LS-Designs: Bitslice Encryption for Efficient Masked Software Implementations

Vincent Grosso[1], Gaëtan Leurent[1,2], François-Xavier Standaert[1], Kerem Varici[1]

[1] ICTEAM/ELEN/Crypto Group, Université catholique de Louvain, Belgium.
[2] Inria, EPI SECRET, Rocquencourt, France.

Abstract. Side-channel analysis is an important issue for the security of embedded cryptographic devices, and masking is one of the most investigated solutions to mitigate such attacks. In this context, efficient masking has recently been considered as a possible criteria for new block cipher designs. Previous proposals in this direction were applicable to different types of masking schemes (e.g. Boolean and polynomial). In this paper, we study possible optimizations when specializing the designs to Boolean masking. For this purpose, we first observe that bitslice ciphers have interesting properties for improving both the efficiency and the regularity of masked software implementations. Next we specify a family of block ciphers (denoted as LS-designs) that can systematically take advantage of bitslicing in a principled manner. Eventually, we evaluate both the security and performance of such designs and two of their instances, confirming excellent properties for physically secure applications.

1 Introduction

Lightweight cryptography has been an active research area over the last 10 years. Many innovative ciphers have been proposed in order to optimize various performance criteria. Recently, resistance against side-channel attacks has been considered as an additional optimization goal for low-cost ciphers, as exemplified by the algorithms PICARO [41] and `Zorro` [20]. Both proposals aim at leading to efficient *masked* implementations (i.e. where all the computations are performed on shared secrets). Starting from the observation that the performance overheads in such implementations primarily come from non-linear operations, Piret et al. [41] first investigated how to reduce their amount by considering non-bijective S-boxes. Next, Gérard et al. [20] further exploited "irregular" SPN structures, i.e. where not all the state goes through the (bijective again[1]) S-boxes in each round. Both examples lead to performance gains over the AES Rijndael, which become more significant as the number of shares increases.

These previous works lead to several useful observations regarding the relation between masking and the linear/non-linear operations used in block ciphers. In this paper, we aim to complement them by focusing on two important scopes for further research they left open. First from the performance point-of-view,

[1] Motivated by the recent results in [51], showing that non-bijective S-boxes lead to easily exploitable targets for generic (non-profiled) Differential Power Analysis.

both PICARO and Zorro minimize the number of field multiplications per encrypted plaintext. This is a natural direction as it leads to improvements applicable to both Boolean [45] and polynomial [43] masking schemes. Yet, further specialization to Boolean masking could potentially lead to additional gains. For example, the S-boxes of lightweight ciphers PRESENT [7] and NOEKEON [15] require three multiplications in $GF(16)$, which makes them less suitable than Zorro and PICARO for polynomial masking. But they have efficient representations minimizing the number of AND gates which could be exploited in Boolean masked implementations. Next from the security point-of-view, both designs are based on somewhat unusual Feistel/SPN structures (in order to deal with non-bijective S-boxes in [41], and to minimize the number of S-boxes per round in [20]). So another (quite pragmatic) open question is whether we can design ciphers for efficient masking based on more standard techniques (e.g. directly exploiting the wide-trail strategy [16] as other lightweight algorithms).

In this context, we base our investigations on two additional observations. First, Boolean masking is particularly efficient when applied to operations that are linear over $GF(2)$ (since such operations can be performed independently on each share). As a result, and in contrast with many existing block ciphers, it appears interesting to have linear diffusion layers implemented as look-up tables, since they can be straightforwardly exploited for any number of shares[2]. Second, since our focus is on software implementations, we also have a strong incentive for simple and regular designs, where computations are always performed on well aligned data. For example, manipulating bits and bytes such as in PRESENT raises additional challenges for the implementers (to guarantee that the bit manipulations do not leak more information than the byte ones). These observations combine into the conclusion that a bitslice cipher with look-up table-based diffusion layers and non-linear S-boxes with efficient gate-level representation seems an excellent candidate for efficient Boolean masked software implementations.

Following, our contributions are threefold. First, we separately analyze S-boxes and linear layers meeting the previous objectives, and compare a number of constructions from the cryptanalytic and efficient masking point-of-view. Interestingly, this part of our study confirms the previous observation that if side-channel resistance via masking is added as a block cipher design criteria, the balance between linear and non-linear operations has to be changed towards more linear ones. Such investigations open a large space of possible ciphers that we define as LS-designs (essentially made of a combination of look-up table-based L-boxes and bitslice S-boxes). We then argue that such designs have interesting properties for efficient masking. For this purpose and for concreteness, we specify two instances of 128-bit block ciphers and analyze their security against a number of standard cryptanalytic techniques. Doing so, we paid attention to make our studies as generic as possible (i.e. leading to conclusions for LS-designs rather than for their instances). We also considered the impact of choosing involutive or non-involutive components, and show that the first ones mainly lead the security guarantees provided by the wide-trail strategy to be tighter. Eventually, we

[2] Masking non-linear look-up tables has a cost that is quadratic in this number [11].

compare the performances of our two exemplary instances with the ones of the AES, PICARO, `Zorro` and NOEKEON on an 8-bit microcontroller (and argue that they also behave well on desktop CPUs with SIMD units). Overall, these results confirm the interest of bitslice ciphers in the context of physically secure implementations, as hinted in [14]. While the instances of designs we suggest are not yet optimal (because of the hardness of finding optimal L- and S-boxes) and do require more analysis, their performances are comparable to (or slightly better than) the ones of NOEKEON, which is known to have an extremely compact gate-level representation [29]. Therefore, we believe LS-designs formalize an interesting family of ciphers combining excellent performances (also for unprotected implementations), strong security guaranties, regularity/simplicity and efficient masking, i.e. properties that generally benefit to resistance against side-channel attacks and are also appealing for more general applications.

2 Design rationale

2.1 Bitslice S-boxes

In this first subsection, we analyze various S-boxes having an efficient bitslice representation. Our comparisons will consider various sizes (namely 4-bit, 8-bit and 16-bit), in order to study their tradeoff with the different diffusion layers in the next subsection. They will also take into account both standard cryptographic properties (such as the non-linearity, differential profile and algebraic degree of which the definitions are recalled in Appendix A) and masking efficiency considerations. For this purpose, and following the techniques in [26], we will simply consider the number of AND and XOR gates needed for each S-box. As already mentioned, the cost of XOR gates is linear in the number of shares in the masking scheme, while it is quadratic for AND gates (which will therefore count as a more important criterion in our evaluations). Note that since we are only interested in non-linear S-boxes, each Boolean function defining them has to be linearly independent of the other ones. As a result, we need at least the same number of AND gates as the output size of the S-box to reach this goal.

Why bitslicing? In Appendix B, we recall the method to perform secure non-linear operations first proposed in [26] and generalized to extension fields in [45]. From Algorithm 2, it is easy to see that the difference of performances between Boolean and polynomial masking can (at least partially) be explained by the implementation efficiency of the underlying non-linear operation. For example, an AND gate can usually be performed in a single clock cycle on most computing devices. By contrast, a field multiplication generally requires the use of log/alog tables (if large fields are considered) which will typically account for 20 to 40 clock cycles in embedded microcontrollers [22]. These numbers can be improved when small fields are considered, e.g. multiplication can tabulated if elements are represented with less than 4 bits, but even in this case the non-linear operations will require 3 to 5 cycles. As a result, masking at the gate-level in a bitslice manner as we investigate next should bring performance improvements.

