Improving First-Order Threshold Implementations of SKINNY

Andrea Caforio[1], Daniel Collins[1], Ognjen Glamočanin[2], and Subhadeep Banik[1]

[1] LASEC, Ecole Polytechnique Fédérale de Lausanne, Switzerland
{andrea.caforio,daniel.collins,subhadeep.banik}@epfl.ch
[2] PARSA, Ecole Polytechnique Fédérale de Lausanne, Switzerland
ognjen.glamocanin@epfl.ch

Abstract. Threshold Implementations have become a popular generic technique to construct circuits resilient against power analysis attacks. In this paper, we look to devise efficient threshold circuits for the lightweight block cipher family SKINNY. The only threshold circuits for this family are those proposed by its designers who decomposed the 8-bit S-box into four quadratic S-boxes, and constructed a 3-share byte-serial threshold circuit that executes the substitution layer over four cycles. In particular, we revisit the algebraic structure of the S-box and prove that it is possible to decompose it into (a) three quadratic S-boxes and (b) two cubic S-boxes. Such decompositions allow us to construct threshold circuits that require three shares and executes each round function in three cycles instead of four, and similarly circuits that use four shares requiring two cycles per round. Our constructions significantly reduce latency and energy consumption per encryption operation. Notably, to validate our designs, we synthesize our circuits on standard CMOS cell libraries to evaluate performance, and we conduct leakage detection via statistical tests on power traces on FPGA platforms to assess security.[1]

Keywords: DPA · Masking · SKINNY· Threshold Implementation

1 Introduction

Side-channel attacks have been widely successful at efficiently attacking implementations of cryptosystems. Power analysis has been particularly effective in part due to the relatively low cost of the requisite equipment. In differential power analysis [17] (DPA) and its generalizations [6,18], an attacker observes the power consumption of a cryptographic primitive over time and applies statistical analysis to infer the underlying secret key. An attacker can perform a d-th order attack, e.g., by probing up to d internal wires of the circuit at once [15].

In an attempt to mitigate the damaging effects of side-channel attacks, the development of countermeasures has proliferated. Masking is one such approach

[1] For reproducibility's sake, we provide a public repository containing the source code to all proposed schemes together with a script to run the SILVER verification suite [8].

which uses secret sharing to randomize input and intermediate values within a circuit. To standardise the error-prone and often ad-hoc process of designing secure masked circuits, Threshold Implementations (TI) were introduced which provide provable security with respect to side-channel attacks [4,10,20]. When implemented in hardware, a TI is secure even in the presence of glitches, an inherent side effect not considered in earlier schemes [15].

A correct TI must satisfy so-called non-completeness and uniformity to ensure security. Satisfying these properties for linear components of a given circuit is relatively straight-forward. Non-linear components are less trivial; a t-degree function must be split into at least $(td + 1)$ coordinate functions in the canonical higher-order TI [4] to provide d-th order security guarantees. Approaches to reduce this complexity like adding additional randomness exist [3], but there is an inherent trade-off between area, randomness requirements and latency when designing a TI of a given circuit. Unsurprisingly, TI schemes for AES and Keccak have enjoyed the most attention the literature. Recent works include [25,27,29] and [1,28] respectively.

1.1 SKINNY

SKINNY is a lightweight family of tweakable block ciphers designed by Beierle et al. [2]. The cipher performs extremely well on both software and hardware platforms, and is the core encryption primitive used in the authenticated encryption scheme Romulus [14] which is a finalist in the NIST lightweight cryptography competition [26]. Moreover, a criterion for the competition is the efficiency of protected circuit implementations. In the 64-bit block size versions of SKINNY, the underlying S-box defined over four bits. Designing Threshold Implementations for 4-bit S-boxes is a well-studied problem [5], and so in this work we focus on the 128-bit block size versions of SKINNY which use an 8-bit S-box, hereafter denoted by S.

S is very lightweight and uses only sixteen cross-connected two-input logic gates. Using the fact that S can be decomposed in the form $I \circ H \circ G \circ F$ (hereafter denoted by S_{2222}[2]) where each sub-function is quadratic, the designers of SKINNY proposed a first-order TI of SKINNY using a byte-serial circuit. However, when this decomposition is used to construct a TI of a round-based circuit, a single S-box layer takes four cycles to execute. This increases the latency and hence energy consumption per encryption operation in the circuit, as was shown in [7].

1.2 Contributions and Organization

In this paper, we take a closer look at first-order Threshold Implementations of the 8-bit substitution box of round-based SKINNY instantiations. As previously mentioned, the only in-depth analysis and indeed proposal of such a masked

[2] Note that throughout this paper we use the notation $\mathsf{S}_{i_1 \dots i_k}$ to denote decompositions of the same S-box S into k component S-boxes of algebraic degrees $i_1 \dots i_k$.

circuit is that of S_{2222} which appeared in the design paper [2] for the byte-serial variant of SKINNY. This 3-share scheme is likely the optimal choice for a first-order secure realization in the byte-serial setting when it comes to area, latency and power/energy consumption. However, for round-based circuits, this assertion does not hold true anymore. In fact, we propose two novel decompositions that eclipse the existing variant in both latency, power and energy consumption without significantly increasing the circuit area. More specifically, our contributions are summarized as follows:

