Analyzing Masked Ciphers Against Transition and Coupling Effects

Siemen Dhooghe

imec-COSIC, ESAT, KU Leuven, Belgium
name.lastname@esat.kuleuven.be

Abstract This paper discusses how to analyze the probing security of masked symmetric primitives against the leakage effects from CHES 2018; glitches, transitions, and coupling effects. This is illustrated on several architectures of ciphers like PRESENT, AES, and ASCON where we transform glitch-extended probing secure maskings into transition and/or coupling secure ones. The analysis uses linear cryptanalytic methods and the diffusion layers of the cipher to efficiently protect against the advanced leakage effects.

Keywords: Hardware · Linear Cryptanalysis · Masking · Robust Probing Security · Side-Channel Analysis

1 Introduction

From the moment a symmetric primitive is implemented on a physical device, it becomes susceptible to side-channel attacks. The most well-known attack in this line is differential power analysis where the power consumption of the device is correlated to its processed secrets [29]. Masking methods form a popular countermeasure against these attacks. Here each secret variable is split into multiple random shares. A masking method allows for algorithmic protection aiming to catch vulnerabilities before production. This algorithmic protection is based on security models trying to capture realistic attacks. The most popular model is the probing model originally proposed by Ishai *et al.* [27]. In the d^{th}-order and single-shot variant of this model, it is stated that any set of d intermediate values in the computation of a symmetric key primitive need to be independent of any secret value.

While the probing model is a good step towards finding reliable algorithmic countermeasures against side-channel attacks, it does not capture all realistic leakage effects in hardware. Faust *et al.* [22] formalizes some realistic effects which are not captured in the probing model and effectively extends the security model. The three effects discussed are glitches, transitions, and coupling effects. The extension of the formal probing model allows designers to find effective maskings to protect hardware implementations even against more advanced leakages.

A popular masking method is the one by Nikova, Rechberger, and Rijmen [33]. The method, known as a "threshold implementation", specifies properties the masked Boolean functions need to follow to secure a hardware implementation even in face of glitch effects. Originally, threshold implementations only protect against first-order side-channel attacks. However, a recent work provides a security analysis to analyze the higher-order security of threshold implementations [5]. This analysis relies on linear cryptanalytic methods which allow designers to analyze the masked primitive as a whole whereas previous analysis methods only analyze separate nonlinear gates or small parts of the primitive.

We currently succeed in protecting hardware maskings against glitches. However, it still remains an open problem how to efficiently protect against transition or coupling leakage and, more importantly, against a combination of leakage sources. While the work from Faust *et al.* provides a model how to capture these effects, it remains an open problem how to analyze the security of a masking in this new model.

Contributions This paper introduces an analysis technique to assess the transition and coupling-extended probing security defined by Faust *et al.* [22]. The analysis is based on the work by Beyne *et al.* [5] and extends it for the advanced leakage effects. In essence the method transforms a glitch-extended probing secure masking to a transition and coupling secure one as follows:

- Take a glitch-extended probing secure implementation of a symmetric primitive. For example, using threshold implementations [33] or a masking created using glitch-extended SNI or PINI secure gates [9, 22].
- Use the work by Faust *et al.* [22] and analyze the architecture of the masking to determine what an adversary can view. For example, via memory recombinations an adversary can view the output of two different masked S-boxes.
- Use the work by Beyne *et al.* [5] and determine whether there are trails between the observed values. In case zero-correlation approximations are found between the probed values, the countermeasure is deemed secure.

In this work we go over the theoretical analysis of several symmetric primitives and over various architectures a designer can use to implement the primitive. The primitives PRESENT, AES, and ASCON are taken as case studies. As similar primitives would have similar security arguments, these case studies represent a large class of primitives. The analysis is made as general as possible by considering black box masked S-boxes. Meaning that our analysis applies to *any* masking of the considered primitives as long as that masking is glitch-extended probing secure. For example, one can apply the analysis to the state of the art first-order masked architecture of the AES to make it secure against transition-extended probes using additional randomness.

We explicitly use the diffusion properties of the symmetric primitive to minimize the cost of protecting against transition or coupling leakage. We show that one typically requires only a total few random bits to protect a masking against

these effects. This should be compared to the current state of the art where there is an area and randomness overhead per shared multiplication like the work by Dhooghe and Nikova [19] and by Cassiers and Standaert [10, Table 2]. In particular, the work by Cassiers and Standaert [10, Table 2] requires several additional thousands of bits of randomness to secure a first-order glitch-resistant PRESENT against transition leakage. Instead, our analysis shows their glitch-resistant masking (for any of their proposed architectures) can be made secure against transition leakage using at most a single additional random bit.

2 Preliminaries

This work uses the tools of linear cryptanalysis to analyze the security of masking implementations following the threshold implementation model against probing adversaries which are extended to view advanced leakage such as glitches, transitions, and couplings. In this section, we recall the basics of hardware, the probing model, threshold implementations, and linear cryptanalysis over masked variables.

2.1 The Physical World

This section recalls the basics of hardware and side-channel attacks.

Synchronous Circuits A synchronous circuit consists of combinatorial gates (AND, XOR, etc.) and sequential logic (memory, registers). When the circuit is powered on, all registers, gates, and wires are powered too at which point they all carry a digital value. There is a clock synchronizing the operations of different circuit elements. A clock cycle is the time between two clock ticks. During each clock cycle the combinatorial logic is re-evaluated and results are stored in the registers.

Registers A register (or memory cell) has one input and one output and its functionality is controlled by the clock. Registers release a signal by opening its "out-line" while the "input-line" is closed (only one is open at a time). The register out-line stays open until the signal in the logic becomes stable, after that it stores the newly computed value - hence the register closes the output-line and opens the input-line.

Logical Gates Logical gates perform simple Boolean operations. They have several wires as input and a single wire as output. Each gate can have a different time to propagate a signal from its inputs to its output and each gate can have a different power consumption. A change of its inputs causes re-evaluation of the gate and hence may change the output value.

2.2 The Bounded-Query Probing Model

This section recalls the bounded-query probing model, its expansion considering the effect of glitches, its security analysis, and a note on key schedules. Later on, the probing model is further expanded to capture transition and coupling effects.

Threshold Probing A d^{th}-order probing adversary $\mathcal{A}$, as first proposed by Ishai *et al.* [27], can view up to d gates or wires in a circuit per query. This circuit encodes an operation, such as a cipher call, and consists of gates, such as AND or XOR gates, and wires. The adversary $\mathcal{A}$ is computationally unbounded, and must specify the location of the probes before querying the circuit. However, the adversary can change the location of the probes over multiple circuit queries. The adversary's interaction with the circuit is mediated through encoder and decoder algorithms, neither of which can be probed.

In the bounded query model, the security of a circuit C with input k against a d^{th}-order probing adversary is quantified by means of the left-or-right security game. The challenger picks a random bit b and provides an oracle $\mathcal{O}^b$, to which adversary $\mathcal{A}$ is given query access. The adversary queries the oracle by choosing up to d wires to probe – we denote this set of probe positions by $\mathcal{P}$ – and sends it to the oracle along with chosen inputs k_0 and k_1. The oracle responds with the probed wire values of $C(k_b)$. After a total of q queries, the adversary responds to the challenger with a guess for b. For $b \in \{0, 1\}$, denote the result of the adversary after interacting with the oracle $\mathcal{O}^b$ using q queries by $\mathcal{A}^{\mathcal{O}^b}$. The left-or-right advantage of the adversary $\mathcal{A}$ is then as defined as

$$\text{Adv}_{\text{-thr}}(\mathcal{A}) = |\Pr[\mathcal{A}^{\mathcal{O}^0} = 1] - \Pr[\mathcal{A}^{\mathcal{O}^1} = 1]|\,.$$

Modeling Glitches Let us consider "basic" combinatorial logic, namely the logic which connects two layers of registers. When a cycle starts, the output signals of the registers are inputs for the basic logic and these signals will start propagating through the wires and the gates until they reach the output registers. Gate evaluation may happen several times until the signals (and hence the gate) become stable. This can be due to many reasons, we list three of them: a) the wire signals propagate with different speed; b) the wires have different length; and c) each gate has different propagation time. We will refer to these value changes on the wires and gates as glitches.