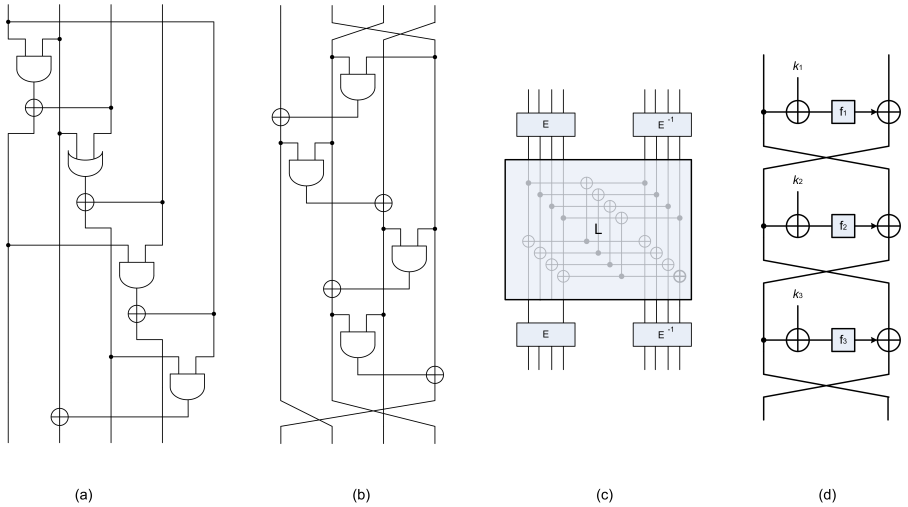


Fig. 1. (a) Non involutive 4-bit S-box with optimal bitslice representation. (b) Involutive 4-bit S-box with optimal bitslice representation. (c) Construction of 8-bit S-boxes from 4-bit ones as in the Whirlpool hash function [44]. (d) Construction of $2s$-bit S-boxes from s-bit ones as in the MISTY block cipher [36].

We now present a couple of S-boxes with efficient gate-level representation. The main challenge is that the enumeration of S-boxes is rapidly out of reach as their size increases. Besides, finding the best gate-level description of a large S-box is also a hard problem. As a result, we will start from the 4-bit case for which exhaustive analysis is possible, and then take advantage of heuristics from the block cipher literature, in order to turn these 4-bit S-box into larger ones.

4-bit S-boxes. An exhaustive search of optimal bitslice S-boxes can be found in [48]. Its main result is that the so-called "Class 13" is the best option for this purpose, and can be implemented with 4 non-linear gates and a total of 9 instructions, with the best differential and linear probabilities that can be reached. It is represented in Figure 1 (a). Its only limitation is that it is not involutive. We ran a similar exhaustive search with a slightly larger instruction set (including nor gates, nand gates, and more copy instructions) while restricting the number of non-linear gates to 4. As a result, we could find an involutive S-box with similar properties as the Class 13 one: it is represented in Figure 1 (b). Note that the use of Toffoli gates (defined in [47]) allows to see this S-box as a generalized Feistel network, which explains the involution property.

From 4-bit S-boxes to larger ones. Various constructions can be considered for this purpose, ranging from ad hoc (scaling 4-bit S-boxes to 8-bit ones) to generic solutions (scaling s-bit S-boxes to $2s$-bit ones). As in [20], we analyzed a couple of natural candidates and report on the most interesting results.

Table 1. Comparison of our different S-box proposals.

	# AND	# XOR	size	Involutive	deg(S)	$\Pr_{\text{diff}}$	$\Pr_{\text{lin}}$
NOEKEON	4	7	4	Yes	3	2^{-2}	2^{-1}
Class 13 [48]	4	4	4	No	3	2^{-2}	2^{-1}
Figure 1 (b)	4	4	4	Yes	3	2^{-2}	2^{-1}
AES [9]	32	83	8	No	7	2^{-6}	2^{-3}
Whirlpool + Class 13	16	41	8	No	6	$2^{-4.68}$	2^{-2}
Whirlpool + Figure 1(b)	16	42	8	No	6	$2^{-4.68}$	2^{-2}
MISTY + Class13	12	24	8	Yes	6	2^{-4}	2^{-2}
MISTY + Figure 1(b)	12	24	8	Yes	5	2^{-4}	2^{-2}
MISTY + 3/5-bit S-boxes	11	25	8	No	5	2^{-4}	2^{-2}
MISTY^2 + Class13	36	96	16	Yes	13	2^{-8}	2^{-4}

In the first (ad hoc) case, we considered a proposal coming from the Whirlpool hash function (which only requires four 4-bit S-boxes) and generalized it by considering a linear layer between the two levels of S-boxes (see Figure 1 (c)). Since our goal is to minimize the number of AND gates, such a solution was better than proposals with six 4-bit S-boxes as in the KHAZAD block cipher [1].

Alternatively, we looked at Feistel networks such as used in the MISTY block cipher [36] and represented in Figure 1 (d). A minimum of three rounds were considered in order to avoid trivial weaknesses with respect to linear and differential cryptanalyses. One advantage of such a construction is that it directly gives rise to involutive components. Besides, it has been shown that three rounds of such a network allow squaring the linear and differential probabilities of the round function, on average over the keys. Note however that the impact of this averaging only appears for 16-bit (or larger) S-boxes (i.e. when applying the MISTY structure twice, recursively), because the impact of the linear hull effect only becomes significant from this size on [40]. The exhaustive search over all the 16-bit S-boxes having the structure of Figure 1 (d) was too computationally intensive and we only report on the best candidate we found. Note that for 8-bit S-boxes, we additionally investigated unbalanced Feistel networks built from 3- and 5-bit ones (as also proposed with the MISTY cipher), which provided a slightly improved non-involutive candidate (with one less AND gate).

The results of our different S-box searches are summarized in Table 1. For comparison purposes, we also reported the same metrics for the NOEKEON S-box, and the bitslice representation of the AES S-box proposed in [9].

2.2 Table-based diffusion layers

In a bitslice implementation of a block cipher, the same register i holds the i-th bit of several S-box inputs/outputs. In this context, we are interested in linear diffusion boxes (next denoted as L-boxes) that mix bits inside these registers and can be applied to them in parallel. From an implementation point of view, computing an L-box will just correspond to a table access. Yet, and compared to the usual case of non-linear tables, we benefit from more flexibility. Namely, since

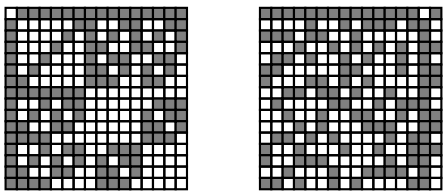
the table is linear, it can be decomposed into several smaller tables. This can be useful in order to only store tables that are adapted to the memory characteristics of the target platform. For instance, a 16-bit L-box can be implemented as four 8-bit to 8-bit look-up tables and two XORs. Our comparisons of diffusion layers will mainly be based on the branch number. Thanks to this number, we can directly evaluate the linear and differential properties of any design based on a combination of S-boxes and L-boxes according to the wide-trail strategy. We define the branch number of an L-box L as:

$$\mathcal{B}(\mathsf{L}) = \min_{x \neq 0} \big(|x| + |\mathsf{L}(x)|\big).$$

It follows that the linear branch number and the differential branch number of our diffusion layer is also $\mathcal{B}(\mathsf{L})$. This guarantees that any non-trivial trail in two consecutive cipher rounds will have at least $\mathcal{B}(\mathsf{L})$ active S-boxes (see [16, Th. 1]). Note that an l-bit L-box with branch-number b is equivalent to a binary linear code with parameters $[2l, l, b]$ (length $2l$, dimension l, distance b). Therefore, we can use results from coding theory to design our L-boxes, such as [21].