1. We devise an approach that exploits the simple 4×4 cross-connected structure of S and automatizes the search for decompositions and thus Threshold Implementations.
2. The proposed technique is then used as a gateway to efficiently decompose S into three quadratic functions $\mathsf{S}_{222} = H \circ G \circ F$ that is computed over three cycles. The resulting 3-share masked circuit exhibits a similar area footprint to S_{2222} but cuts the number of required cycles for an encryption by one quarter and consumes around 30% less energy across different clock frequencies and cell libraries.
3. In a second step, by extending the previous technique, we propose a decomposition of S into two cubic functions $\mathsf{S}_{33} = G \circ F$ that is thus computed in two cycles. The corresponding 4-share TI halves the number of encryption cycles and consumes 30% less energy while moderately increasing the circuit area relative to S_{2222}. We emphasise that neither of the above circuits require additional randomness beyond the initial plaintext masking.
4. We provide an extensive suite of synthesis measurements on both ASIC and FPGA targets for all investigated schemes showcasing the advantages of both S_{222} and S_{33}.
5. The proposed schemes are proven sound via the SILVER verification framework [16] that performs its analysis on ASIC netlists, which in our case are generated by the NanGate 45 nm standard cell library. In addition, we perform practical leakage assessments using the TVLA methodology [12,24] by taking power traces on FPGA targets.

The paper unfolds as follows: Section 2 reiterates some preliminaries regarding masking and Threshold Implementations. Subsequently in Section 3, we detail the derivation of S_{222} and S_{33}. Synthesis results are given in Section 4 and leakage assessment is performed in Section 5. Finally, we conclude in Section 6.

2 Preliminaries

Masked hardware implementations of cryptographic algorithms use the secret sharing methodology in which key-related, intermediate values x_i are split into s independent shares $x_{i,0}, x_{i,1}, \ldots, x_{i,s-1}$ such that $\sum_{j=0}^{s-1} x_{i,j} = x_i$. In practice, sharing variables implies that each function $f(x_{n-1}, \ldots, x_0) = z$ within an algorithm needs to be decomposed into functions $f_i(\cdot) = z_i$ adhering to the same correctness requirement $\sum_{i=0}^{s-1} f_i = f$.

In the following, we assume that an attacker is capable of probing individual wires of a circuit and can extract their intermediate values during the computation [15]. More specifically, we consider d-th order security, where information of any d wires can be gathered and processed. Different d-th order security properties can be defined and satisfied by a given design [9], the most natural being *d-probing* security which is satisfied given that any observation made on up to d wires is statistically independent of the secret [15]. Security properties are further considered with respect to a leakage model. Two such models of interest are the *standard* model, where a circuit without any glitching or unintended behaviour is assumed, and the *glitch-robust* model [10,11] which accounts for such behaviour. Hereafter, we say that a masked implementation in a given leakage model is d-th order secure if it is d-probing secure. There is a correspondence between d-probing security and security against d-th order differential power analysis (hereafter DPA), where the latter is implied by d-th glitch-robust probing security [3].

Threshold Implementation. The task of designing d-th order secure masking schemes has spawned various approaches, of which Threshold Implementations have crystallized themselves as one of the most adopted strategies. First introduced by Nikova et al. [4,19] Threshold Implementations provide some d-th order security guarantees against DPA in the presence of hardware glitches that are inherent to any CMOS circuit. We note that higher-order TI as defined below does not necessarily ensure d-th order security without additional measures [22,23]. Nonetheless, in the first-order setting, our setting of interest in this work, a first-order Threshold Implementation achieves first-order security in the glitch-robust model [10].

The decomposition of an n-variable Boolean function $f(x_{n-1}, \ldots, x_0) = z$ into a set of s functions $f_0, \ldots, f_{s-1}$ such that $\sum_{i=0}^{s-1} f_i = f$ is a d-th order Threshold Implementation if and only if the following conditions are met:

1. *Non-Completeness.* The functions $f_0, \ldots, f_{s-1}$ are d-th order non-complete, if any combination of at most d functions is independent of at least one input share.
2. *Uniformity.* For all x such that $f(x) = z$, the input masking is said to be uniform if each set of valid input shares of x (i.e., those sum to x) have equal probability of occurring. If this holds, the shared implementation of f is said to be uniform if each valid output share also have equal probability of occurring.

The number of input shares s_{in} respectively output shares s_{out} required to achieve a non-complete and uniform sharing of a function of algebraic degree t is given by the below bounds [4]:

$$s_{\text{in}} \geq td + 1, \quad s_{\text{out}} \geq \binom{td+1}{t}.$$

Note that a first-order TI of a quadratic function can thus be obtained with $s_{\text{in}} = s_{\text{out}} = 3$. In this work, we will bootstrap the sharing of an arbitrary

quadratic function via the canonical direct sharing of the function $f(x_2, x_1, x_0) = x_0 + x_1x_2$, i.e.,

$$\begin{aligned} f_0 &= x_{0,1} + x_{1,1}x_{2,1} + x_{1,1}x_{2,2} + x_{1,2}x_{2,1} \\ f_1 &= x_{0,2} + x_{1,0}x_{2,0} + x_{1,2}x_{2,0} + x_{1,0}x_{2,2} \\ f_2 &= x_{0,0} + x_{1,0}x_{2,0} + x_{1,1}x_{2,0} + x_{1,0}x_{2,1}. \end{aligned}$$

We use an analogous direct sharing for cubic terms.