In a cycle there are two main phases. The first phase is one in which the wires and gates do not have a stable value. This phase is followed by one in which all values are stable. The power consumption at a time sample is the sum of the power consumption of the wires, gates, and registers belonging to this simple logic. For CMOS technologies, the power consumption during the first phase is higher and more apt to change because of glitches compared to the second phase. We stress that glitches occur in the logic between two memory gates and are stopped by registers. In other words, glitches do not propagate through memory gates.

Glitches can result in significant leakage that is not accounted for by the standard probing model, see for example the attacks of Mangard *et al.* on several masked AES implementations [31]. Consequently, it is necessary to extend the capabilities of threshold probing adversaries in order to capture the physical effect of glitches on a hardware platform. Whereas one of the adversary's probes normally results in the value of a single wire, a glitch-extended probe allows obtaining the values of all wires in a bundle. This extension of the probing model has been discussed by several authors, here we give the definition from Faust *et al.* [22] who describes it as follows:

"**Specific model for glitches.** For any ϵ-input circuit gadget G, combinatorial recombinations (aka glitches) can be modeled with specifically ϵ-extended probes so that probing any output of the function allows the adversary to observe all its ϵ inputs."

Security Analysis The main theoretical result of [5] is that the bounded-query probing security of a masked cipher can be related to its linear cryptanalysis. The first step towards this result is provided by Theorem 1 below, which relates the security of the masked cipher to the Fourier transform of the probability distribution of wire values obtained by probing. The link with linear cryptanalysis will be developed in detail in Section 2.4.

The Fourier transform of a function $V \to \mathbb{C}$, where V is a subspace of $\mathbb{F}_2^n$, can be defined as in Definition 1 below. For the purposes of this section, only probability mass functions on $\mathbb{F}_2^n$ need be considered. Despite this, Definition 1 considers more general functions on an arbitrary subspace $V \subseteq \mathbb{F}_2^n$. Since any vector space over $\mathbb{F}_2$ is isomorphic to $\mathbb{F}_2^n$ for some n, this generalization is mostly a matter of notation. Nevertheless, this extended notation will be convenient in Section 2.4.

Definition 1 ([5], §2.1). *Let $V \subseteq \mathbb{F}_2^n$ be a vector space and $f : V \to \mathbb{C}$ a complex-valued function on V. The Fourier transformation of f is a function $\widehat{f} : \mathbb{F}_2^n/V^\perp \to \mathbb{C}$ defined by*

$$\widehat{f}(u) = \sum_{x \in V} (-1)^{u^\top x} f(x),$$

where we write u for $u + V^\perp$. Equivalently, $\widehat{f}$ is the representation of f in the basis of functions $x \mapsto (-1)^{u^\top x}$ for $u \in \mathbb{F}_2^n/V^\perp$.

Recall that the orthogonal complement $V^\perp$ of a subspace V of $\mathbb{F}_2^n$ is the vector space $V^\perp = \{x \in \mathbb{F}_2^n \mid \forall v \in V : v^\top x = 0\}$. The quotient space $\mathbb{F}_2^n/V^\perp$ is by definition the vector space of cosets of $V^\perp$. For convenience, an element $x + V^\perp \in \mathbb{F}_2^n/V^\perp$ will simply be denoted by x. For $x \in \mathbb{F}_2^n/V^\perp$ and $v \in V$, the expression $x^\top v$ is well-defined. Consequently, the above definition is proper.

The main theorem on the advantage of an adversary in the bounded-query probing model can now be stated. It relies on the observation that, for a bounded-query probing secure circuit, all probed wire values either closely resemble uniform randomness or reveal nothing about the secret input.

Theorem 1 ([5], §4). *Let $\mathcal{A}$ be a t-threshold-probing adversary for a circuit C. Assume that for every query made by $\mathcal{A}$ on the oracle $\mathcal{O}^b$, there exists a partitioning (depending only on the probe positions) of the resulting wire values into two random variables $\mathbf{x}$ ('good') and $\mathbf{y}$ ('bad') such that*

1. *The conditional probability distribution $p_{\mathbf{y}|\mathbf{x}}$ satisfies $\mathbb{E}_{\mathbf{x}}\|\widehat{p}_{\mathbf{y}|\mathbf{x}} - \delta_0\|_2^2 \leq \varepsilon$ with δ_0 the Kronecker delta function,*
2. *Any t-threshold-probing adversary for the same circuit C and making the same oracle queries as $\mathcal{A}$, but which only receives the 'good' wire values (*i.e. *corresponding to $\mathbf{x}$) for each query, has advantage zero.*

The advantage of $\mathcal{A}$ can be upper bounded as

$$\mathrm{Adv}_{t\text{-thr}}(\mathcal{A}) \leq \sqrt{2\,q\,\varepsilon}\,,$$

where q is the number of queries to the oracle $\mathcal{O}^b$.

This work only considers a 1-threshold-probing adversary, but extends the probing model such that one probe can provide multiple shares even over different rounds. Furthermore, we consider the effect of transitions and couplings which typically provide shares over two consecutive rounds. As a result, we use the above theorem only for the 'bad' values. Moreover, in this work we are only interested to find out whether the 2-norm $\|\widehat{p}_{\mathbf{z}} - \delta_0\|_2$ is zero or not. As potential trails have to be short, the correlation is bound to be high. Thus, we are interested to see whether the diffusion layers of a masked cipher allow for zero-correlation approximations.

Key Schedule This work focuses on the state function of a cipher and considers the (masked) key to be constant. This focus is based on two reasons.

- To create security arguments independent of the used mode of operation. Since in some modes the key input can be public, one cannot rely on entropy coming from the key schedule.
- In practice, the masked key of a block cipher is not frequently re-masked with fresh randomness. Over several queries, the masked key is thus without fresh entropy.

The key is thus labeled a 'good' variable. Depending on the use case, the designer can nevertheless opt to include the key schedule for a more in-depth analysis.

2.3 Boolean Masking and Threshold Implementations

In this section, we recall Boolean masking and threshold implementations as countermeasures against side-channel analysis. We specifically recall threshold implementations since we require the maskings of the S-box in our case studies to be uniform and we need the property of non-completeness to protect against glitches.

Boolean masking, as originally proposed by Goubin and Patarin [23] and Chari *et al.* [11], has become a popular countermeasure against side-channel analysis. Intuitively, each sensitive variable is split in multiple pieces such that the adversary is forced to recombine those, exponentially increasing the noise on the data in the number of pieces. Formally, a secret sharing scheme is used. For Boolean masking, each secret x is split in the variables $\bar{x} = (x^1, x^2, \ldots, x^{s_x})$ such that $x = \sum_{i=1}^{s_x} x^i$ where the sum is taken over a binary finite field K. We call a random Boolean masking of a fixed secret uniform if all sharings of that secret are equally likely.

A masking countermeasure shares each intermediate variable of a primitive such that at no point in time a secret value is directly processed. There are several methods how to achieve this given in the literature. In this work, we focus on the method of threshold implementations as introduced by Nikova *et al.* [33]. A threshold implementation consists of several layers of Boolean functions. Each layer is calculated in one clock cycle and stores its output in registers.

Let $\bar{F}$ be a layer in the threshold implementation corresponding to a part of the circuit $F : \mathbb{F}_2^n \to \mathbb{F}_2^m$. For example, F might be the linear layer of a block cipher. The function $\bar{F} : \mathbb{F}_2^{ns_x} \to \mathbb{F}_2^{ms_y}$, where we assume s_x shares per input bit and s_y shares per output bit, will be called a *sharing* of F. A share of a function is denoted by $F^i : \mathbb{F}_2^{ns_x} \to \mathbb{F}_2^m$, for $i \in \{1, .., s_y\}$. The main properties of threshold implementations are summarized in Definition 2.