8-bit L-boxes. The highest branch number possible for an 8-bit L-box is 5. We ran an exhaustive search and we found $2^{25.2}$ candidates with such a branch number, including 33 involutions. They activate at least 5/16 of the S-boxes.

16-bit L-boxes. The highest branch number possible for a 16-bit L-box is 8. It is not feasible to run an exhaustive search, but there are several known codes with parameters $[16, 8, 8]$. In particular, it is possible to build a quite structured 16-bit involution with branch number 8 from a systematic generator of the Reed-Muller code $RM(2, 5)$, as represented in the left part of the figure below (a non-involutive candidate is given on the right part of the same figure). This kind of L-boxes activates at least one fourth of the S-boxes over two rounds.



32-bit L-boxes. For a 32-bit L-box, the optimal branch number is not known. The best known code gives a branch number of 12, and the known upper bound shows that it is impossible to reach a branch number higher than 16. Therefore the best known option will only activate 12/64 of the S-boxes.

We show a comparison of the best known L-box diffusion layer and the AES diffusion layer in Table 2. This shows that L-box diffusion layers do not have security bounds as good as AES-like diffusion layers (mainly because they are obtained over two rounds rather than four), but the ability to use a bitslice implementation is an important advantage for side-channel resistance.

Table 2. Comparison of linear layers.

	# S-boxes	Active S-boxes
8-bit L-box	8	5/16 (31.25%)
16-bit L-box	16	8/32 (25%)
32-bit L-box	32	12/64 (18.75%)
AES linear layer	16	25/64 (39.06%)

Alternatively if the state is considered as a vector of elements over a larger field (the S-box outputs), the diffusion layer can be written as a binary matrix. This approach has been used previously, *e.g.* in the design of ARIA [33].

2.3 Which S-box with which L-box?

The composition of s-bit S-boxes and l-bit L-boxes directly gives rise to various candidate $n = l \times s$-bit ciphers. In this subsection, we illustrate the tradeoffs resulting from these choices in a 64-bit case (with 8-bit and 16-bit L-boxes).

4-bit S-box and 16-bit L-box. Using the components described previously, the best involutive S-box requires 4 linear operations and 4 non-linear ones, and achieves $\Pr_{\text{diff}} = 2^{-2}$ and $\Pr_{\text{lin}} = 2^{-1}$. Therefore, we need at least 32 active S-boxes to have a secure cipher. Since we have 8 active S-boxes every 2 rounds using a 16-bit L-box, this corresponds to at least 8 rounds. In an 8-bit CPU, it would require 64 non-linear operations, 128 XORs and 128 table look-ups.

8-bit S-box and 8-bit L-box. Using the components described previously, the best involutive S-box requires 24 linear operations and 12 non-linear operations, and achieves $\Pr_{\text{diff}} = 2^{-4}$ and $\Pr_{\text{lin}} = 2^{-2}$. Therefore, we need at least 16 active S-boxes to have a secure cipher. Since we have 5 active S-boxes every 2 rounds using a 8-bit L-box, this corresponds to roughly 6 rounds. In an 8-bit CPU, it would require 72 non-linear operations, 144 XORs and 48 table look-ups.

Interestingly, we can see that the first option requires a total of 320 elementary operations, to be compared with only 264 ones for the second one. By contrast, the first option has a reduced number of non-linear operations, which will gradually dominate if a masked implementation with large number of shares is considered. While somewhat specific, this example confirms the trend already observed in [20] that the ratio between the amount of linear and non-linear operations increases in block ciphers that are easier to mask. Besides, it also shows that a small L-box can activate a larger proportion of the S-boxes, but these larger S-boxes are generally more expensive (if they are selected to have good cryptographic properties). In this 64-bit comparison, the two effects are of similar magnitude, but the conclusion is of course dependent on the block cipher size and on the knowledge we have about large bitslice S-boxes and L-boxes.

Algorithm 1 LS-design with l-bit L-boxes and s-bit S-boxes ($n = l \cdot s$)

$x \leftarrow P \oplus K$; ▷ x is a $s \times l$ bits matrix
for $0 \leq r < N_r$ **do**
  for $0 \leq i < l$ **do** ▷ S-box Layer
    $x[i, \star] = \mathsf{S}[x[i, \star]]$;
  for $0 \leq j < s$ **do** ▷ L-box Layer
    $x[\star, j] = \mathsf{L}[x[\star, j]]$;
  $x \leftarrow x \oplus K \oplus C(r)$; ▷ Key addition and round constant
return x

3 LS-designs specifications

Following the previous section, we can define LS-designs as the family of block ciphers specified in Algorithm 1. The description directly suggests simplicity and regularity as one important advantage of such ciphers: instances can be characterized by selecting a bitslice S-box S, an L-box L, a number of rounds N_r and constants $C(r)$. In the next sections, we will consider two 128-bit instances of LS-designs in order to illustrate their security against cryptanalysis and good implementation properties. The bit-size was chosen both because of the observations in [49] and in order to be comparable with NOEKEON (which is among the best ciphers published so far for efficient bitslice representation).

Involutive instance (Robin). We take the S-box denoted as "MISTY + Class13" from Table 1 and the involutive L-box in Section 2.2. The cipher has 16 rounds and constants are computed as $[\mathsf{L}(i)\ 0\ \ldots\ 0]$ with i the round index.

Non-involutive instance (Fantomas). We take the S-box denoted as "MISTY + 3/5-bit S-boxes" from Table 1 and the non-involutive L-box in Section 2.2. The cipher has 12 rounds and uses the same constants as Robin.

4 Security evaluation

We now investigate the security properties of LS-designs, trying to extract general conclusions that apply to our family of ciphers in the first place. For concreteness, we will also consider more specific claims related to the aforementioned instances. In this respect, we note that Robin and Fantomas were specified with slightly different goals. Namely, the first one aims to have security margins similar to NOEKEON, while the second was mainly defined in order to illustrate the impact of choosing involutive components with respect to the efficiency limits that can be expected with LS-designs. Note that we aim for single-key security in both cases (i.e. exclude related-key and chosen key attacks from our claims). In particular, there is a simple related-differential with a single active S-box per round if the state difference can be corrected using a key difference.

4.1 Security against linear and differential cryptanalysis

As explained in Section 2.2, the structure of the cipher gives a simple upper bound on the maximum probability of differential characteristics, and the maximum bias of linear trails. Any two-round trail activates at least $\mathcal{B}(L)$ S-boxes, and this gives the following bounds for any $2r$-round trail:

$$\mathrm{Pr}_{\mathrm{lin}}(2r) \leq \mathrm{Pr}_{\mathrm{lin}}^{\max}(S)^{r \cdot \mathcal{B}(L)}, \qquad \mathrm{Pr}_{\mathrm{diff}}(2r) \leq \mathrm{Pr}_{\mathrm{diff}}^{\max}(S)^{r \cdot \mathcal{B}(L)}. \tag{1}$$

With the parameters of Section 3, this gives:

$$\mathrm{Pr}_{\mathrm{lin}}(2r) \leq 2^{-16 \cdot r}, \qquad \mathrm{Pr}_{\mathrm{diff}}(2r) \leq 2^{-32 \cdot r}.$$

Such bounds prevent simple linear and differential attacks based on a trail over more than 8 rounds. We use 16 (resp. 12) rounds in Robin (resp. Fantomas), so as to have a good (resp. less conservative) security margin. We now study the tightness of these bounds, and how to build optimal differential/linear trails.