2.1 SKINNY-128 Substitution Box

As Threshold Implementations of linear functions are obtained by simple decompositions, the crux lies in finding efficient sharings for non-linear mappings. In our case, this involves the 8-bit substitution box of the 128-bit block size variants of SKINNY with different tweakey sizes which we denote by SKINNY-128, SKINNY-256 and SKINNY-384 given by the iterative mapping

$$\Pi' \circ T \circ [\Pi \circ T]^3 (x_7, x_6, x_5, x_4, x_3, x_2, x_1, x_0) = (z_7, z_6, z_5, z_4, z_3, z_2, z_1, z_0),$$

composed of a transformation T and two bitwise permutations Π, Π' such that

$$\begin{aligned} T(x_7, \ldots, x_1, x_0) &= (x_7, x_6, x_5, x_4 + (x_7 \mathbin{\overline{\vee}} x_6), x_3, x_2, x_1, x_0 + (x_3 \mathbin{\overline{\vee}} x_2)) \\ \Pi(x_7, \ldots, x_1, x_0) &= (x_2, x_1, x_7, x_6, x_4, x_0, x_3, x_5) \\ \Pi'(x_7, \ldots, x_1, x_0) &= (x_7, x_6, x_5, x_4, x_3, x_1, x_2, x_0). \end{aligned}$$

Here, $\overline{\vee}$ denotes the logical NOR gate, i.e., $x \mathbin{\overline{\vee}} y = xy + x + y + 1$. A graphical depiction of the 8-bit S-box circuit is given in Figure 1a. Note that the highest algebraic degree of six is reached in output term z_0. The full expression of each term is given in Appendix A.

3 Partitioning the S-box S

In [21], the authors showed how to decompose the S-box S_P of the PRESENT block cipher into two quadratic S-boxes F, G such that $\mathsf{S}_P = G \circ F$. This enabled the authors to construct a 3-share TI of PRESENT by constructing Threshold Implementations of F and G separately with a register bank in between which suppresses and thus prevents the glitches produced by the F layer from propagating to the G layer. This however means that every evaluation of the shared S-box requires two cycles to complete. However, this is compensated by the fact that the construction requires only three shares and thus the total silicon area required for the circuit is minimal. The approach used by the authors to obtain the decomposition can be summarized as follows:

1. Evaluate all quartets of 4-bit vectorial Boolean functions f_0, f_1, f_2, f_3 such that all the f_i's are quadratic. There are 2^{11} quadratic functions in 4 bits and so a total of 2^{44} such quartets are possible.

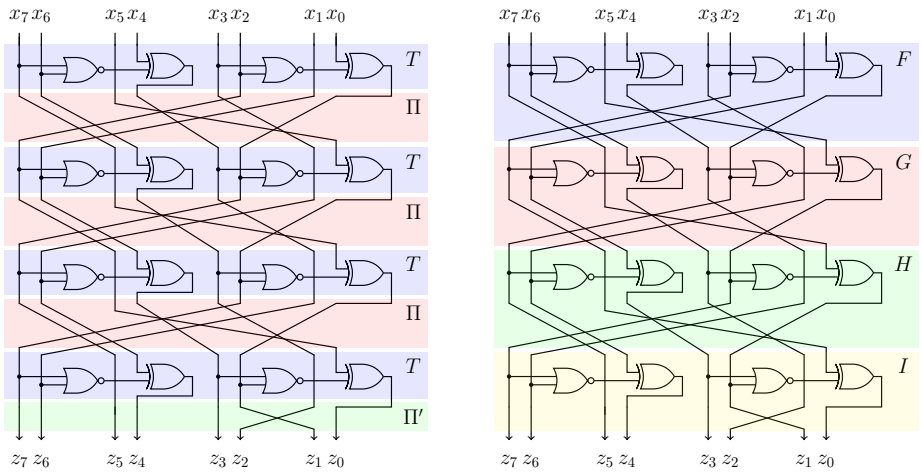

Fig. 1. (a) Definition of the 8-bit SKINNY-128 substitution box given the transformation T and two permutations Π, Π'. (b) TI decomposition proposed in [2] using four quadratic functions F, G, H and I.

2. Of the above list only filter for the quartets such that the function $F : \{0,1\}^4 \to \{0,1\}^4$ with $F(x_0, x_1, x_2, x_3) = (f_0, f_1, f_2, f_3)$ is a bijective S-box.
3. For all such F check if $G = \mathsf{S}_P \circ F^{-1}$ is also a quadratic S-box. If so, output the pair of S-boxes (G, F).

It was later shown in [5] that S_P belongs to the affine equivalence class $\mathcal{C}_{266}$ of 4-bit S-boxes. All S-boxes in this class allows decomposition into two quadratic S-boxes. The above approach can not be extended to 8-bit S-boxes even considering the authors' suggested optimisations. To begin with there are 2^{37} quadratic functions over 8 bits, and therefore the number of octets of the form $f_0, f_1, \ldots, f_7$ will be $2^{37\times 8} = 2^{296}$.

