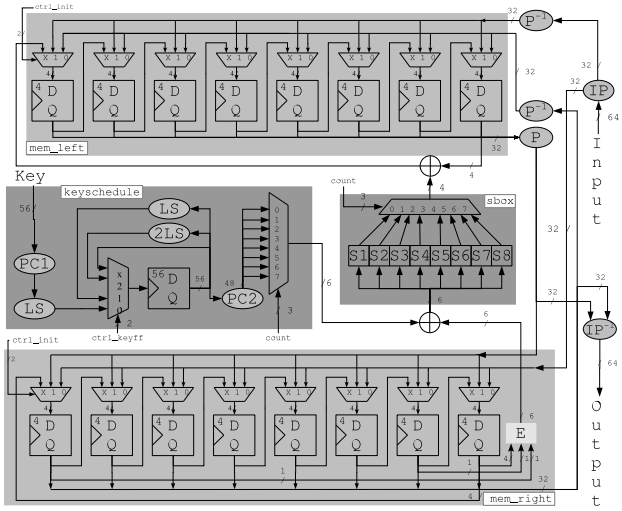

Fig. 1. Datapath of the serialized DES ASIC

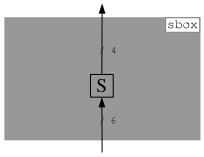

Fig. 2. *sbox* module of the DESL algorithm

output signals: instead of a 4-bit wide output it has a 6-bit wide output, due to the *expansion* function of DES[5]. The *sbox* module consists of eight S-boxes of the DES algorithm and an output multiplexor. The S-boxes are realized in combinatorial logic, i.e. a sum of products (SOP) [10]. Furthermore, we investigated whether there exists a correlation between the design criteria (i.e. linear and differential characteristics) and the required number of logic gates (AND, OR, NOT) for each S-box. However, we did not observe a significant deviation for any S-box.

Figure 1 shows the datapath of our serialized DES design. The 56-bit *key* is stored in the key flip-flop register after the PC1 and LS1 permutations have been applied. The plaintext is first confused using the *Initial Permutation* (IP), then, it is split into two 32-bit inputs for the modules *mem_left* and *mem_right*, respectively. The input of *mem_left* is modified by the inverse of the P permutation and stored in the registers

[5]Note that the design in a shift register manner in this module saves even more transistors (288, 72 GE) than in the *mem_left* module, because here a 6-bit wide output multiplexor can be saved. Altogether 480 transistors (120 GE) can be saved by our memory design compared to a regular design.

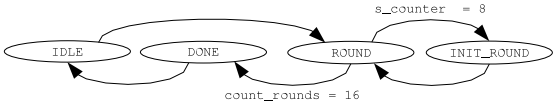

Fig. 3. Finite State Machine of the DES ASIC

of the modules *mem_left* and *mem_right* in one cycle. Next, the output of the last register in *mem_right* is both stored in the first register of *mem_right* and expanded to six bits. After an XOR operation with the appropriate block of the current round key, this expanded value is processed by the *sbox* module, which is selected by the *count* signal, provided by the *controller* module. Finally, the result is XORed with the output of the *mem_left* module, and stored in the first flip-flop of the *mem_left* module. This is repeated eight times, until all 32 bit of the right half are processed.

In our design, we applied the P permutation in each ninth clock cycle. Because the P^{-1} permutation is applied before the left 32-bit half L_i is stored in the *mem_left* module, we perform the (P) permutation on the resulting right half R_{i+1}: $R_{i+1} = P\left(P^{-1}\left(L_i\right) \oplus S\left(E\left(R_i\right) \oplus K_i\right)\right)$, where L_i denotes the left half, R_i denotes the right half, and K_i denotes the round key.

By reducing the datapath from a 32-bit bus to a 4-bit bus, only $6 * 10 + 4 * 10 = 100$ transistors (25 GE) are needed for the XOR operations, compared to $48 * 10 + 32 * 10 = 800$ (200 GE) transistors in a not-serialized design. This saving comes with the disadvantage of two additional multiplexors, each one for the round key (288 transistors, 72 GE) and for the S-box output (192 transistors, 48 GE). As we will see in Section II-C, our DESL algorithm does not need an output multiplexor in the *sbox* module.