Product trails for Robin. To study differential and linear trails, we first consider a set of special states that can be written as the tensor product of an s-bit vector (corresponding to the S-box input and denoted with Greek letters) and an l-bit vector (corresponding to the L-box input and denoted with Latin letters):

$$\alpha \otimes x = \begin{bmatrix} \alpha_0 x_0 & \alpha_0 x_1 & \alpha_0 x_2 & \alpha_0 x_3 & \alpha_0 x_4 & \cdots & \alpha_0 x_l \\ \alpha_1 x_0 & \alpha_1 x_1 & \alpha_1 x_2 & \alpha_1 x_3 & \alpha_1 x_4 & & \alpha_1 x_l \\ \alpha_2 x_0 & \alpha_2 x_1 & \alpha_2 x_2 & \alpha_2 x_3 & \alpha_2 x_4 & \cdots & \alpha_2 x_l \\ \vdots & & & & \vdots & \ddots & \vdots \\ \alpha_s x_0 & \alpha_s x_1 & \alpha_s x_2 & \alpha_s x_3 & \alpha_s x_4 & \cdots & \alpha_s x_l \end{bmatrix}.$$

For those states, the S-box layer and the L-box layer act independently:

$$\mathsf{S\text{-}layer}(\alpha \otimes x) = \mathsf{S}(\alpha) \otimes x, \qquad \mathsf{L\text{-}layer}(\alpha \otimes x) = \alpha \otimes \mathsf{L}(x).$$

Hence we have the same behavior for differences and for linear masks. Namely, we can build differential characteristics (resp. linear trails) where the differences (resp. selection masks) are written as tensor products. In particular, if $\mathsf{L}[x] = y$, and $\alpha \rightsquigarrow \beta$ with probability p through the S-box, then $x \otimes \alpha \rightsquigarrow y \otimes \beta$ through one round with probability $p^{|x|}$, where $|x|$ denotes the Hamming weight of x.

When the cipher is built as an involution, we further have $\beta \rightsquigarrow \alpha$ with probability p, and $\mathsf{L}[y] = x$, hence $y \otimes \beta \rightsquigarrow x \otimes \alpha$ through one round with probability $p^{|y|}$, giving an iterated two-round trail (as illustrated with a toy example in Figure 2). If α, β, x, and y are chosen optimally, this path reaches the security bound of Equation (1), hence showing that the bound is tight. Using the parameters of Robin, optimal choices of α and β give $p = 2^{-4}$, while optimal choices of x and y give $|x| + |y| = 8$. This directly leads to a two-round iterated differential characteristic with probability 2^{-32} (or 2^{-128} for 8 rounds), and a two-round iterated linear trail with bias 2^{-16} (or 2^{-64} for 8 rounds).

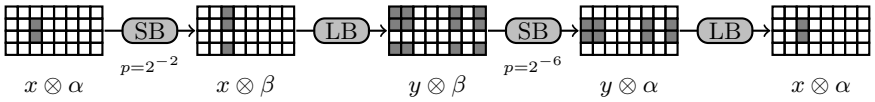


Fig. 2. Toy example of product trail for an involution-based 32-bit cipher, with $\alpha = \texttt{0110}$, $\beta = \texttt{1101}$, $x = \texttt{00100000}$, $y = \texttt{11000101}$, $\Pr[\alpha \rightsquigarrow \beta] = 2^{-2}$.

Truncated differentials. Since the L-layer and the S-layer operate independently on product states, there are many trails with the same active S-boxes but with different inputs and outputs. As long as all the S-boxes in a round have the same output difference, the state is a product state, and all these differences contribute to the same truncated trail. The space of such product states is quite small and we can search exhaustively for the best truncated differential using product trails. We note that there could be better truncated trails but it seems that the product trails are dominant when we consider an involution.

For Robin, the best truncated trails start with a single active S-box, and have alternatively 1 and 7 active S-boxes. A round with a single active S-box has a probability one (to follow the truncated trail), and a round with 7 active S-boxes has a probability of $2^{-28.5}$ if the input difference is random (if it comes from a previous round, the probability is slightly skewed). The best such trails for 8 and 9 rounds have a probability of $2^{-112.1}$, while the best trails for 10 and 11 rounds have a probability of $2^{-139.8}$. These results can be used in a truncated differential attack as follows. If one takes a pair of states with a single active S-box, there will be a single active S-box (the same one) after 9 rounds with probability 2^{-112}. By using a structure of 256 plaintexts with 120 bits set to a fixed value and the last S-box input taking all possible values, we obtain 2^{15} different pairs with a single active S-box. This gives 2^{112} input pairs if we collect 2^{97} such structures, and we expect one pair with only one active S-box in the output. Since such events only happen with probability 2^{-120} with a random function, we don't expect any false positive after 2^{112} pairs. As a result, we have a distinguisher for 9 rounds with a cost of 2^{104}. We additionally expect that this distinguisher can be extended to a few more rounds using partial decryption.

From involutive to non-involutive components. If we do not restrict the design to involutive S- and L-boxes, we can hope that the bound given by Equation (1) will not be tight. More precisely, we expect that there should not be any trail reaching the minimal number of active S-boxes with the optimal probability for every S-box transition. For this purpose, we first count the number of active S-boxes for truncated trails. That is, for each state we only care about which columns are non-zero and build all the possible transitions. In this context, it is important to note that a truncated input to the diffusion layer can give several different truncated outputs, and does not necessarily behave linearly. For instance, if we start from 00101000, we have to consider five possible transitions:

$$\mathsf{L}[\texttt{00101000}], \mathsf{L}[\texttt{00100000}] \vee \mathsf{L}[\texttt{00001000}], \mathsf{L}[\texttt{00101000}] \vee \mathsf{L}[\texttt{00100000}],$$
$$\mathsf{L}[\texttt{00101000}] \vee \mathsf{L}[\texttt{00001000}], \mathsf{L}[\texttt{00101000}] \vee \mathsf{L}[\texttt{00100000}] \vee \mathsf{L}[\texttt{00001000}].$$

More generally, the possible transitions are of the following form:

$$x_0 \vee x_1 \vee \ldots \vee x_l \quad \rightsquigarrow \quad \mathsf{L}[x_0] \vee \mathsf{L}[x_1] \vee \ldots \vee \mathsf{L}[x_l]$$

Non-linear transitions will usually lead to states with more active columns, but the extra degrees of freedom from the non-linearity allow better trails than the product trails with only linear transitions. For $l = 8$, we ran an exhaustive search over all L-boxes with branch number 5, and found that the best ones give trails with at least 53 active S-boxes for 16 rounds (rather than 40 active S-boxes when L is an involution). For $l = 16$, building all the possible transitions for a fixed L-box is already a hard problem, so we cannot test many different L-boxes. We ran a randomized search by permuting the lines and columns of the $RM(2, 5)$ systematic generator used in Section 2.2. The best non-involutive L-box we found (given in Section 2.2) gives truncated trails with at least 64 active S-boxes over 12 rounds. More precisely, we can compute the minimum number of active S-boxes with an involutive L-box and our best non-involutive L-box as:

Rounds	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Involutive	1	8	9	16	17	24	25	32	33	40	41	48	49	56	57	64
Non-involutive	1	8	12	20	24	30	34	40	46	52	58	64	68	74	80	86

If we consider a 64-bit cipher with $l = 16$ and $s = 4$, we can go further in the analysis and find the best differential trails with completely instantiated differences. We expect that this will again improve the upper bound on the probability of trails, because in general it is not possible to select a specific difference so that all S-box transitions have maximal probability. We ran this search using an A* algorithm [24], with some additional ideas from Matsui's branch-and-bound algorithm [35]. We used the best truncated trails as heuristic estimate for future path-cost in A*, and refined it by computing the best trails with increasing numbers of rounds. Once we know the probability of the best instantiated r-round trail, we update the heuristic if some truncated r-round trails were expected to have a higher probability. Doing so, we found that some choices of L-box and S-box give 6-round trails with probability at most 2^{-64} and 8-round trails with probability at most 2^{-90} (the candidate L-box in Section 2.2 together with S-box $S = \{6, 1, 0, 7, \mathtt{E}, 4, \mathtt{F}, \mathtt{D}, 5, \mathtt{B}, 2, \mathtt{C}, 3, 8, \mathtt{A}, 9\}$ is an example). These values should be compared with 2^{-48} (resp. 2^{-64}) if L and S are restricted to involutions, and to the previous bound of 2^{-56} (resp. 2^{-80}) for the same components using the analysis with truncated trails. This indicates that LS-designs based on involutive components require about 4/3 as many rounds as with non-involutive components to reach a similar security level for these parameters ($l = 16$ and $s = 4$). The computation took several days and dozens of gigabytes of RAM. We believe it gives a good indication about the relative security of involutive vs. non-involutive ciphers that should also be valid with larger S-boxes, even though we cannot run the search for optimal trails with $l = 16$ and $s = 8$ in practice.