3.1 The Techniques

As done with PRESENT our goal lies in finding decompositions of the 8-bit SKINNY S-box S that allow for efficient Threshold Implementations in terms of circuit area, latency and energy consumption. In turn, this implies finding an appropriate balance between the number of shares, coordinate functions, and their degrees and gate complexity. To obtain a similar decomposition of S let us first state the following definitions:

Definition 1 (i-representable). *A Boolean function B has AND-complexity n, if its circuit can be constructed with a total of n 2-input AND gates or fewer. Its AND-depth is i (or equivalently it is i-representable) if there exists a circuit*

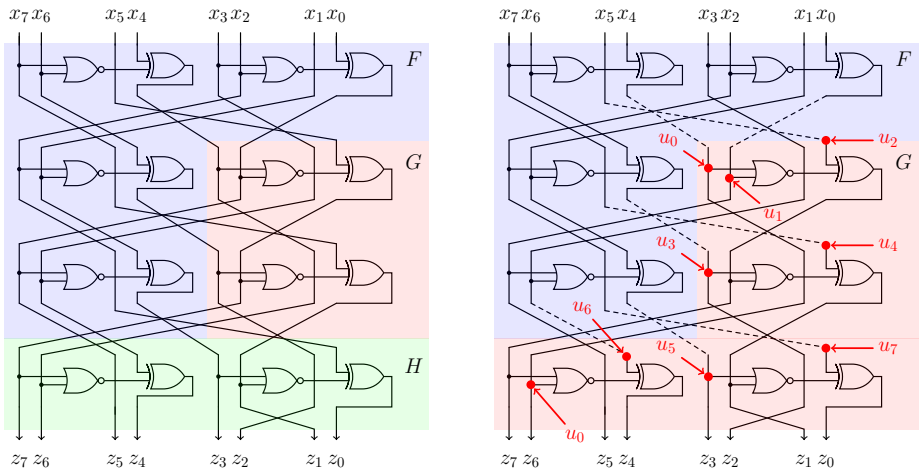

Fig. 2. (a) $\mathsf{S}_{232} = H \circ G \circ F$ decomposition with $\deg(F) = \deg(H) = 2$ and $\deg(G) = 3$. (b) $\mathsf{S}_{24} = G \circ F$ decomposition with $\deg(F) = 2$ and $\deg(G) = 4$. We later introduce the terminology $\mathsf{S}_{\mathsf{Blue}}$ and $\mathsf{S}_{\mathsf{Red}}$ to denote F, G respectively in (b).

in which the AND gates can be arranged in i distinct levels in the following sense: all quadratic functions are 1-representable of some order, and a function B_i is i-representable if it can be expressed as $B_i = Q(t_0, t_1, \ldots, t_{m-1})$ where Q is quadratic and the functions $t_0, t_1, \ldots, t_{m-1}$ are each k-representable of some order for $k \leq (i-1)$. B is i-representable of order n if there exists a circuit which constructs it with AND-depth i and AND-complexity n.

Thus a function which is i-representable of order n can be necessarily implemented by n or a smaller number of 2-input AND gates (connected such that the total AND-depth is at most i) along with other linear gates. Thus all four coordinate functions of S_P are 2-representable of some fixed order, which allows a 3-share TI over two clock cycles.

Regarding S, the eight output functions $z_0, z_1, \ldots, z_7$ are of different algebraic degrees. z_2, z_3, z_5, z_6 are themselves quadratic and their algebraic expressions contain only one quadratic term and hence are 1-representable of order one. z_4, z_7 have algebraic degree four: the fact that z_7 is 2-representable of order three can be easily deduced from Figure 3a: the paths from the input bits to the z_7 node go through exactly three NOR gates arranged so that the depth is two. We have $z_4 = z_7 \barwedge z_6 + x_3$. Hence z_4 is at most 3-representable (in fact we will later prove that it is 2-representable too). z_0 and z_1 have algebraic degree six and five respectively: they can not be 2-representable since the set of all 2-representable functions contains members of degree four or less.

3.2 Exhaustive Partition Search

As mentioned, the byte-serial scheme presented in the SKINNY design paper [2], and later adapted to round-based setting in [7], considers a three-share decomposition into four functions of degree two which we denote by S_{2222}. As a consequence, the S-box operation is performed in a pipelined fashion over four clock cycles which incurs a large latency thus energy penalty, i.e., a single encryption of a plaintext takes four times the number of rounds when implemented as a round-based circuit.

Since z_0 and z_1 are not 2-representable, the decomposition of S into quadratic S-boxes $\mathsf{F}_i \circ \mathsf{F}_{i-1} \circ \cdots \circ \mathsf{F}_1$ is not possible for $i \leq 2$. Consequently, we aim to decompose every coordinate Boolean function of S into 3-representable functions of low order. Given that S can be realized in only 16 logical two-input gates, a natural approach to obtain efficient decompositions is by partitioning the circuit into connected sub-circuits. For example, the S_{2222} decomposition corresponds to making three horizontal cuts after each row of gates. The number of possible partitions of 16 gates into n sets is n^{16}, however among those, only a small fraction of those partitions respect functional correctness. Hence, if $n = 3$, it is feasible to enumerate all correct partitions. Although this procedure does not admit a 3-representable decomposition of each coordinate function, we found many decompositions of the form $\mathsf{S} = H \circ G \circ F$ where $\deg(F) = \deg(H) = 2$ and $\deg(G) = 3$. One such example denoted by S_{232} is shown in Figure 2a.