Once all eight 4-bit blocks of both halves have been processed, they are concatenated to two 32-bit wide outputs of the modules *mem_left* and *mem_right*. The output of the module *mem_left* is transformed by the P permutation and stored as the new content of the *mem_right* module, while the output of the *mem_right* module is stored as the new content of the *mem_left* module.

This execution flow repeats another 15 rounds. Finally, both outputs of the memory modules *mem_left* and *mem_right* are concatenated to a 64-bit wide output. This output is confused by the *Inverse Initial Permutation* (IP^{-1}), which results in a valid ciphertext of the DES algorithm.

We used Synopsys Design Vision V-2004.06-SP2 to map our DES design to the Artisan UMC 0.18μm L180 Process 1.8-Volt Sage-X Standard Cell Library and Cadence Silicon Ensemble 5.4 for the Placement & Routing-step.

It takes 144 clock cycles to encrypt one 64-bit block of plaintext. For one encryption at 100 kHz the average current consumption is 1.19 μA and the throughput reaches 5.55 KB/s.

B. Further Optimization Considerations

In our DES ASIC design registers take up the main part of chip size (33.78%), followed by the S-boxes (32.11%), and multiplexors (31.19%). Chip size of registers and multiplexors can not be minimized any further, hence we thought about further possibilities to optimize the chip size of the S-boxes.

While it does not seem to be possible to find better logic minimizations of the original DES S-boxes, there have been other approaches to alter the S-boxes, e.g. key-dependent S-boxes [11], [12] or the so-called s^i*DES* [13]–[15]. While all these approaches, despite the fact that some of them have worse cryptographic properties than DES [16], just change the content and not the **number** of S-boxes. To the best of our knowledge, no DES variant has been proposed in the past which uses a single S-box, repeated eight times.

S															
14	5	7	2	11	8	1	15	0	10	9	4	6	13	12	3
5	0	8	15	14	3	2	12	11	7	6	9	13	4	1	10
4	9	2	14	8	7	13	0	10	12	15	1	5	11	3	6
9	6	15	5	3	8	4	11	7	1	12	2	0	14	10	13

TABLE I
IMPROVED DESL S-BOX

Our work regarding a strengthened S-box is based on the original design criteria for DES as published by Coppersmith [17] and the work of Kim et al. [13]–[15] where several criteria for DES type S-boxes are presented to strengthen the resistance against differential and linear cryptanalysis, and the Davies-Murphy-Attack [18].

The mathematical background of this part would go beyond the scope of this special session. Therefore we briefly list our conditions below:

1: If two inputs to an S-box differ in their first bit and are identical in their last two bits, the two outputs must not be the same.

2: $|S_b^{\mathcal{W}}(a)| \leq 28$ for all $a \in \mathrm{GF}(2)^6,\ b \in \mathrm{GF}(2)^4$.

3: $S_b^{\mathcal{W}}(a) \leq 4$ for all $a \in \mathrm{GF}(2)^6, b \in \mathrm{GF}(2)^4$ with $\mathrm{wt}(a) = \mathrm{wt}(b) = 1$.

4: $S_b^{\mathcal{W}}(a) \leq 16$ for all $a \in \mathrm{GF}(2)^6, b \in \mathrm{GF}(2)^4$ with $\mathrm{wt}(a), \mathrm{wt}(b) \leq 2$.

5: $|S_{b_1}^{\mathcal{W}}(a) S_{b_2}^{\mathcal{W}}(a)| \leq 240$ for all $a \in \mathrm{GF}(2)^6, b_1, b_2 \in \mathrm{GF}(2)^4$ with $\mathrm{wt}(a) = 1, \mathrm{wt}(b_1 + b_2) = 1$.

6: $S_b^{\mathcal{W}}(a) = 0$ for $a \in \{(010000), (000010)\}, b \in \mathrm{GF}(2)^4$ with $\mathrm{wt}(b) = 1$.

7: $|S_{b_1}^{\mathcal{W}}(000010) S_{b_2}^{\mathcal{W}}(000010)| = 0$ for all $b_1, b_2 \in \mathrm{GF}(2)^4$ with $\mathrm{wt}(b_1 + b_2) = 1$.