Application to Fantomas. Using our search for truncated trail, we have found the following bounds for linear and differential trails on Fantomas:

$$\Pr_{\text{lin}}(6) \leq 2^{-56}, \qquad \Pr_{\text{diff}}(6) \leq 2^{-112},$$
$$\Pr_{\text{lin}}(7) \leq 2^{-68}, \qquad \Pr_{\text{diff}}(7) \leq 2^{-136}.$$

These bounds imply that simple linear and differential attacks on Fantomas can only work with trails over 6 rounds or less, and we have a security margin of 6 rounds. We expect that this bound is not tight, as shown by our analysis of instantiated trails on a 64-bit LS-design (but we cannot run a similar analysis on Fantomas in practice). In addition, we note that the best attack on Robin is based on a truncated differential corresponding to several simple trails, but this effect will be quite limited for Fantomas because optimal trails do not have the strong structure of product trails (on 6 rounds, the best truncated differential using a collection of product trails has a probability of 2^{-117}).

Impossible differentials. We finally searched for a class of impossible differentials where we do not use the S-boxes properties, i.e. we only considered which S-boxes are active at each round, and we used the possible transitions of the linear layer combined with the fact that the S-box is a bijection. This search is similar to the search for truncated trails described above, and the hardest part is again to build all the possible transitions through L operations. We found that the longest impossible differential for Robin and Fantomas (in this class) only spans three rounds. For Robin, there are are 48420 impossible input-output patterns, an example is given by $0000000000000001 \not\rightarrow 0000000000000010$. For Fantomas, there are 35951 impossible differentials, an example is given by $0000000000000001 \not\rightarrow 0000000000000001$. We can compare this result with similar results on the AES. The best known impossible differential is in this class of impossible differentials and spans four rounds. Since our diffusion layer mixes all the 16 S-box input every round, it makes sense to have shorter impossible differentials.

4.2 Generic attacks against Even-Mansour ciphers

In 1991, a simple block cipher design was proposed by Even and Mansour, using only one permutation and two different key values [19]. It was later revisited in [19] and extended towards key-alternating ciphers by using n permutations and $n + 1$ key values. One special case of such ciphers is the Single-key Even-Mansour (SEM) scheme, which is defined by using n permutations together with one key. Due to their simplicity, these designs have attracted the attention of several cryptanalysts, and various published results apply to their generic versions [12, 17, 18] or particular instances such as LED or Zorro [23, 27, 37, 38].

Since LS-designs correspond to SEM schemes, we discuss the applicability of these previous results to our case study. In particular, and although we do not claim security against related-key and chosen-key attacks, we briefly look at these adversarial scenarios in a generic manner. Starting with chosen-key differentials,

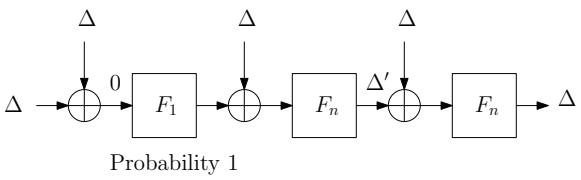


Fig. 3. Three-round related-key characteristic for LS-designs.

the attacker can not only control the n-bit input value but also the n-bit key value. As a result, the number of freedom degrees he has is doubled (to $2n$) compared to the standard differential cryptanalysis. We know from Section 4.1 that the best differential characteristics for LS-designs have probability:

$$\Pr_{\text{diff}}(2r) \leq \Pr_{\text{diff}}^{\max}(S)^{r \times \mathcal{B}(L)}.$$

Therefore, it is possible to mount an attack for $2r$ rounds if $2^{-2n} \leq \Pr_{\text{diff}}(2r)$. Taking the example of Robin, the best 2-round characteristic has probability 2^{-32} and the number of freedom degrees equals 256. Thus, in the worst case it could be possible to attack $8 \cdot 2 = 16$ rounds by using all degrees of freedom. Similar observations can be made in the related-key setting. Namely, by applying the results from [38], the best differential characteristic over two rounds can be extended to three rounds for LS-designs (as represented in Figure 3). Hence, the required number of rounds to achieve n-bit security will increase by 50%.

4.3 Algebraic attacks

In algebraic cryptanalysis, a cipher is expressed as a large system of non-linear equations (typically over $GF(2)$) and a solution for the system is searched. Although it is possible to describe any algorithm in terms of multivariate equations, solving them is an NP-hard problem already for quadratic ones. The precise complexity of algebraic cryptanalysis is difficult to evaluate and security against these attacks is usually argued by exhibiting the size and number of unknowns in the systems, together with a reasoning about the cipher's algebraic degree.

S-boxes in LS-designs can be described in the same number of equations as the number of non-linear gates. Let e denote the number of non-linear gates, l denote the size of the L-box and N_r by the number of rounds. Then, the entire system for a fixed key LS-design consists of $(N_r \cdot e \cdot 128/l)$ quadratic equations in $(N_r \cdot 128 \cdot 2)$ variables. That leads to 3072 equations in 4096 variables for Robin and 2112 equations in 3072 variables for Fantomas (the AES has 6400 equations in 2560 variables). We expect these numbers to be sufficient for both instances to be secure against algebraic attacks, in view of the time and memory complexities needed to solve small-scale AES variants presented in [10]. As for the algebraic degree, we used the work [8] to compute the cumulative algebraic degree in function of the number of rounds. This algebraic degree reaches maximum after five rounds for both Robin and Fantomas, in which case a partition of size 2^{127} is required to construct zero-sum distinguishers. More precisely, we have:

# of rounds	1	2	3	4	5	6	7	8
Robin	6	36	112	125	127	127	127	127
Fantomas	5	25	110	125	127	127	127	127

4.4 Other cryptanalyses

We use round constants to make all rounds different and prevent slide attacks [6] (that are based on the self-similarity of the round function). Rotational cryptanalysis [30] is a powerful technique against Addition-Rotation-XOR (ARX)-based block ciphers. But the application of S-box operations in LS-designs makes this attack unlikely to succeed. Integral cryptanalysis or square attacks [13] are primarily purposed for word-oriented ciphers but can be adapted to bitslice ones. Our analyses suggest that up to 4 rounds of Robin or Fantomas can be targeted in this way (which also leaves comfortable security margins). Eventually, boomerang attacks [50] assume that we can find two characteristics for $(2 \cdot r_1 + 1)$ and $(2 \cdot r_2 + 1)$ rounds in the target algorithms. By using Equation (1), we can approximate these probabilities as $\Pr(2 \cdot r + 1) \leq \Pr_{\text{diff}}^{\max}(S)^{r \times \mathcal{B}(L)+1}$. Hence, the probability of a boomerang distinguisher becomes $\Pr_{\text{diff}}^{\max}(S)^{2 \cdot ((r_1+r_2) \times \mathcal{B}(L)+2)}$, which must be better than 2^{-n}. Setting the parameters of Robin or Fantomas in this equation, we find $2(r_1 + r_2) + 2 < 5.5$ and the attack works for at most five rounds. Therefore, boomerang attacks should not a concern for LS-designs.