3.3 A Deeper Dive

As noted above, all coordinate functions of S except z_0 and z_1 are 3-representable. If we can argue that z_0 and z_1 are also 3-representable, then it becomes straightforward to decompose S into three quadratic S-boxes. z_1 is clearly 3-representable of order five as can be deduced from Figure 3b. The set of all paths from the input bits to z_1 traverses exactly five NOR gates arranged in three levels and so the result follows (they are marked in red in Figure 3b).

z_0 is of algebraic degree 6 and from Figure 1 it is at least 4-representable of order 7. This is because all but one of the 8 NOR gates are used to produce the z_0 bit and they are clearly arranged in 4 levels. However the question is: *Is z_0 also 3-representable of a suitable low order?* If yes, a 3-share first-order TI which evaluates the S-box in only three cycles is possible.

In this part we will show that z_0 is indeed 3-representable of order 8. Note that since the algebraic expression for z_0 is very complex, we avoid directly working with it to prove 3-representability: it would be very difficult to keep the AND-complexity down to a suitable value. Instead, consider the function $\pi(x, y, z) = (x \,\overline{\vee}\, y) + z$, whose algebraic expression is given by $xy + x + y + z + 1$. Note that π is completely linear in the last input z. In Figure 4, π is represented by a green circular node, and the figure represents the circuit graph for z_0. The figure itself is redrawn by isolating the circuit path for z_0 as in Figure 1, and will help us prove the 3-representability of z_0. Note that Figure 4 also makes it clear that z_0 is 4-representable of order 7.

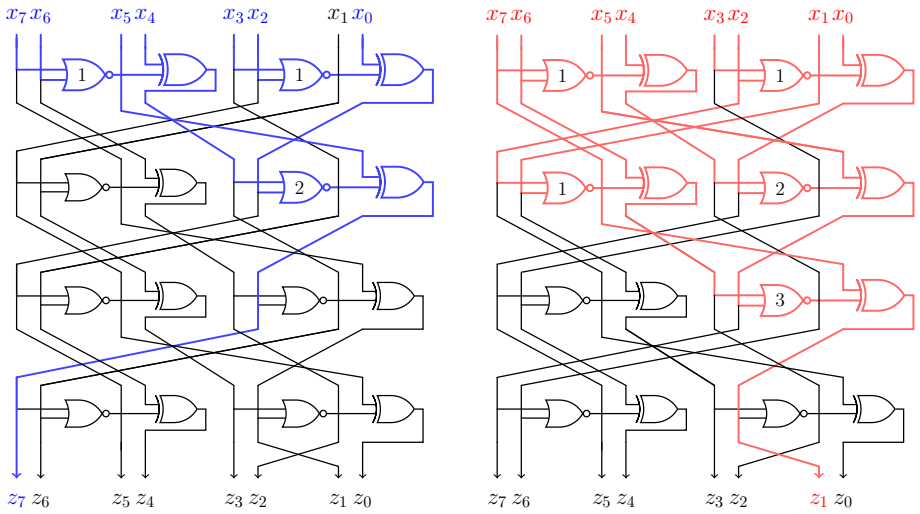

Fig. 3. (a) The path up to z_7 is marked in blue. There are 3 NOR gates, whose levels are marked inside. There is a single NOR gate at level 2, which takes inputs from the 2 other level 1 NOR gates in the first row. (b) The path up to z_1 is marked in red. There are 5 NOR gates, whose levels are marked inside. There is a single NOR gate at level 3, which takes inputs from the level 2 NOR gate and another level 1 NOR gate in the second row.

Lemma 1. *It is possible to transform the circuit graph for z_0 according to the transformation (a) → (b) shown in Figure 5.*

Proof. This transformation is easy to prove: consider the nodes labeled in darker green in Figure 5a. The output bit $e = \pi(b, x_3, x_1)$ is given by the following algebraic expression:

$$\begin{aligned} e &= \pi(b, x_3, x_1) = \pi(\pi(x_2, x_3, x_0), x_3, x_1) \\ &= \pi(x_2x_3 + x_2 + x_3 + x_0 + 1, x_3, x_1) \\ &= x_3(x_2x_3 + x_2 + x_3 + x_0 + 1) + x_3 + (x_2x_3 + x_2 + x_3 + x_0 + 1) + x_1 + 1 \\ &= x_0x_3 + x_2x_3 + x_2 + x_0 + x_1 \\ &= x_3(x_0 + x_2) + (x_0 + x_2) + x_3 + (x_1 + x_3 + 1) + 1 \\ &= \pi(x_0 + x_2, x_3, x_1 + x_3 + 1) \end{aligned}$$

Lemma 2. *It is possible to transform the circuit graph for z_0 according to the transformation (a) → (b) shown in Figure 6. Thus, z_0 is 3-representable of order eight.*

Proof. The proof for this transformation is slightly more involved. Consider again the gates labeled in dark green in Figure 6a. They lie entirely in levels 3 and 4