Definition 2 (Properties of a threshold implementation [33]). *Let $F : \mathbb{F}_2^n \to \mathbb{F}_2^m$ be a function and $\bar{F} : \mathbb{F}_2^{ns_x} \to \mathbb{F}_2^{ms_y}$ a sharing of F. The sharing $\bar{F}$ is said to be*

1. correct *if $\sum_{i=1}^{s_y} F^i(x^1, \ldots, x^{s_x}) = F(x)$ for all $x \in \mathbb{F}_2^n$ and for all shares $x^1, \ldots, x^{s_x} \in \mathbb{F}_2^n$ such that $\sum_{i=1}^{s_x} x^i = x$,*
2. non-complete *if any component function F^i depends on at most $s_x - 1$ input shares,*
3. uniform *if $\bar{F}$ maps a uniform random sharing of any $x \in \mathbb{F}_2^n$ to a uniform random sharing of $F(x) \in \mathbb{F}_2^m$.*

Recall glitch-extended probes as introduced in Section 2.2. Since each component function in a threshold implementation works on a non-complete set of shares and since each function is walled-off by registers, a threshold implementation is secure even in face of glitch effects. The glitch-extended probing security of a threshold implementation has been formally proven by Dhooghe *et al.* [20]. Also we recall from [20, Section 4] that every SNI secure gadget is also uniform. Thus, the secure analysis requirement for a uniform masked S-box is achieved by most maskings in the literature.

2.4 Linear Cryptanalysis of Threshold Implementations

As discussed in Section 2.2, Theorem 1 allows proving the security of higher-order threshold implementations given an upper bound on the Fourier coefficients of

probability distributions of wire values obtained by probing. This section shows how such an upper bound can be obtained using linear cryptanalysis.

For any linear masking scheme, there exists a vector space $\mathbb{V} \subset \mathbb{F}_2^\ell$ of valid sharings of zero. More specifically, an $\mathbb{F}_2$-linear secret sharing scheme is an algorithm that maps a secret $x \in \mathbb{F}_2^n$ to a random element of a corresponding coset of the vector space $\mathbb{V}$. Let $\rho : \mathbb{F}_2^n \to \mathbb{F}_2^\ell$ be a map that sends secrets to their corresponding coset representative. For convenience, we denote $\mathbb{V}_a = a + \mathbb{V}$.

Let $\bar{G}$ be a correct sharing of a function $G : \mathbb{F}_2^n \to \mathbb{F}_2^n$ in the sense of Definition 2. Fix any $x \in \mathbb{F}_2^n$ and let $a = \rho(x)$ and $b = \rho(G(x))$. The correctness property implies that $\bar{G}(\mathbb{V}_a) \subseteq \mathbb{V}_b$. It follows that the restriction $F : \mathbb{V}_a \to \mathbb{V}_b$ of $\bar{G}$ defined by $F(x) = \bar{G}(x)$ is a well defined function.

Linear cryptanalysis is closely related to the propagation of the Fourier transformation of a probability distribution under a function $F : \mathbb{V}_a \to \mathbb{V}_b$. This leads to the notion of correlation matrices due to Daemen *et al.* [13]. The action of F on probability distributions can be described by a linear operator. The coordinate representation of this operator with respect to the standard basis $\{\delta_x\}_{x \in V}$ may be called the *transition matrix* of F. Following [4], the correlation matrix of F is then the same operator expressed with respect to the Fourier basis. The correlation matrix of a sharing can be defined as follows. Note that it only depends on the spaces $\mathbb{V}_a$ and $\mathbb{V}_b$, not on the specific choice of the representatives a and b.

Definition 3 (Correlation matrix). *For a subspace $\mathbb{V} \subseteq \mathbb{F}_2^\ell$, let $F : \mathbb{V}_a \to \mathbb{V}_b$ be a function. The correlation matrix C^F of F is a real $|\mathbb{V}_b| \times |\mathbb{V}_a|$ matrix with coordinates indexed by elements $u, v \in \mathbb{F}_2^n/\mathbb{V}^\perp$ and equal to*

$$C^F_{v,u} = \frac{1}{|\mathbb{V}|} \sum_{x \in \mathbb{V}_a} (-1)^{u^\top x + v^\top F(x)} .$$

The relation between Definition 3 and linear cryptanalysis is as follows: the coordinate $C^F_{v,u}$ is equal to the correlation of a linear approximation over F with input mask u and output mask v. That is, $C^F_{v,u} = 2\Pr[v^\top F(\mathbf{x}) = u^\top \mathbf{x}] - 1$ for $\mathbf{x}$ uniform random on $\mathbb{V}_a$. An important difference with ordinary linear cryptanalysis is that, for shared functions, the masks u and v correspond to equivalence classes. This formalizes the intuitive observation that masks which differ by a vector orthogonal to the space $\mathbb{V}$ lead to identical correlations.

From this point on, we restrict to second-order probing adversaries. The description of the link with linear cryptanalysis presented in [5], is completed by Theorem 2 below. It shows that the coordinates of $\widehat{p}_{\mathbf{z}}$ are entries of the correlation matrix of the state-transformation between the specified probe locations. In Theorem 2, the restriction of $x \in \mathbb{V}_a$ to an index set $I = \{i_1, \ldots, i_m\}$ is denoted by $x_I = (x_{i_1}, \ldots, x_{i_m}) \in \mathbb{F}_2^{|I|}$. This definition depends on the specific choice of the representative a, but the result of Theorem 2 does not.

Theorem 2 ([5], §5.2). *Let $F : \mathbb{V}_a \to \mathbb{V}_b$ be a function with $\mathbb{V} \subset \mathbb{F}_2^\ell$ and $I, J \subset \{1, \ldots, \ell\}$. For $\mathbf{x}$ uniform random on $\mathbb{V}_a$ and $\mathbf{y} = F(\mathbf{x})$, let $\mathbf{z} = (\mathbf{x}_I, \mathbf{y}_J)$.*

The Fourier transformation of the probability mass function of $\mathbf{z}$ *then satisfies*

$$|\widehat{p}_{\mathbf{z}}(u,v)| = |C^F_{\widetilde{v},\widetilde{u}}|,$$

where $\widetilde{u},\widetilde{v} \in \mathbb{F}_2^\ell/\mathbb{V}^\perp$ *are such that* $\widetilde{u}_I = u$, $\widetilde{u}_{[\ell]\setminus I} = 0$, $\widetilde{v}_J = v$ *and* $\widetilde{v}_{[\ell]\setminus J} = 0$.

Theorem 2 relates the linear approximations of F to $\widehat{p}_{\mathbf{z}}(u)$ and hence provides a method to upper bound $\|\widehat{p}_{\mathbf{z}} - \delta_0\|_2$ based on linear cryptanalysis. Upper bounding the absolute correlations $|C^F_{\widetilde{v},\widetilde{u}}|$ is nontrivial in general. However, the piling-up principle [32,35] can be used to obtain heuristic estimates.

Importantly, Theorem 2 relates to linear cryptanalysis with respect to $\mathbb{V}$ rather than $\mathbb{F}_2^\ell$. The differences are mostly minor, but there is a subtle difference in relation to the important notion of 'activity'. In standard linear cryptanalysis, an S-box is said to be active if its output mask is nonzero. The same definition applies for linear cryptanalysis with respect to $\mathbb{V}$, but one must take into account that the mask is now an element of the quotient space $\mathbb{F}_2^\ell/\mathbb{V}^\perp$. In particular, if the mask corresponding to the shares of a particular bit can be represented by an all-one vector $(1,1,\ldots,1)^\top$, it may be equivalently represented by the zero vector. It is still true that a valid linear approximation for a permutation must have either both input masks equivalent to zero or neither equivalent to zero. More generally, this condition is ensured by any uniform sharing.

3 Analyzing Transition Leakage

This section studies the effect of transition leakage. Consider registers as recalled in Section 2.1. When the register input is open to store the incoming value, the new value has to overwrite the so far stored value. If these are different values then the attacker can measure a peak in the power consumption compared to the case when the values are the same – this is called transition leakage. Similar to registers, if a new value different from the wire current value starts propagating through a wire then the power consumption will differ compared to the case when the new and the current value are the same.

As a result, if in a memory cell the element x is erased and instead y is stored, transitions can leak both x and y. We integrate such leakage effects in the probing model following the work by Faust *et al.* [22]. There the model is described as follows.

"**Specific model for transitions.** For a memory cell m, memory recombinations (aka transitions) can be modeled with specifically 2-extended probes so that probing m allows the adversary to observe any pair of values stored in 2 of its consecutive invocations."

As mentioned above, transition leakage does not only occur in memory elements. Thus, we extend the description such that the extended leakage is viewed in any gate or wire.