5 Performance evaluations

The main objective of LS-designs is to allow efficient and secure software implementations for 8-bit micro-controllers. Therefore, we first report on the performances of protected implementations of Robin and Fantomas on an Atmel ATmega644p micro-controller, together with the AES, `Zorro`, PICARO and NOEKEON[3]. The results in Figure 4 (given for different number of shares in the masking scheme) show that the performances of Robin and NOEKEON (both involutive ciphers) are remarkably close. They confirm that bitslice ciphers optimized for Boolean masking allow more efficient implementations that previously obtained, e.g. with the AES, `Zorro` or PICARO. They also illustrate the additional gains that can be obtained by considering non-involutive components (e.g. with Fantomas). Combined with a highly regular design, with all operations operating on well-aligned 8-bit data, we believe this evaluation supports the conclusion that LS-designs are promising ciphers for side-channel resistance.

Besides, we also found that LS-designs are very efficient on desktop CPUs with large SIMD units, at least for unprotected implementations. Taking the example of Fantomas in counter mode, we can evaluate several inputs in parallel and use the full width of SIMD units. Let us describe in more details an implementation using SSSE3 instructions with 128-bit registers. We will compute 16

[3] LED and PRESENT have the same number of non-linear gates, but encrypt only 64-bit. So we do not expect them to bring improvements in our masked setting.

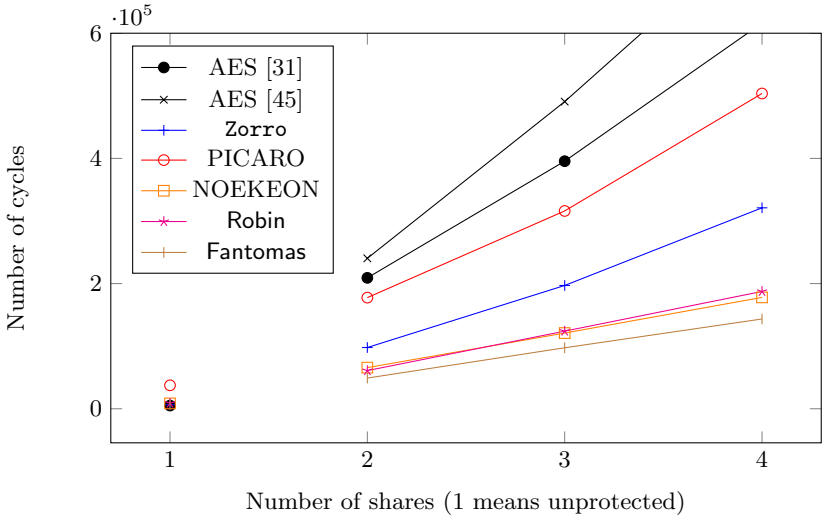


Fig. 4. Encryption time for a 128-bit block in an Atmel AtMega644p.

instances of Fantomas in parallel (with 16 different plaintexts), using 16 SSE registers, every register containing one byte from each copy of Fantomas (8 registers for the high order bytes, and 8 other registers the low order bytes). The S-box layers compute two sets of 128 S-boxes in parallel, using 128-bit wide bitwise operations; this takes 96 instructions. For the L-box layer, we use the `pshufb` instruction as a 4-bit to 8-bit look-up table. The 16-bit L-box is decomposed as eight 4-bit to 8-bit look-up tables and 6 XORs[4]; our implementation requires 280 instructions to compute 16 parallel linear layers (*i.e.* 128 16-bit L-boxes). With these figures, our implementation of Fantomas runs at 6.3 cycles/byte (for long messages) on a Core i7 CPU (Nehalem micro-architecture). Thanks to the `pshufb` instruction, the L-box layer is not subject to cache timing attacks.

As a point of comparison, the bitsliced AES implementation of Käsper and Schwabe [28] would take respectively 326 and 102 cycles for the same number of S-boxes and linear layers (the full AES takes 6.9 cycles/byte on the same CPU – this is the faster known implementation of AES of this CPU). On the one hand our S-box is much easier to implement in a bitslice way than the AES S-box, since it was one of our design goals. On the other hand, our linear-layer is optimized for a table-based implementation, and more complex than that of the AES. It can still be implemented rather efficiently, but it becomes the dominant factor in this implementation. This shows that LS-designs can reach performances comparable to the AES on high-end CPUs, excluding implementations using hardware AES instructions. We also expect reasonable performances on Atom or ARM Cortex-A CPUs, which are used in some embedded systems and include a good vector engine with a permutation instruction (SSSE3 and NEON, respectively).

[4] This can be reduced to seven table look-ups for Robin, thanks to the L-box structure.

Table 3. Implementation results with a parallel mode for long messages.

	Fantomas	Robin	AES	
			w/o AES-NI [28]	w/AES-NI
ARM Cortex A15	14.2	18.1	17.8	N/A
Atom	33.3	43.5	17	N/A
Core i7 Nehalem	6.3	8.1	6.9	N/A
Core i7 Ivy Bridge	4.2	5.5	5.4	1.3

Moreover, the latest Intel CPUs support 256-bit wide SIMD operation using AVX2 operations; we expect that this will give even better performances [5].

6 Open problems

This paper introduces LS-designs as an interesting family of secure and efficient block ciphers, with good properties for masked implementations. Since their instantiation mainly depends on the selection of good S- and L-boxes, a natural scope for further research is to find better such components, in particular for large bit-sizes (e.g. 8-bit and more for S-boxes, 32-bit and more for L-boxes). Improvements in these lines would directly lead to more optimized ciphers.

Besides, our current investigations mainly considered software implementations. But the efficient gate-level representation of Robin and Fantomas makes them potentially suitable for hardware implementations as well. As a result, it would be interesting to study their threshold implementations, and to compare the resulting performances with other algorithms that are efficient in this setting, such as NOEKEON again [39] or more recent designs like FIDES [5].

Acknowledgements. This work has been funded in parts by the ERC project 280141 (acronym CRASH). François-Xavier Standaert is an associate researcher of the Belgian Fund for Scientific Research (FNRS-F.R.S.).

References

1. Barreto, P., Rijmen, V.: The KHAZAD legacy-level block cipher. Primitive submitted to NESSIE (2000) 4
2. Bertoni, G., Coron, J.S., eds.: Cryptographic Hardware and Embedded Systems - CHES 2013 - 15th International Workshop, Santa Barbara, CA, USA, August 20-23, 2013. Proceedings. In Bertoni, G., Coron, J.S., eds.: CHES. Volume 8086 of Lecture Notes in Computer Science., Springer (2013)
3. Biham, E., ed.: Fast Software Encryption, 4th International Workshop, FSE '97, Haifa, Israel, January 20-22, 1997, Proceedings. In Biham, E., ed.: FSE. Volume 1267 of Lecture Notes in Computer Science., Springer (1997)

[5] At the time of writing we haven't had access to an AVX2-enabled CPU yet, and the 256-bit version of `pshufb` is not available in the first version of AVX.