8: $S_{(0100)}^{\mathcal{W}}(000100) = 0$

$S_b^{\mathcal{W}}(a)$ denotes the *Walsh-coefficient* and wt(x) denotes the *Hamming weight* of x. We randomly generated S-boxes, which fulfill the original DES criteria (S-1), (S-3), (S-4), (S-5), (S-7) (see [17]), and the above listed conditions 1 to 8. It can be shown, that S-boxes that fullfill these criteria are immune to linear and differential cryptanalysis, and the Davies-Murphy-Attack. Table I shows the S-box, which is used in DESL.

C. Implementation of DESL

The main difference between DESL and DES lies in the f-function. We substituted the eight original DES S-boxes by a single but cryptographically stronger S-box, which is repeated eight times. Furthermore, we omitted the initial permutation (IP) and its inverse (IP^{-1}), because they do not provide additional cryptographic strength, but at the same time require area for wiring. The design of our DESL algorithm is exactly the same as for the DES algorithm, except for the (IP) and (IP^{-1}) wiring and the *sbox* module. The changed *sbox* module

	gate equiv. total	gate equiv. rel.	cycles / block	μA at 100 kHz	Process μm
DESL	**1848**	**1**	**144**	**0.89**	0.18
DES	2309	1.25	144	1.19	0.18
DESX	2629	1.42	144	–	0.18
DESXL	2168	1.17	144	–	0.18
AES-128 [3]	3400	1.84	1032	3.0	0.35
Trivium [20]	2599	1.41	–	–	0.13
Grain-80 [20]	1294	0.70	–	–	0.13
HIGHT [21]	3048	1.65	34	–	0.25

TABLE II

COMPARISON OF EFFICIENT CIPHERS BASED ON GATE COUNT, CLOCK CYCLES, AND CURRENT CONSUMPTION

implements only one S-box. As one can see in Figure 2, this module neither needs the *count* control signal nor an output multiplexor, which saves another 192 transistors (48 GE). It takes 144 clock cycles to encrypt one 64-bit block of plaintext. For one encryption at 100 kHz the average current consumption is 0.89 μA and the throughput reaches 5.55 KB/s. All results are summarized in Table II.

III. RESULTS AND CONCLUSION

We presented our implementation results of DES and DESL in Section II. Table II shows, that our DESL cipher needs 20% less transistors compared with our DES implementation and 38% less transistors compared with the implementation in [6]. This table also shows, that DESL uses 25% less average current than DES. In comparison with the AES design in [3], our design needs 45% less gate equivalents and 86% less clock cycles as shown in Table II. Note that the AES design in [3] was implemented in a $0.35\mu m$ standard cell technology, whereas our design was implemented in a $0.18\mu m$ standard cell technology. Therefore a fair comparison is only possible with regard to the gate equivalences. Regarding area consumption, our DESL is competitive even to stream ciphers recently proposed within the eSTREAM project [19]. More interesting, DESL would be the second smallest stream cipher in terms of gate count compared to all eSTREAM candidates (see Table II). Due to the low current consumption and the small chip size required for our DESL design, it is especially suited for resource limited applications, for example RFID tags and wireless sensor nodes.

Finally, we can conclude, that DESL is more secure against linear and differential cryptanalysis and the Davies-Murphy attack, more size-optimized, and more power efficient than DES, which makes it especially suited for RFID applications. Furthermore, DESL is worth to be considered as an alternative for stream ciphers.

IV. ACKNOWLEDGMENTS

The work presented in this paper was supported in part by the European Commission within the STREP UbiSec&Sens of the EU Framework Programme 6 for Research and Development (`www.ist-ubisecsens.org`). The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the UbiSec&Sens project or the European Commission.

REFERENCES

[1] K. Finkenzeller, *RFID-Handbook: Fundamentals and Applications in Contactless Smart Cards and Identification*. John Wiley and Sons, 2003.

[2] "Advanced Encryption Standard (AES)," National Institute of Standards and Technology (NIST), Federal Information Processing Standards (FIPS) Publication 197, November 2001.

[3] M. Feldhofer, J. Wolkerstorfer, and V. Rijmen, "AES Implementation on a Grain of Sand," *Information Security, IEE Proceedings*, vol. 152, no. 1, pp. 13–20, 2005.

[4] J. Daemen and V. Rijmen, *The Design of Rijndael*. Springer Verlag, Berlin, 2002.

[5] "Data Encryption Standard (DES)," National Institute of Standards and Technology (NIST), Federal Information Processing Standards (FIPS) Publication 46-3, October 1999.