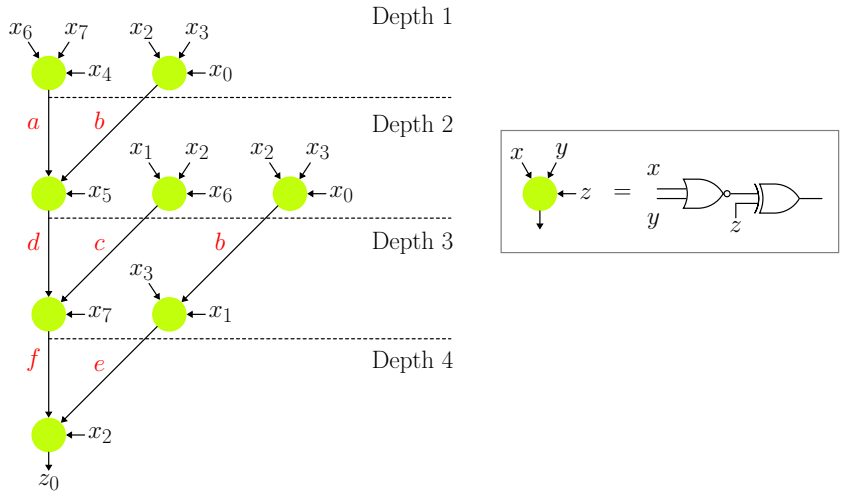

Fig. 4. Circuit graph for z_0. Its AND-complexity is 7 (note the gate $\pi(x_2, x_3, x_0)$ is shown twice for a clearer representation).

of the circuit graph, and takes as input the signals d, c, e, x_7, x_2 and produces z_0 as output. The expression can be written as:

$$\begin{aligned}
z_0 &= \pi(f, e, x_2) = \pi(\pi(d, c, x_7), e, x_2) \\
&= \pi(dc + d + c + x_7 + 1, e, x_2) \\
&= e(dc + d + c + x_7 + 1) + e + (dc + d + c + x_7 + 1) + x_2 + 1 \\
&= edc + ed + ec + ex_7 + dc + d + c + x_7 + x_2 \\
&= d(ec + e + c + 1) + ec + ex_7 + c + x_7 + x_2 \\
&= d(\pi(e, c, 0)) + (ec + e + c + 1) + d + (d + e + 1 + ex_7 + x_7 + x_2) \\
&= d(\pi(e, c, 0)) + \pi(e, c, 0) + d + (ex_7 + e + x_7 + x_2 + 1 + d) \\
&= \pi\Big(\pi(e, c, 0), d, d + \pi(e, x_7, x_2)\Big)
\end{aligned}$$

This completes the proof of the transformation. Figure 6 also proves that z_0 can be constructed with a AND-depth of 3 and so it is 3-representable.

This allows us to decompose the S-box into $H \circ G \circ F = \mathsf{S}_{222}$, where $F : \{0,1\}^8 \to \{0,1\}^8$, $G : \{0,1\}^8 \to \{0,1\}^9$ and $H : \{0,1\}^9 \to \{0,1\}^8$ are each quadratic S-boxes. The algebraic expressions are as follows:

$$\begin{aligned}
F(x_7, x_6, x_5, x_4, x_3, x_2, x_1, x_0) = \ & (u_7, u_6, u_5, u_4, u_3, u_2, u_1, u_0) \\
u_0 = x_4 + x_6x_7 + x_6 + x_7 + 1, \ & u_1 = x_0 + x_2x_3 + x_2 + x_3 + 1 \\
u_2 = x_0x_3 + x_0 + x_1 + x_2x_3 + x_2, \ & u_3 = x_1x_2 + x_1 + x_2 + x_6 + 1
\end{aligned}$$

$$
\begin{aligned}
&u_4 = x_2,\ u_5 = x_3,\ u_6 = x_5,\ u_7 = x_7\\
G(u_7,u_6,u_5,u_4,u_3,u_2,u_1,u_0) &= (v_8,v_7,v_6,v_5,v_4,v_3,v_2,v_1,v_0)\\
&v_0 = u_6 + u_0u_1 + u_0 + u_1 + 1,\ v_1 = u_5 + u_6u_0 + u_6 + u_0u_1 + u_1\\
&v_2 = u_2u_3,\ v_3 = u_0,\ v_4 = u_1,\ v_5 = u_2,\ v_6 = u_3,\\
&v_7 = u_4,\ v_8 = u_7\\
H(v_8,v_7,v_6,v_5,v_4,v_3,v_2,v_1,v_0) &= (z_7,z_6,z_5,z_4,z_3,z_2,z_1,z_0)\\
&z_0 = v_2v_0 + v_2 + v_5v_0 + v_8v_5 + v_6v_0 + v_6 + v_0 + v_8 + v_7,\\
&z_1 = v_8 + v_0v_6 + v_0 + v_6 + 1,\\
&z_2 = v_6,\ z_3 = v_5,\ z_4 = v_1,\ z_5 = v_4,\ z_6 = v_3,\ z_7 = v_0
\end{aligned}
$$

Note that the additional output bit $v_2 = u_2u_3$ roughly corresponds to the $\pi(e,c,0)$ node created at level 2, i.e. v_2 is the only non-linear term in $\pi(e,c,0)$. As can be seen that this output bit of S_2 is constructed by a standalone AND gate, and correction terms have to be added to construct a 3-input/3-output share TI of the SKINNY S-box. In the supplementary material [8], we present explicit algebraic expressions for all 3 shares of the S-boxes F, G and H. While non-completeness and correctness are easy to argue, we additionally argue uniformity of our construction too.