4. Biham, E., Shamir, A.: Differential Cryptanalysis of DES-like Cryptosystems. In Menezes, A., Vanstone, S.A., eds.: CRYPTO. Volume 537 of Lecture Notes in Computer Science., Springer (1990) 2–21
5. Bilgin, B., Bogdanov, A., Knezevic, M., Mendel, F., Wang, Q.: Fides: Lightweight Authenticated Cipher with Side-Channel Resistance for Constrained Hardware. [2] 142–158
6. Biryukov, A., Wagner, D.: Slide Attacks. [32] 245–259
7. Bogdanov, A., Knudsen, L.R., Leander, G., Paar, C., Poschmann, A., Robshaw, M.J.B., Seurin, Y., Vikkelsoe, C.: PRESENT: An Ultra-Lightweight Block Cipher. In Paillier, P., Verbauwhede, I., eds.: CHES. Volume 4727 of Lecture Notes in Computer Science., Springer (2007) 450–466
8. Boura, C., Canteaut, A., Cannière, C.D.: Higher-Order Differential Properties of Keccak and *Luffa*. In Joux, A., ed.: FSE. Volume 6733 of Lecture Notes in Computer Science., Springer (2011) 252–269
9. Boyar, J., Peralta, R.: A New Combinational Logic Minimization Technique with Applications to Cryptology. In Festa, P., ed.: SEA. Volume 6049 of Lecture Notes in Computer Science., Springer (2010) 178–189
10. Cid, C., Murphy, S., Robshaw, M.J.B.: Small Scale Variants of the AES. In Gilbert, H., Handschuh, H., eds.: FSE. Volume 3557 of Lecture Notes in Computer Science., Springer (2005) 145–162
11. Coron, J.S.: Higher Order Masking of Look-up Tables. Cryptology ePrint Archive, Report 2013/700 (2013) `http://eprint.iacr.org/2013/700`.
12. Daemen, J.: Limitations of the Even-Mansour Construction. [25] 495–498
13. Daemen, J., Knudsen, L.R., Rijmen, V.: The Block Cipher Square. [3] 149–165
14. Daemen, J., Peeters, M., Assche, G.V.: Bitslice Ciphers and Power Analysis Attacks. In Schneier, B., ed.: FSE. Volume 1978 of Lecture Notes in Computer Science., Springer (2000) 134–149
15. Daemen, J., Peeters, M., Assche, G.V., Rijmen, V.: Nessie proposal: NOEKEON (2000) Available online at `http://gro.noekeon.org/Noekeon-spec.pdf`.
16. Daemen, J., Rijmen, V.: The Wide Trail Design Strategy. In Honary, B., ed.: IMA Int. Conf. Volume 2260 of Lecture Notes in Computer Science., Springer (2001) 222–238
17. Dinur, I., Dunkelman, O., Keller, N., Shamir, A.: Key Recovery Attacks on 3-round Even-Mansour, 8-step LED-128, and Full AES^2. Cryptology ePrint Archive, Report 2013/391 (2013) `http://eprint.iacr.org/2013/391`.
18. Dunkelman, O., Keller, N., Shamir, A.: Minimalism in Cryptography: The Even-Mansour Scheme Revisited. In Pointcheval, D., Johansson, T., eds.: EUROCRYPT. Volume 7237 of Lecture Notes in Computer Science., Springer (2012) 336–354
19. Even, S., Mansour, Y.: A Construction of a Cipher From a Single Pseudorandom Permutation. [25] 210–224
20. Gérard, B., Grosso, V., Naya-Plasencia, M., Standaert, F.X.: Block Ciphers That Are Easier to Mask: How Far Can We Go? [2] 383–399
21. Grassl, M.: Bounds on the minimum distance of linear codes and quantum codes. Online available at `http://www.codetables.de` (2007) Accessed on 2013-11-08.
22. Grosso, V., Standaert, F.X., Faust, S.: Masking vs. Multiparty Computation: How Large Is the Gap for AES? [2] 400–416
23. Guo, J., Nikolic, I., Peyrin, T., Wang, L.: Cryptanalysis of Zorro. Cryptology ePrint Archive, Report 2013/713 (2013) `http://eprint.iacr.org/2013/713`.
24. Hart, P.E., Nilsson, N.J., Raphael, B.: A Formal Basis for the Heuristic Determination of Minimum Cost Paths. IEEE Trans. Systems Science and Cybernetics **4**(2) (1968) 100–107

25. Imai, H., Rivest, R.L., Matsumoto, T., eds.: Advances in Cryptology - ASIACRYPT '91, International Conference on the Theory and Applications of Cryptology, Fujiyoshida, Japan, November 11-14, 1991, Proceedings. In Imai, H., Rivest, R.L., Matsumoto, T., eds.: ASIACRYPT. Volume 739 of Lecture Notes in Computer Science., Springer (1993)
26. Ishai, Y., Sahai, A., Wagner, D.: Private Circuits: Securing Hardware against Probing Attacks. In Boneh, D., ed.: CRYPTO. Volume 2729 of Lecture Notes in Computer Science., Springer (2003) 463–481
27. Isobe, T., Shibutani, K.: Security Analysis of the Lightweight Block Ciphers XTEA, LED and Piccolo. In Susilo, W., Mu, Y., Seberry, J., eds.: ACISP. Volume 7372 of Lecture Notes in Computer Science., Springer (2012) 71–86
28. Käsper, E., Schwabe, P.: Faster and Timing-Attack Resistant AES-GCM. In Clavier, C., Gaj, K., eds.: CHES. Volume 5747 of Lecture Notes in Computer Science., Springer (2009) 1–17
29. Kerckhof, S., Durvaux, F., Hocquet, C., Bol, D., Standaert, F.X.: Towards Green Cryptography: A Comparison of Lightweight Ciphers from the Energy Viewpoint. In Prouff, E., Schaumont, P., eds.: CHES. Volume 7428 of Lecture Notes in Computer Science., Springer (2012) 390–407
30. Khovratovich, D., Nikolic, I.: Rotational Cryptanalysis of ARX. In Hong, S., Iwata, T., eds.: FSE. Volume 6147 of Lecture Notes in Computer Science., Springer (2010) 333–346
31. Kim, H., Hong, S., Lim, J.: A Fast and Provably Secure Higher-Order Masking of AES S-Box. [42] 95–107
32. Knudsen, L.R., ed.: Fast Software Encryption, 6th International Workshop, FSE '99, Rome, Italy, March 24-26, 1999, Proceedings. In Knudsen, L.R., ed.: FSE. Volume 1636 of Lecture Notes in Computer Science., Springer (1999)
33. Kwon, D., Kim, J., Park, S., Sung, S.H., Sohn, Y., Song, J.H., Yeom, Y., Yoon, E.J., Lee, S., Lee, J., Chee, S., Han, D., Hong, J.: New Block Cipher: ARIA. In Lim, J.I., Lee, D.H., eds.: ICISC. Volume 2971 of Lecture Notes in Computer Science., Springer (2003) 432–445
34. Matsui, M.: Linear Cryptanalysis Method for DES Cipher. In Helleseth, T., ed.: EUROCRYPT. Volume 765 of Lecture Notes in Computer Science., Springer (1993) 386–397
35. Matsui, M.: On Correlation Between the Order of S-boxes and the Strength of DES. [46] 366–375
36. Matsui, M.: New Block Encryption Algorithm MISTY. [3] 54–68
37. Mendel, F., Rijmen, V., Toz, D., Varici, K.: Differential Analysis of the LED Block Cipher. In Wang, X., Sako, K., eds.: ASIACRYPT. Volume 7658 of Lecture Notes in Computer Science., Springer (2012) 190–207
38. Nikolic, I., Wang, L., Wu, S.: Cryptanalysis of Round-Reduced LED. to appear in FSE (2013)
39. Nikova, S., Rijmen, V., Schläffer, M.: Secure Hardware Implementation of Nonlinear Functions in the Presence of Glitches. J. Cryptology **24**(2) (2011) 292–321
40. Nyberg, K.: Linear Approximation of Block Ciphers. [46] 439–444
41. Piret, G., Roche, T., Carlet, C.: PICARO - A Block Cipher Allowing Efficient Higher-Order Side-Channel Resistance. In Bao, F., Samarati, P., Zhou, J., eds.: ACNS. Volume 7341 of Lecture Notes in Computer Science., Springer (2012) 311–328
42. Preneel, B., Takagi, T., eds.: Cryptographic Hardware and Embedded Systems - CHES 2011 - 13th International Workshop, Nara, Japan, September 28 - October