[6] I. Verbauwhede, F. Hoornaert, J. Vandewalle, and H. D. Man, "Security and Performance Optimization of a New DES Data Encryption Chip," *IEEE Journal of Solid-State Circuits*, vol. 23, no. 3, pp. 647–656, 1988.

[7] K. T. Akashi Satoh, Sumio Morioka and S. Munetoh, "A Compact Rijndael Hardware Architecture with S-Box Optimization," in *ASIACRYPT 2001*, ser. LNCS, vol. 2248. Springer-Verlag, 2001, pp. 239 – 254.

[8] S. Kumar, C. Paar, J. Pelzl, G. Pfeiffer, and M. Schimmler, "Breaking Ciphers with COPACOBANA - A Cost-Optimized Parallel Code Breaker," in *Workshop on Cryptographic Hardware and Embedded Systems — CHES 2006,Yokohama, Japan*. Springer Verlag, 2006.

[9] M. Matsui, "Linear Cryptanalysis of DES Cipher," in *Advances in Cryptology — EUROCRYPT '93*, T. Hellenseth, Ed., vol. LNCS 0765. Berlin, Germany: Springer-Verlag, 1994, pp. 286 – 397.

[10] "espresso," available for download at http://embedded.eecs.berkeley.edu/pubs/downloads/espresso/index.htm.

[11] Biham and Biryukov, "How to Strengthen DES Using Existing Hardware," in *ASIACRYPT: Advances in Cryptology – ASIACRYPT: International Conference on the Theory and Application of Cryptology*. LNCS, Springer-Verlag, 1994, available for download at citeseer.ist.psu.edu/biham94how.html.

[12] E. Biham and A. Shamir, "Differential Cryptanalysis of the Full 16-Round DES," in *CRYPTO '92*, 1992, pp. 487–496, available for download at citeseer.ist.psu.edu/biham93differential.html.

[13] K. Kim, S. Lee, S. Park, and D. Lee, "DES Can Be Immune to Linear Cryptanalysis," in *Proceedings of the Workshop on Selected Areas in Cryptography SAC'94*, May 1994, pp. 70–81, available for download at citeseer.csail.mit.edu/kim94des.html.

[14] ——, "Securing DES S-boxes Against Three Robust Cryptanalysis," in *Proceedings of the Workshop on Selected Areas in Cryptography SAC'95*, 1995, pp. 145–157, "available for download at citeseer.ist.psu.edu/kim95securing.html".

[15] K. Kim, S. Park, and S. Lee, "Reconstruction of s^2-DES S-Boxes and their Immunity to Differential Cryptanalysis," in *Proceedings of 1993 Korea-Japan Joint Workshop on Information Security and Cryptology (JW-ISC'93)*, October 1993, available for download at citeseer.csail.mit.edu/kim93reconstruction.html.

[16] L. R. Knudsen, "Iterative Characteristics of DES and s^2-DES," *Advances in Cryptology: Proceedings of CRYPTO '92*, pp. 497–511, 1992.

[17] D. Coppersmith, "The Data Encryption Standard (DES) and its Strength Against Attacks," IBM Journal of Research and Development, IBM Thomas J. Watson Research Center, Technical Report RC 186131994, December 1994.

[18] D. Davies and S. Murphy, "Pairs and Triplets of DES S-Boxes," *Journal of Cryptology*, vol. 8, no. 1, pp. 1–25, 1995. [Online]. Available: citeseer.ist.psu.edu/davies93pairs.html

[19] F. K. Gurkaynak, "Hardware Evaluation of eSTREAM Candidate Algorithms," available for download at http://asic.ethz.ch/estream/E.png.

[20] T. Good and M. Benaissa, "Hardware Results for selected Stream Cipher Candidates," State of the Art of Stream Ciphers 2007 (SASC 2007), Workshop Record, pp. 191 – 204, February 2007.

[21] D. Hong, J. Sung, S. Hong, J. Lim, S. Lee, B.-S. Koo, C. Lee, D. Chang, J. Lee, K. Jeong, H. Kim, J. Kim, and S. Chee, "HIGHT: A New Block Cipher Suitable for Low-Resource Device," in *Cryptographic Hardware and Embedded Systems — CHES 2006*, ser. Lecture Notes in Computer Science, L. Goubin and M. Matsui, Eds., vol. 4249. Springer, 2006, pp. 46–59.