The work by Ishai *et al.* [27] and the publications that followed considered a circuit model that represents a deterministic circuit as a directed acyclic graph whose vertices are combinatorial gates and its edges are wires carrying elements

from a finite field. However such a simplification of the circuit model does not take into account the circuit topology. While the leakage of glitches does not depend on the circuit architecture/topology, for the transitions and the wire coupling models the leakage is mainly influenced by the circuit's architecture. As a result, we study circuits with loops and a notion of time and consider particular architectures when discussing the side-channel security of symmetric primitives.

Finally, we consider the combined effect of glitches and transition leakage. According to Faust *et al.* [22] probing a memory gate is equivalent to probe the sole input (i.e. the wire) to it which can be considered also as output of the particular simple logic which ends up with the considered memory gate. In that regard, leakage caused by glitches might seem stronger than the leakage caused by transitions. However, in practice the two leakages can manifest differently in the time interval representing a single cycle, namely the glitches and the logic transitions will occur in the beginning while the memory transition will occur only at the end. As illustrated by Faust *et al.*, there could be a time window between the computational and the storage phases.

We apply the extended probing model to arbitrary glitch-extended probing secure masking of several architectures of PRESENT, AES, and ASCON to investigate how to best protect the masked primitives. In our analysis we consider a black box masking of the S-box to make the analysis more general. We just assume the linear layers are masked share-wise and that the masked S-boxes do not share inputs such as recycled randomness.

3.1 PRESENT

We recall the PRESENT cipher from the work of Bogdanov *et al.* [7]. The input to PRESENT is a 64-bit plaintext m. Each round comprises an XOR with the round key, a substitution layer, and a permutation layer. The substitution layer consists of 16 applications of a four-bit cubic S-box. The permutation layer of PRESENT is a bit permutation which is depicted in Figure 1. The following arguments are also applicable to the GIFT cipher [1].

Round-Based Architectures Require Extra Protection We first consider a glitch-extended probing secure masking (such as a threshold implementation) in a round-based architecture. In such an architecture each masked S-box in the round function is implemented separately on the platform. When an adversary places a transition-extended probe in an S-box of a round-based architecture, it can view the computation of the same implemented S-box in two consecutive rounds. In this architecture, the diffusion layer of PRESENT allows for some weak points concerning transition leakage. More specifically, bits $0, 21, 42$ and 63 are mapped to the same position. As a result, transition leakage from the $0, 5, 10$ or 15^{th} masked S-box could reveal information. An example activity pattern is indicated in dotted blue in Figure 1. The weakness can be resolved by re-masking the bits which remain fixed through the diffusion layer. This randomness can be re-used every round and can be the same over the bits $0, 21, 42$ and 63. For

the previously mentioned sharing of the S-box this countermeasure would cost a total of two random bits for the entire masked cipher.

So far we assumed the shared S-box was implemented as a rolled-out circuit on the platform. However, the typical threshold implementation of the PRESENT S-box consists of two degree two maps $\bar{G}$, this sharing is given in Appendix A. A designer could implement $\bar{G}$ and evaluate it twice to compute an S-box. We denote the two parts of the shared S-box by $\bar{S}_1$ and $\bar{S}_2$. If an adversary uses a transition-extended probe on the shared S-box it can view the computation of both $\bar{S}_1$ and $\bar{S}_2$. This is depicted in blue in Figure 2.

We give an example that the above explained weakness can constitute a probing attack for a particular sharing of the S-box. In particular, we consider the blue activity pattern from Figure 2. Considering the sharing of the S-box from Appendix A. Consider, for a secret x, the shares x^1, x^2, x^3 such that $\sum_{i=1}^{3} x^i = x$. An adversary placing a glitch and transition-extended probe $\mathcal{P}$ in the first component of $\bar{S}_1(\bar{x}, \bar{y}, \bar{z}\bar{w})$ is given back the shares $x^1, y^1, y^2, z^1, z^2, w^1$, and w^2. However, from the second part of the shared S-box $\bar{S}_2$ the adversary views the input values (equivalently $\bar{S}_1$'s output values)

$$\bar{S}_1^1 = w^1 + x^1y^1 + x^1y^2 + x^2y^1\,,$$
$$\bar{S}_1^2 = w^2 + x^2y^2 + x^2y^3 + x^3y^2\,.$$

Thus, $\mathcal{P} = \{\bar{S}_1^1, \bar{S}_1^2, x^1, y^1, y^2, z^1, z^2, w^1, w^2\}$. Consider random secrets x, y, z, w, then we find that

$$I(x; \mathcal{P}) = H(x) - H(x|\mathcal{P}) \neq 0,$$

with I the mutual information and H the Shannon entropy. In other words, a glitch- and transition-extended probing adversary can break the design.

As a conclusion, a straightforward round-based architecture of PRESENT could be vulnerable to transition leakage and thus extra costs would be required to secure the architecture.

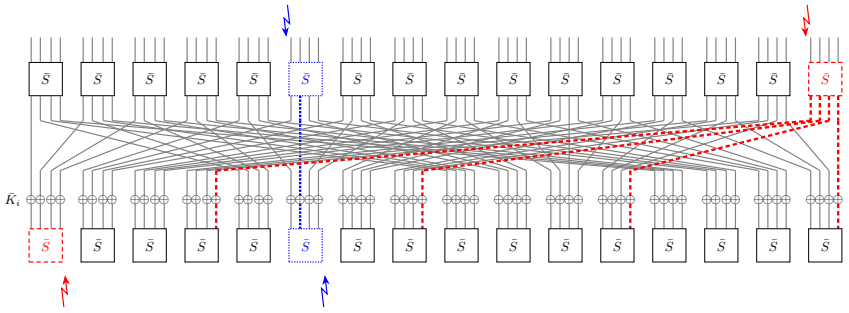

Figure 1: An activity pattern caused by transition leakage in PRESENT. In dotted blue lines, we find harmful transition leakage using a round-based architecture. In dashed red lines, we find perfect security using a serial architecture.

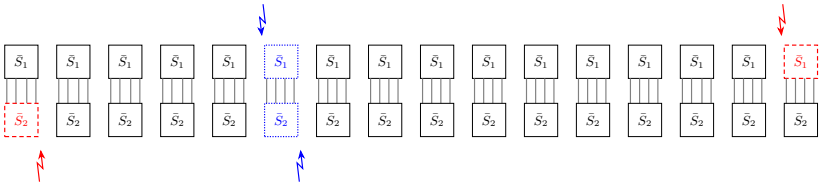

Figure 2: Transition leakage in the calculation of the shared S-box divided in $\bar{S}_1$ and $\bar{S}_2$. The dotted blue lines denote leakage in a round-based architecture. The dashed red lines denote leakage in a serial architecture.

Serial Architectures are Secure As a second example, we consider a glitch-extended probing secure masking in an S-box serial architecture. In this architecture only one shared S-box is implemented on the platform and each S-box in the cipher is computed in series. Following Figure 1, the architecture computes the S-boxes from left to right. Transition leakage from such a design occurs either between two S-boxes in the same round or between the last and first S-box of two consecutive rounds. Assuming the shared S-boxes are uniform, the former case never constitutes a weakness. The latter case is harmless thanks to the linear layer of PRESENT as indicated in red in Figure 1. We see, no matter which input or output mask is chosen, the resulting hull will always be a zero-correlation linear approximation. Thus, no extra precautions need to be made to ensure security against transition leakage.

In case the shared S-box is calculated over two cycles using the same resources, the architecture can be secured against transition leakage if all $\bar{S}_1$ are first calculated from left to right and before calculating the $\bar{S}_2$. A transition-extended probe either only views two separate $\bar{S}_1$ (similarly $\bar{S}_2$) in the same cycle or it views the parts as depicted in red in Figure 2. It is clear that in no case the adversary can break the masking using transition leakage.

Considering architectures of a masked PRESENT in light of transition leakage, we find that glitch-extended secure serial architectures provide protection against transition leakage without requiring additional costs.

3.2 AES

We quickly recall the standardized AES cipher by Daemen and Rijmen [14]. AES-128 consists of a 128-bit state and 128-bit key divided into 16 bytes. The cipher is composed of 10 rounds each applying an addition of a subkey, a bricklayer of S-Boxes, a `ShiftRows` operation, and a `MixColumns` operation. There are many primitives using a similar diffusion layer such as LED [26], PHOTON [25], PRINCE [8], SKINNY [2], etc. The security analysis considered here applies to all these primitives.