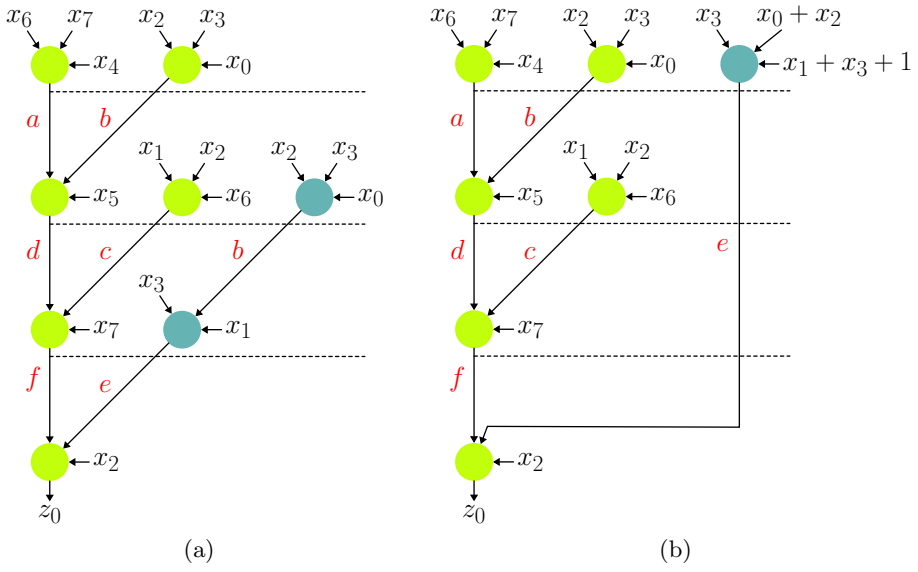

Fig. 5. Transformation (a)→(b) of the circuit graph of z_0 for Lemma 1.

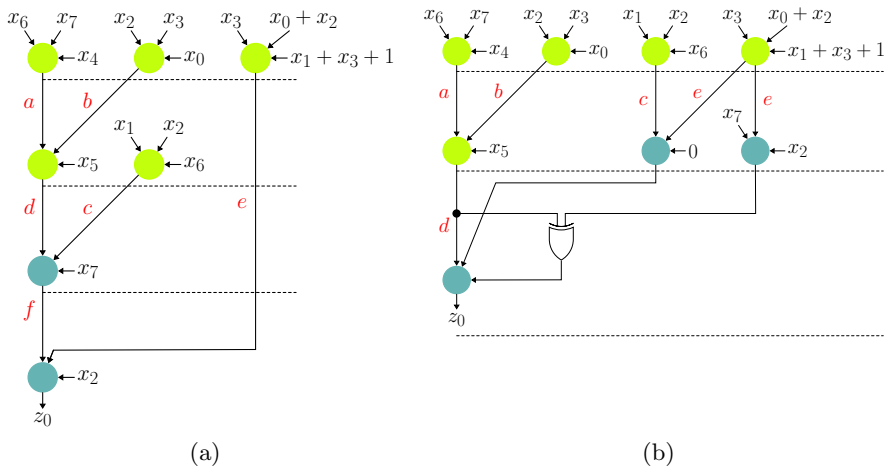

Fig. 6. Transformation (a)→(b) of the circuit graph of z_0 for Lemma 2, proving that z_0 is 3-representable of order 8 (right).

3.4 Decomposition into Two Cubic S-boxes

Note that it is straightforward to decompose S into two S-boxes of degree 4 each. For example from $\mathsf{S}_{2222} = I \circ H \circ G \circ F$, both $G \circ F$ and $I \circ H$ are degree 4 S-boxes. A first order TI of degree 4 S-box requires 5 shares. So by using the above decomposition we can implement a circuit that evaluates the shared S-box in only 2 clock cycles but requires 5 shares. Suppose we were able to decompose S into two cubic S-boxes: if this were so then a first order TI would need only 4 shares. Such a circuit would require smaller circuit area and hence consume less power on account of the reduced number of shares and also consume less energy to encrypt a plaintext on account of the reduced power consumption. So in principle it is an interesting exercise to see if this decomposition is at all possible.

In order to decompose S into two cubic S-boxes, we can again mount an exhaustive search on all partitions of two sets as done in Section 3.3. This procedure does not yield such a decomposition but many of the form $\mathsf{S} = G \circ F$ where $\deg(F) = 2$ and $\deg(G) = 4$ or vice-versa as shown in Figure 2b. However, we can follow a similar strategy as in detailed in the previous section. We begin with the following definition:

Definition 2. *A Boolean function B is said to have cubic depth 2, if it can be expressed as $B = C(c_1, c_2, \ldots, c_n)$ where $C, c_1, c_2, \ldots, c_n$ are each either cubic Boolean functions or functions of algebraic degree strictly less than 3. The cubic order of such a function is said to be i, if the total number cubic terms in the algebraic expressions of $C, c_1, c_2, \ldots, c_n$ combined is i.*