1, 2011. Proceedings. In Preneel, B., Takagi, T., eds.: CHES. Volume 6917 of Lecture Notes in Computer Science., Springer (2011)
43. Prouff, E., Roche, T.: Higher-Order Glitches Free Implementation of the AES Using Secure Multi-party Computation Protocols. [42] 63–78
44. Rijmen, V., Barreto, P.: Nessie proposal: WHIRLPOOL (2000) Available online at `https://www.cosic.esat.kuleuven.be/nessie/`.
45. Rivain, M., Prouff, E.: Provably Secure Higher-Order Masking of AES. In Mangard, S., Standaert, F.X., eds.: CHES. Volume 6225 of Lecture Notes in Computer Science., Springer (2010) 413–427
46. Santis, A.D., ed.: Advances in Cryptology - EUROCRYPT '94, Workshop on the Theory and Application of Cryptographic Techniques, Perugia, Italy, May 9-12, 1994, Proceedings. In Santis, A.D., ed.: EUROCRYPT. Volume 950 of Lecture Notes in Computer Science., Springer (1995)
47. Toffoli, T.: Reversible Computing. In de Bakker, J.W., van Leeuwen, J., eds.: ICALP. Volume 85 of Lecture Notes in Computer Science., Springer (1980) 632–644
48. Ullrich, M., Cannière, C.D., Indesteege, S., Küçük, Ö., Mouha, N., Preneel, B.: Finding Optimal Bitsliced Implementations of 4 x 4-bit S-boxes. In: Symmetric Key Encryption Workshop, Copenhagen,DK (2011) 20
49. Veyrat-Charvillon, N., Gérard, B., Standaert, F.X.: Security Evaluations beyond Computing Power. In Johansson, T., Nguyen, P.Q., eds.: EUROCRYPT. Volume 7881 of Lecture Notes in Computer Science., Springer (2013) 126–141
50. Wagner, D.: The Boomerang Attack. [32] 156–170
51. Whitnall, C., Oswald, E., Standaert, F.X.: The Myth of Generic DPA...and the Magic of Learning. In Benaloh, J., ed.: CT-RSA. Volume 8366 of Lecture Notes in Computer Science., Springer (2014) 183–205

A S-boxes cryptanalytic properties

We consider the S-box $\mathsf{S} : \mathbb{F}_{2^n} \to \mathbb{F}_{2^n}$ as a vector of Boolean functions $\mathsf{S} = (f_0, \ldots, f_{n-1})$, $f_i : \mathbb{F}_{2^n} \to \mathbb{F}_2$. For $x \in \mathbb{F}_{2^n}$ and $u \in \mathbb{F}_2^n$, the notation x^u stands for the product $\prod_{i=0}^{n-1} x_i^{u_i}$, with the convention $0^0 = 1$. The cardinality of a set A, is denoted by $\#A$. For two elements $a, b \in \mathbb{F}_{2^n}$, we denote the dot product as: $a.b = \sum_{i=0}^{n-1} a_i b_i$. Hw(.) denotes the Hamming weight function.

Non-linearity. We use the Walsh transform and spectrum to evaluate the correlation of a linear approximation $(a, b) \neq (0, 0)$. Their definitions are given below.

Definition 1. *Walsh transform of a Boolean vector* S*:*

$$W_{\mathsf{S}}(a, b) = \sum_{x \in \mathbb{F}_{2^n}} (-1)^{a.x + b.\mathsf{S}(x)}.$$

Definition 2. *Walsh spectrum of a Boolean vector* S*:*

$$\Omega_{\mathsf{S}} = \{W_{\mathsf{S}}(a, b) | a, b \in \mathbb{F}_{2^n}, (a, b) \neq (0, 0)\}.$$

The smaller is the maximum of Ω_S, the stronger is the S-box regarding linear cryptanalysis [34]. In particular, a value $max(\Omega_{\mathsf{S}})$ for the Walsh spectrum of S implies that its best linear approximation has probability $\Pr_{lin} = \frac{max(\Omega_{\mathsf{S}})}{2^n}$.

Differential profile. We similarly use the differential spectrum to evaluate the resistance of an S-box against differential cryptanalysis [4].

Definition 3. *Differential spectrum of a Boolean vector S:*

$$\Delta_{\mathsf{S}} = \{\#\{X|\mathsf{S}(X+a) = \mathsf{S}(X) + b\}|a, b \in \mathbb{F}_{2^n}, (a,b) \neq (0,0)\}.$$

The smaller is the maximum of Δ_{S}, the strongest is the S-box regarding differential cryptanalysis. In particular, a value $max(\Delta_{\mathsf{S}})$ for the differential spectrum of S implies that its best differential has probability $\Pr_{diff} = \frac{max(\Delta_{\mathsf{S}})}{2^n}$.

Algebraic degree. Although the tools for analyzing algebraic attacks are not as advanced as for linear and differential ones, the algebraic degree is generally considered as a good indicator of security against them. Moreover, having a non-maximal algebraic degree allows distinguishing a function from a random one. For any Boolean function, the algebraic degree can be defined as follows.

Definition 4. *Algebraic degree of a boolean function f. A Boolean function f can be uniquely represented using its Algebraic Normal Form (ANF):*

$$f(x) = \sum_{u \in \mathbb{F}_2^n} a_u x^u.$$

The algebraic degree of f is defined as:

$$\deg(f) = \max_{u \in \mathbb{F}_2^n} \{\mathrm{Hw}(u), a_u \neq 0\}.$$

Definition 5. *Algebraic degree of a Boolean vector S. The algebraic degree of a vector is defined as the maximum degree of its coordinates:*

$$\deg(\mathsf{S}) = \max_{0 \leq i < n} \deg(f_i).$$

B Secure computation of non-linear operations

In this section, we use the notation $\in_R \mathbb{F}$ to mean that a value is chosen uniformly in $\mathbb{F}$ and $\cdot$ denotes the non-linear operation of $\mathbb{F}$, i.e. the field multiplication for extensions of $\mathbb{F}_2$ and the AND operator for vector spaces over $\mathbb{F}_2$.

Algorithm 2 Non linear operation performed on two masked secrets x and y

Require: Shares $(x_i)_i$ and $(y_i)_i$ satisfying $\oplus_i x_i = x$ and $\oplus_i y_i = y$.
Ensure: Shares $(w_i)_i$ satisfying $\oplus_i w_i = x \cdot y$.

```
for i from 0 to d do
    for j from i + 1 to d do
        r_{i,j} ∈_R F;
        r_{i,j} ← (r_{i,j} ⊕ x_i · y_j) ⊕ x_j · y_i;
for i from 0 to d do
    w_i ← x_i · y_i;
    for j from 0 to d, j ≠ i do
        w_i ← w_i ⊕ r_{i,j};
```