Every Architecture Requires Extra Protection For a glitch-extended probing secure AES, we find that architectures are vulnerable against transition leakage in the application of the `MixColumns` operation. First, notice that no matter the

architecture, transition leakage from the computation of an S-box is never usable for an adversary due to the branch number of the `MixColumns`. An example of such an activity pattern is depicted in Figure 3. However, during the computation of the `MixColumns` and due to the effect of glitches, the adversary can view the input of the operation and place a mask such that only one output byte is active. This active byte can only shift over the state, causing the adversary to also observe the computation of the following `MixColumns` due to transition leakages. An activity pattern for a round-based architecture is shown in Figure 4. Even by changing the order of the `MixColumns` harmful activity patterns can always be found.

One can prevent this leakage by re-masking some cells (for example, the top row for the round-based architecture) of the AES state. This randomness cost is low as the same randomness can be used for every round and for every cell. More specifically, considering a two-shared threshold implementation, a total of eight random bits are needed to prevent any transition leakage from occurring.

Another countermeasure is based on the verification of the Linear Approximation Table (LAT) of the shared AES S-box using the definitions from Section 2.4. More specifically, note that a masking of a linear operation works sharewise. As a result, a transition-extended probe on the AES diffusion layer only views one share each of two `MixColumns` operations and thus one input and one output share of the shared S-box. Given an s-sharing of the AES S-box $\bar{S}$ and $\mathbb{V} = \bigoplus_{i=1}^{s} \mathbb{V}^i$ such that each mask $u \in \mathbb{F}_2^{8s}/\mathbb{V}^\perp$ can be decomposed in $(u^1, ..., u^s)$ with $u^i \in \mathbb{F}_2^8/(\mathbb{V}^i)^\perp$ then the linear approximation between the observed values is only harmful if

$$\forall \alpha, \beta \in \mathbb{F}_2^{8s}/\mathbb{V}^\perp : \; C^{\bar{S}}_{\alpha,\beta} = 0 \text{ when } \mathrm{wt}(\alpha) = 1, \mathrm{wt}(\beta) = 1 \,.$$

In words, the sharing is secure if the masked S-box has nontrivial diffusion between the shares. A sharing using randomness, for example the sharing of the AES S-box by De Cnudde *et al.* [17], typically has this property. Currently there exists no uniform sharing of the AES S-box and thus every AES S-box sharing (except those using changing of the guards as explained later) uses randomness. However, applying the above property to sharings of primitives such as LED or PRINCE could be interesting future work.

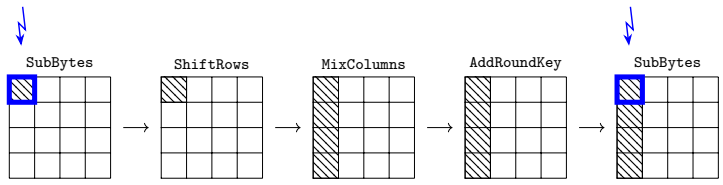

Figure 3: Activity patterns between the top left S-boxes of rounds i and $i+1$ of a masked AES. The figure denotes in blue what the adversary can observe and hatched cells denote active cells.

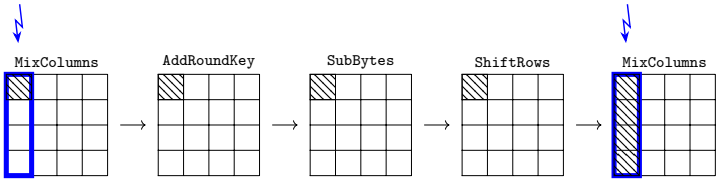

Figure 4: An activity pattern caused by transition leakage following the AES diffusion layers. The figure denotes in blue what the adversary can observe and hatched cells denote active cells.

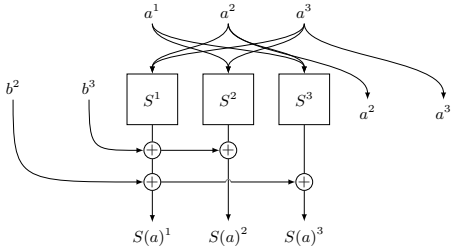

Figure 5: The "Changing of the Guards" method to make a sharing uniform.

Using Changing of the Guards Currently, the AES S-box has no known uniform sharing. As a result, designers typically use the changing of the guards technique by Daemen [12] to ensure uniformity. The technique adds input shares of one S-box to the output of another in order to embed the sharing in a permutation similar to the Feistel construction. The method is depicted in Figure 5. Since the technique chains S-boxes, extra diffusion is added to the round function. A depiction of such an example is given in the left picture of Figure 6. The figure shows which inputs (start of an arrow) are used to "re-mask" a shared S-box (end of an arrow). When this technique is used with care, one might use it to strengthen the masking against transition leakage. For example, while the diffusion following the left picture still allows for harmful transition leakage in a round-based architecture, the diffusion following the second picture prevents the adversary from learning a secret variable. This is due to the pattern ensuring that each active cell in the state activates at least one different column after the application of the `SubBytes` and `ShiftRows` operations. The third picture of Figure 6 depicts which column each cell activates after the `SubBytes` and `ShiftRows` operations.

We note that the second pattern shown in Figure 6 is not unique. It is also assumed the changing of the guards technique is only applied once per round. If it is applied multiple times (such as in the work by Wegener *et al.* [36] or by

Sugawara [34]), different patterns should be used to secure the cipher against transition leakage.

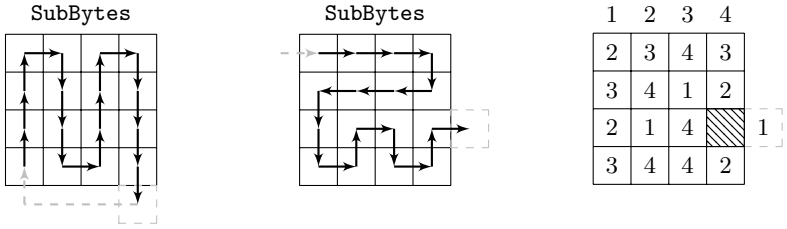

Figure 6: Two example diffusion patterns using the changing of the guards technique. In gray we denote the added extra cells. The left pattern combined with the AES diffusion layers is vulnerable to transition leakage, the right provides resistance. The third figure shows the activation of columns of the second pattern combined with `ShiftRows`.

3.3 ASCON

ASCON [21] consists of a mode of operation which uses a specific permutation. In this work, we focus on the permutation. The substitution layer is the parallel column-wise application of 64 5-bit S-boxes which are an affine transformation of the χ mapping of Keccak [3]. These S-boxes have linear branch number three. The linear layer consists of five row-wise applications of a linear function Σ. Each Σ function has linear branch number four and uses different rotation values depending on the row to optimize diffusion over several rounds.

Every Architecture is Secure Consider again an arbitrary glitch-extended probing secure masking of ASCON. We investigate whether we can transform this masking to be transition-extended probing secure. First, while the S-box of ASCON has a nontrivial linear branch number, its sharing might not have a similar property. Denoting the ASCON S-box by S. While S has a correlation zero transition between its first input bit x and first output bit z, the sharing of S can allow for such a transition. Meaning that one can place a nontrivial mask on the sharing of x and on the sharing of z and still find nonzero correlation. The sharing of the S-box given in Appendix B has been verified to still have a nontrivial linear branch number over the bits. More specifically, given that $\mathbb{V} = \bigoplus_{i=1}^{5} \mathbb{V}_i$ such that each mask $u \in \mathbb{F}_2^{20}/\mathbb{V}^\perp$ can be decomposed in $(u_1, u_2, u_4, u_4, u_5)$ with $u_i \in \mathbb{F}_2^4/(\mathbb{V}_i)^\perp$ then

$$\min_{\alpha,\beta \in \mathbb{F}_2^{20}/\mathbb{V}^\perp, C^{\bar{S}}_{\alpha,\beta} \neq 0} \{\mathrm{wt}(\alpha) + \mathrm{wt}(\beta)\} = 3\,.$$

By adding linear correction terms, we can cycle through other non-complete sharings of the ASCON S-box. Via this search method we found sharings which were uniform but did not attain the above property.

In case the sharing of the ASCON S-box has a nontrivial branch number, a round-based implementation is automatically secure against any harmful transition leakage. A typical simplified activity pattern is shown in Figure 7. Moreover, the same applies for other architectures like a bit-serial implementation. In other words, except for some trivial share-serial approaches, the round function of ASCON requires no additional care to prevent any harmful transition leakage.

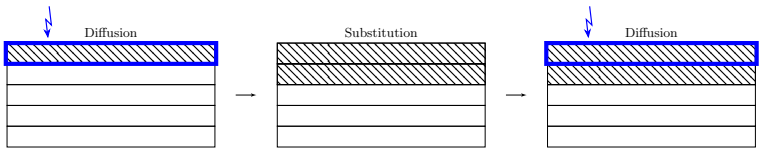

Figure 7: A simplified activity pattern in ASCON caused by a transition-extended probe.

4 Analyzing Coupling Leakage

We consider a leakage effect originating from coupling capacitors between circuit wires, and between circuit wires and ground which are influenced by the switching activity on that wire causing recombinations of the wire values. Effectively when observing leakage from one wire, one can observe leakage from nearby coupled wires. De Cnudde *et al.* [15] has shown that the security of masked hardware implementations can be affected due to coupling effects. These effects are integrated into the probing model following the work by Faust *et al.* [22].

"**Specific model for couplings.** For any set of adjacent wires $\{w_1, ..., w_d\}$, routing recombinations (aka couplings) can be modeled with c-extended probes so that probing one wire w_i allows the adversary to observe c wires adjacent to w_i."

Defending against the above model is not straightforward as two wires carrying different shares of a secret can be coupled. The work by De Cnudde *et al.* [16] discusses three potential solutions to "separate" logic.

- To perform sequential operations instead of parallelism where the implementation processes a non-complete set of shares at each clock cycle. While this provides security this reduces the throughput and avoids making use of the full parallelism feature of hardware.
- Via the use of embedded voltage regulators (VRM) inside the chip, which are already used in commercial smart cards. However, it remains unclear whether electromagnetic signals exhibit potential issues.
- Via the chip having separate Vdd lines to supply functions associated to each share independently. However, it is still an open problem how to supply nonlinear functions which operate on sets of shares and prove the security of the countermeasure.

In this work, we assume one of the above countermeasures has been applied to the masking. To that end, we assume the masking is "domain non-complete" meaning that each domain processes only a non-complete set of shares per cycle. An example is shown in Figure 8. As a result, a coupling-extended probe (for any c in the above definition) does not yield all the shares of a secret. This non-completeness property alone is sufficient to protect an implementation against coupling-extended probes. However, a more detailed security analysis is needed when combining multiple leakage effects together.

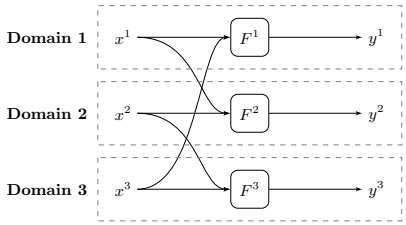

Figure 8: Separation into domains of a layer of masked Boolean functions.

5 Analyzing Glitches, Transitions, and Couplings

For the final analysis, we study maskings which secure against all combined effects described by Faust *et al.* [22], *i.e.* glitches, transitions, and coupling leakage. Considering coupling effects, we study the effect when $c = 1$ following the definition above. That means the adversary is capable of observing the probed wire along with one coupled wire. Recall that we consider that a coupling-extended probe can not view all shares of a secret in one cycle. Combining coupling effects with glitches and transitions, using a single probe, the adversary observes all inputs to the probed wire, the previous values which flowed through those resources, and glitches and transition leakage from a coupled operation. We revisit the case studies from Section 3.

5.1 PRESENT: Serial Architecture

We revisit the masking of PRESENT from Section 3.1. More specifically, we consider a glitch-extended probing secure masking in an S-box serial architecture which we showed was secure against the combined effect of glitches and transitions. We adapt the architecture to work share-serial meaning that each cycle one share of one S-box (or quadratic function in the S-box) is calculated. In particular, such an architecture is domain non-complete.

In this case, glitch effects already include the effect of couplings as only one operation per cycle is calculated. When the architecture would calculate two shares of the same S-box consecutively, transitions could leak the input secret

of the S-box. Instead, we interleave computation on shares of a secret with the computation of other parts of the state. Thus, the architecture would first calculate the i^{th} share of each S-box before calculating a $(i+1)^{\text{th}}$ share. This secures from harmful transition leakage. The activity patterns are the same as before, given in red in Figure 1.

5.2 AES: Serial Architecture

We consider a glitch-extended probing secure masking in a serial architecture where only one S-box and `MixColumns` is implemented. This is a popular architecture for maskings, examples can be found in the works by De Meyer *et al.* [18, Figure 6] and Gross *et al.* [24, Figure 5]. In this case, we consider that a domain separation through different Vdd lines or via embedded voltage regulators has been implemented such that the design is domain non-complete.

Even though there is only one masked S-box on the implementation, it typically consists of multiple register stages. Thus, the S-box can compute on several bytes at once in a pipelined manner. Recall that we assume that $c = 1$, thus a probe on an S-box views the computation of another coupled S-box. Due to transition leakage, the adversary can view the two S-boxes over two consecutive cycles. Due to the `MixColumns` having branch number five, this leakage can never reveal a secret. However, there is still leakage from the calculation of the `MixColumns` as noted in Section 3.2. Finally, there can be coupling leakage between the `MixColumns` and an S-box (from a different column). However, these cases do not add extra harmful activity patterns. Thus, as explained in Section 3.2, we only require adding one cell of extra randomness to secure against the combined effect of glitches, transitions, and couplings.

5.3 ASCON: Round-Based Architecture

Finally, we consider a glitch-extended probing secure masking with a round-based architecture for ASCON. Again, we consider the case where $c = 1$ meaning that a probe can observe an additional operation in that cycle and we consider domain non-complete maskings where there is some countermeasure ensuring coupling leakage alone does not reveal all shares of a secret.

Due to the combined effect of transitions and couplings, a probed S-box gives information on two S-boxes over two consecutive rounds. Previously, we argued ASCON was secure when probing an S-box due to the nontrivial branch number of the linear layer. This argument no longer holds as the linear layer only has branch number four. However, the case remains secure as the layer does not allow for transitions of two active input bits to two active output bits. This can be verified on sight from the equations of the Σ function:

$$\Sigma_{\alpha,\beta}(x) = x \oplus (x \ggg \alpha) \oplus (x \ggg \beta)\,,$$

with $\ggg$ the right circular shift and $\alpha, \beta \in \mathbb{N}$ constants specific for ASCON.

When the adversary probes the linear layer, it can potentially activate two input and two output bits of a shared S-box. The diffusion of the S-box is not sufficient to prevent harmful leakage. However, we observe that the probe returns bits from the same share, e.g. both from the first share of the input and first share of the output. Thus, one can protect the implementation when the shared S-box $\bar{S}$ has the following property. Given an s-sharing $\bar{S}$ and $\mathbb{V} = \bigoplus_{i=1}^{s} \mathbb{V}^i$ such that each mask $u \in \mathbb{F}_2^{5s}/\mathbb{V}^\perp$ can be decomposed in $(u^1, ..., u^s)$ with $u^i \in \mathbb{F}_2^5/(\mathbb{V}^i)^\perp$ then for $i \in \{1, ..., s\}$

$$\forall \alpha, \beta \in \mathbb{F}_2^{5s}/\mathbb{V}^\perp : \; C^{\bar{S}}_{\alpha,\beta} = 0 \text{ when } \alpha = (0, ..., \alpha^i, ..., 0), \beta = (0, ..., \beta^i, ..., 0)\,.$$

A similar property was explored in Section 3.2 on diffusion between shares. Since the S-box is quadratic, one can easily find non-complete sharings of the entire S-box. This non-completeness can be used to argue that the i^{th} share of the input and output of the S-box do not reveal any secret information. An example sharing with the above property is given in Appendix B. For example, the first bit of the second output share is

$$\chi_1^2 = x_1^2 + x_3^2 + (x_2^2 + x_2^3 + x_2^4)(x_3^2 + x_3^3 + x_3^4)\,.$$

Together with the second input shares x_i^2 for $i \in \{1, 2, 3, 4, 5\}$, one always misses the first input shares x_i^1 to retrieve an input secret.

6 Conclusion

This work discussed the security of masked symmetric primitives against the combined leakage effects of glitches, transitions, and couplings. This was done using the standard tools from linear cryptanalysis and on case studies of symmetric primitives. The case studies were made considering black box glitch-extended probing secure maskings. Moreover, we covered case studies on PRESENT, AES, and ASCON to show our analysis is applicable to a wide range of primitives.

Interesting future work would be transform our analysis method (which currently is only done by hand) into a tool which can verify netlists on potential transition leakage using the linear cryptanalytic properties of basic gates such as ANDs and XORs. We also did not investigate methods to analyze transition leakage when re-using randomness between S-boxes or transition leakage inside a single S-box. However, for such cases, brute-force verification is possible allowing for extensions of existing tools such as SILVER [28]. Finally, we noted a lack of theory concerning correlation matrices of masked functions together with interesting examples which indicate that bounds on the maximum absolute correlation or branch number of a masking might be difficult to find. More research on this topic could improve the security and efficiency of masked designs.

Acknowledgment. I thank Vincent Rijmen, Svetla Nikova, Venci Nikov, Tim Beyne, and Adrián ranea for the interesting discussions and their advice. Siemen Dhooghe is supported by a PhD Fellowship from the Research Foundation – Flanders (FWO).

References

1. Banik, S., Pandey, S.K., Peyrin, T., Sasaki, Y., Sim, S.M., Todo, Y.: GIFT: A small present - towards reaching the limit of lightweight encryption. In: Fischer, W., Homma, N. (eds.) CHES 2017. LNCS, vol. 10529, pp. 321–345. Springer, Heidelberg (Sep 2017). https://doi.org/10.1007/978-3-319-66787-4˙16
2. Beierle, C., Jean, J., Kölbl, S., Leander, G., Moradi, A., Peyrin, T., Sasaki, Y., Sasdrich, P., Sim, S.M.: The SKINNY family of block ciphers and its low-latency variant MANTIS. In: Robshaw, M., Katz, J. (eds.) CRYPTO 2016, Part II. LNCS, vol. 9815, pp. 123–153. Springer, Heidelberg (Aug 2016). https://doi.org/10.1007/978-3-662-53008-5˙5
3. Bertoni, G., Daemen, J., Peeters, M., Van Assche, G.: The Keccak Reference. http://http://keccak.noekeon.org/
4. Beyne, T.: Block cipher invariants as eigenvectors of correlation matrices. In: Peyrin, T., Galbraith, S. (eds.) ASIACRYPT 2018, Part I. LNCS, vol. 11272, pp. 3–31. Springer, Heidelberg (Dec 2018). https://doi.org/10.1007/978-3-030-03326-2˙1
5. Beyne, T., Dhooghe, S., Zhang, Z.: Cryptanalysis of masked ciphers: A not so random idea. In: ASIACRYPT 2020, Part I. pp. 817–850. LNCS, Springer, Heidelberg (Dec 2020). https://doi.org/10.1007/978-3-030-64837-4˙27
6. Bilgin, B., Daemen, J., Nikov, V., Nikova, S., Rijmen, V., Assche, G.V.: Efficient and first-order DPA resistant implementations of keccak. In: Francillon, A., Rohatgi, P. (eds.) Smart Card Research and Advanced Applications - 12th International Conference, CARDIS 2013, Berlin, Germany, November 27-29, 2013. Revised Selected Papers. Lecture Notes in Computer Science, vol. 8419, pp. 187–199. Springer (2013). https://doi.org/10.1007/978-3-319-08302-5_13, https://doi.org/10.1007/978-3-319-08302-5_13
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 2007. LNCS, vol. 4727, pp. 450–466. Springer, Heidelberg (Sep 2007). https://doi.org/10.1007/978-3-540-74735-2˙31
8. Borghoff, J., Canteaut, A., Güneysu, T., Kavun, E.B., Knežević, M., Knudsen, L.R., Leander, G., Nikov, V., Paar, C., Rechberger, C., Rombouts, P., Thomsen, S.S., Yalçin, T.: PRINCE - A low-latency block cipher for pervasive computing applications - extended abstract. In: Wang, X., Sako, K. (eds.) ASIACRYPT 2012. LNCS, vol. 7658, pp. 208–225. Springer, Heidelberg (Dec 2012). https://doi.org/10.1007/978-3-642-34961-4˙14
9. Cassiers, G., Standaert, F.: Trivially and efficiently composing masked gadgets with probe isolating non-interference. IEEE Trans. Inf. Forensics Secur. **15**, 2542–2555 (2020). https://doi.org/10.1109/TIFS.2020.2971153, https://doi.org/10.1109/TIFS.2020.2971153
10. Cassiers, G., Standaert, F.: Provably secure hardware masking in the transition- and glitch-robust probing model: Better safe than sorry. IACR Trans. Cryptogr. Hardw. Embed. Syst. **2021**(2), 136–158 (2021). https://doi.org/10.46586/tches.v2021.i2.136-158, https://doi.org/10.46586/tches.v2021.i2.136-158
11. Chari, S., Jutla, C.S., Rao, J.R., Rohatgi, P.: Towards sound approaches to counteract power-analysis attacks. In: Wiener, M.J. (ed.) CRYPTO'99. LNCS, vol. 1666, pp. 398–412. Springer, Heidelberg (Aug 1999). https://doi.org/10.1007/3-540-48405-1˙26

12. Daemen, J.: Changing of the guards: A simple and efficient method for achieving uniformity in threshold sharing. In: Fischer, W., Homma, N. (eds.) CHES 2017. LNCS, vol. 10529, pp. 137–153. Springer, Heidelberg (Sep 2017). https://doi.org/10.1007/978-3-319-66787-4˙7
13. Daemen, J., Govaerts, R., Vandewalle, J.: Correlation matrices. In: Preneel, B. (ed.) FSE'94. LNCS, vol. 1008, pp. 275–285. Springer, Heidelberg (Dec 1995). https://doi.org/10.1007/3-540-60590-8˙21
14. Daemen, J., Rijmen, V.: Advanced Encryption Standard (AES). National Institute of Standards and Technology (NIST), FIPS PUB 197, U.S. Department of Commerce (Nov 2001)
15. De Cnudde, T., Bilgin, B., Gierlichs, B., Nikov, V., Nikova, S., Rijmen, V.: Does coupling affect the security of masked implementations? In: Guilley, S. (ed.) COSADE 2017. LNCS, vol. 10348, pp. 1–18. Springer, Heidelberg (Apr 2017). https://doi.org/10.1007/978-3-319-64647-3˙1
16. De Cnudde, T., Ender, M., Moradi, A.: Hardware masking, revisited. IACR TCHES **2018**(2), 123–148 (2018). https://doi.org/10.13154/tches.v2018.i2.123-148, https://tches.iacr.org/index.php/TCHES/article/view/877
17. De Cnudde, T., Reparaz, O., Bilgin, B., Nikova, S., Nikov, V., Rijmen, V.: Masking AES with d+1 shares in hardware. In: Gierlichs, B., Poschmann, A.Y. (eds.) CHES 2016. LNCS, vol. 9813, pp. 194–212. Springer, Heidelberg (Aug 2016). https://doi.org/10.1007/978-3-662-53140-2˙10
18. De Meyer, L., Reparaz, O., Bilgin, B.: Multiplicative masking for AES in hardware. IACR TCHES **2018**(3), 431–468 (2018). https://doi.org/10.13154/tches.v2018.i3.431-468, https://tches.iacr.org/index.php/TCHES/article/view/7282
19. Dhooghe, S., Nikova, S.: Let's tessellate: Tiling for security against advanced probe and fault adversaries. IACR Cryptol. ePrint Arch. **2020**, 1146 (2020), https://eprint.iacr.org/2020/1146
20. Dhooghe, S., Nikova, S., Rijmen, V.: Threshold implementations in the robust probing model. In: Bilgin, B., Petkova-Nikova, S., Rijmen, V. (eds.) Proceedings of ACM Workshop on Theory of Implementation Security Workshop, TIS@CCS 2019, London, UK, November 11, 2019. pp. 30–37. ACM (2019). https://doi.org/10.1145/3338467.3358949
21. Dobraunig, C., Eichlseder, M., Mendel, F., Schläffer, M.: Ascon v1.2. https://ascon.iaik.tugraz.at/files/asconv12-nist.pdf
22. Faust, S., Grosso, V., Pozo, S.M.D., Paglialonga, C., Standaert, F.X.: Composable masking schemes in the presence of physical defaults & the robust probing model. IACR TCHES **2018**(3), 89–120 (2018). https://doi.org/10.13154/tches.v2018.i3.89-120, https://tches.iacr.org/index.php/TCHES/article/view/7270
23. Goubin, L., Patarin, J.: DES and differential power analysis (the "duplication" method). In: Koç, Çetin Kaya., Paar, C. (eds.) CHES'99. LNCS, vol. 1717, pp. 158–172. Springer, Heidelberg (Aug 1999). https://doi.org/10.1007/3-540-48059-5˙15
24. Groß, H., Mangard, S., Korak, T.: Domain-oriented masking: Compact masked hardware implementations with arbitrary protection order. In: Bilgin, B., Nikova, S., Rijmen, V. (eds.) Proceedings of the ACM Workshop on Theory of Implementation Security, TIS@CCS 2016 Vienna, Austria, October, 2016. p. 3. ACM (2016). https://doi.org/10.1145/2996366.2996426, https://doi.org/10.1145/2996366.2996426

25. Guo, J., Peyrin, T., Poschmann, A.: The PHOTON family of lightweight hash functions. In: Rogaway, P. (ed.) CRYPTO 2011. LNCS, vol. 6841, pp. 222–239. Springer, Heidelberg (Aug 2011). https://doi.org/10.1007/978-3-642-22792-9˙13
26. Guo, J., Peyrin, T., Poschmann, A., Robshaw, M.J.B.: The LED block cipher. In: Preneel, B., Takagi, T. (eds.) CHES 2011. LNCS, vol. 6917, pp. 326–341. Springer, Heidelberg (Sep / Oct 2011). https://doi.org/10.1007/978-3-642-23951-9˙22
27. Ishai, Y., Sahai, A., Wagner, D.: Private circuits: Securing hardware against probing attacks. In: Boneh, D. (ed.) CRYPTO 2003. LNCS, vol. 2729, pp. 463–481. Springer, Heidelberg (Aug 2003). https://doi.org/10.1007/978-3-540-45146-4˙27
28. Knichel, D., Sasdrich, P., Moradi, A.: SILVER - statistical independence and leakage verification. In: ASIACRYPT 2020, Part I. pp. 787–816. LNCS, Springer, Heidelberg (Dec 2020). https://doi.org/10.1007/978-3-030-64837-4˙26
29. Kocher, P.C., Jaffe, J., Jun, B.: Differential power analysis. In: Wiener, M.J. (ed.) CRYPTO'99. LNCS, vol. 1666, pp. 388–397. Springer, Heidelberg (Aug 1999). https://doi.org/10.1007/3-540-48405-1˙25
30. Kutzner, S., Nguyen, P.H., Poschmann, A., Wang, H.: On 3-share threshold implementations for 4-bit S-boxes. In: Prouff, E. (ed.) COSADE 2013. LNCS, vol. 7864, pp. 99–113. Springer, Heidelberg (Mar 2013). https://doi.org/10.1007/978-3-642-40026-1˙7
31. Mangard, S., Pramstaller, N., Oswald, E.: Successfully attacking masked AES hardware implementations. In: Rao, J.R., Sunar, B. (eds.) CHES 2005. LNCS, vol. 3659, pp. 157–171. Springer, Heidelberg (Aug / Sep 2005). https://doi.org/10.1007/11545262˙12
32. Matsui, M.: Linear cryptanalysis method for DES cipher. In: Helleseth, T. (ed.) EUROCRYPT'93. LNCS, vol. 765, pp. 386–397. Springer, Heidelberg (May 1994). https://doi.org/10.1007/3-540-48285-7˙33
33. Nikova, S., Rechberger, C., Rijmen, V.: Threshold implementations against side-channel attacks and glitches. In: Ning, P., Qing, S., Li, N. (eds.) ICICS 06. LNCS, vol. 4307, pp. 529–545. Springer, Heidelberg (Dec 2006)
34. Sugawara, T.: 3-share threshold implementation of AES s-box without fresh randomness. IACR TCHES **2019**(1), 123–145 (2018). https://doi.org/10.13154/tches.v2019.i1.123-145, https://tches.iacr.org/index.php/TCHES/article/view/7336
35. Tardy-Corfdir, A., Gilbert, H.: A known plaintext attack of FEAL-4 and FEAL-6. In: Feigenbaum, J. (ed.) CRYPTO'91. LNCS, vol. 576, pp. 172–181. Springer, Heidelberg (Aug 1992). https://doi.org/10.1007/3-540-46766-1˙12
36. Wegener, F., Moradi, A.: A first-order SCA resistant AES without fresh randomness. In: Fan, J., Gierlichs, B. (eds.) COSADE 2018. LNCS, vol. 10815, pp. 245–262. Springer, Heidelberg (Apr 2018). https://doi.org/10.1007/978-3-319-89641-0˙14

A Three Sharing of the PRESENT S-Box

This appendix provides a decomposition of the PRESENT S-box and a three sharing of the S-box. We denote by (x, y, z, w) the input nibble from most significant to least significant bit.

Following the work by Kutzner *et al.* [30], the PRESENT S-box S can be decomposed as follows

$$S(x, y, z, w) = B'(G(G(C'(x, y, z, w) + d)) + e)\,.$$

In the above, the nonlinear function $G(x, y, z, w)$ is given as

$$G_1 = x + yz + yw \qquad G_2 = w + xy \qquad G_3 = y \qquad G_4 = z + yw\,.$$

This permutation G is shared using a direct balanced sharing. More specifically, for each share $i \in \{1, 2, 3\}$

$$\begin{aligned}
G_1^i &= x^i + y^i z^i + y^i z^{i+1} + y^{i+1} z^i + y^i w^i + y^i w^{i+1} + y^{i+1} w^i\,,\\
G_2^i &= w^i + x^i y^i + x^i y^{i+1} + x^{i+1} y^i\,,\\
G_3^i &= y^i\,,\\
G_4^i &= z^i + y^i w^i + y^i w^{i+1} + y^{i+1} w^i\,,
\end{aligned}$$

where the convention is used that superscripts wrap around at three. The linear layers are masked share-wise.

B Uniform Sharing of the ASCON S-Box

This appendix provides a uniform four-sharing of the ASCON S-box. Recall that the ASCON S-box is affine equivalent to the Keccak S-box. More specifically, for the Keccak S-box χ and the ASCON S-box S we have that

$$S(x) = B(\chi(B(x))) + c\,,$$

with A, B linear transformations and c a constant.

Denoting the five input bits by $\{x_1, x_2, x_3, x_4, x_5\}$ going from least significant to most significant bit. A uniform sharing of the Keccak S-box χ using four shares was given by Bilgin *et al.* [6]. For $i = 1, 2, 3, 5$ we have

$$\begin{aligned}
\chi_i^1 &= x_i^1 + x_{i+2}^1\,,\\
\chi_i^2 &= x_i^2 + x_{i+2}^2 + (x_{i+1}^2 + x_{i+1}^3 + x_{i+1}^4)(x_{i+2}^2 + x_{i+2}^3 + x_{i+2}^4)\,,\\
\chi_i^3 &= x_i^3 + x_{i+2}^3 + x_{i+1}^1(x_{i+2}^3 + x_{i+2}^4) + x_{i+2}^1(x_{i+1}^3 + x_{i+1}^4) + x_{i+1}^1 x_{i+2}^1\,,\\
\chi_i^4 &= x_i^4 + x_{i+2}^4 + x_{i+1}^1 x_{i+2}^2 + x_{i+2}^1 x_{i+1}^2\,,
\end{aligned}$$

where the convention is used that subscripts wrap around at five. For the remaining fourth coordinate function we have

$$\begin{aligned}
\chi_4^1 &= x_4^1\,.\\
\chi_4^2 &= x_4^2 + x_1^2 + x_1^3 + x_1^4 + (x_5^2 + x_5^3 + x_5^4)(x_1^2 + x_1^3 + x_1^4)\,,\\
\chi_4^3 &= x_4^3 + x_1^1 + x_5^1(x_1^3 + x_1^4) + x_1^1(x_5^3 + x_5^4) + x_1^1 x_5^1\,,\\
\chi_4^4 &= x_4^4 + x_5^1 x_1^2 + x_1^1 x_5^2\,,
\end{aligned}$$