Note that lower cubic depth allows us to construct a TI of the given function lower number of cycles using only 4 shares. Since every cubic term wxy in the

algebraic expression has to be opened up as $(w_1+w_2+w_3+w_4)(x_1+x_2+x_3+x_4)(y_1+y_2+y_3+y_4)$ to construct a 4 share TI, a low cubic order will obviously help make the circuit more lightweight and efficient. It is straightforward to see that $z_1, z_2, \ldots, z_7$ all have cubic depth 2: z_2, z_3, z_5, z_6 are quadratic. z_7 has algebraic degree 4 and we have already seen that it is 2-representable, and so it automatically follows that its cubic depth is 2 and cubic order is 0. The fact that z_1, z_4 also have cubic depth equal to two can be seen in Figure 2b of the SKINNY S-box circuit. The part shaded in blue is an 8×8 quadratic S-box, call it $\mathsf{S}_{\mathsf{Blue}}$ and the part in red is another 8×8 S-box of degree 4 (call it $\mathsf{S}_{\mathsf{Red}}$). Note we obviously have $\mathsf{S} = \mathsf{S}_{\mathsf{Red}} \circ \mathsf{S}_{\mathsf{Blue}}$. The algebraic expressions are as follows:

$$
\begin{aligned}
&\mathsf{S}_{\mathrm{Blue}}(x_7, x_6, x_5, x_4, x_3, x_2, x_1, x_0) = (u_7, u_6, u_5, u_4, u_3, u_2, u_1, u_0)\\
&\quad u_7 = x_2,\ u_6 = x_3,\ u_5 = x_3x_2 + x_3x_0 + x_2 + x_1 + x_0,\ u_4 = x_7,\\
&\quad u_3 = x_6 + x_2x_1 + x_2 + x_1 + 1,\ u_2 = x_5,\\
&\quad u_1 = x_3x_2 + x_3 + x_2 + x_0 + 1,\ u_0 = x_7x_6 + x_7 + x_6 + x_4 + 1\\
&\mathsf{S}_{\mathrm{Red}}(u_7, u_6, u_5, u_4, u_3, u_2, u_1, u_0) = (z_7, z_6, z_5, z_4, z_3, z_2, z_1, z_0)\\
&\quad z_7 = u_2 + u_1u_0 + u_1 + u_0 + 1,\ z_6 = u_0,\ z_5 = u_1\\
&\quad z_4 = u_6 + u_2u_0 + u_2 + u_1u_0 + u_1,\ z_3 = u_5,\ z_2 = u_3,\\
&\quad z_1 = u_4 + u_3u_2 + u_3u_1u_0 + u_3u_1 + u_3u_0 + u_2 + u_1u_0 + u_1 + u_0,\\
&\quad z_0 = u_7 + u_5u_4 + u_5u_3u_2 + u_5u_3u_1u_0 + u_5u_3u_1 + u_5u_3u_0 + u_5u_2\\
&\qquad + u_5u_1u_0 + u_5u_1 + u_5u_0 + u_5 + u_4 + u_3u_2 + u_3u_1u_0\\
&\qquad + u_3u_1 + u_3u_0 + u_2 + u_1u_0 + u_1 + u_0 + 1
\end{aligned}
$$

From the expression we can see that z_1 as the output of $\mathsf{S}_{\mathsf{Red}}$ is a cubic function with only a single cubic term. And since the u_i's are at most quadratic this follows that the cubic depth of z_1 is 2 and its cubic order is 1. Also the expression for z_4 is quadratic in $\mathsf{S}_{\mathsf{Red}}$, which proves that not only is its cubic depth 2 and cubic order 0, but it is also 2-representable. It is elementary to verify that its AND-complexity is 3.

The only problematic part is proving that z_0 also has cubic depth 2 of some suitably low order, since it is not clear from this decomposition. Note that there is only one degree 4 term $u_5u_3u_1u_0$ in the expression of z_0. Also $u_5u_1 = x_3x_2x_1 + x_3x_1 + x_1 + x_1x_2 + x_0x_1$ is a cubic expression in the x_i's. Therefore, we construct the following S-box $\mathsf{S}'_{\mathsf{Blue}} : \{0,1\}^8 \to \{0,1\}^9$ where

$$\mathsf{S}'_{\mathsf{Blue}}(x_7, x_6, x_5, x_4, x_3, x_2, x_1, x_0) = (u_8, u_7, u_6, u_5, u_4, u_3, u_2, u_1, u_0)$$

such that $u_8 = x_3x_2x_1 + x_3x_1 + x_1 + x_1x_2 + x_0x_1$ and the other u_i's are as defined for $\mathsf{S}_{\mathsf{Blue}}$. Correspondingly we define $\mathsf{S}'_{\mathsf{Red}} : \{0,1\}^9 \to \{0,1\}^8$ where

$$\mathsf{S}'_{\mathsf{Red}}(u_8, u_7, u_6, u_5, u_4, u_3, u_2, u_1, u_0) = (z_7, z_6, z_5, z_4, z_3, z_2, z_1, z_0)$$

such that $z_0 = u_7 + u_5u_4 + u_5u_3u_2 + u_8u_3u_0 + u_5u_3u_1 + u_5u_3u_0 + u_5u_2 + u_5u_1u_0 + u_5u_1 + u_5u_0 + u_5 + u_4 + u_3u_2 + u_3u_1u_0 + u_3u_1 + u_3u_0 + u_2 + u_1u_0 + u_1 + u_0 + 1$ and the other z_i's are as defined for $\mathsf{S}_{\mathsf{Red}}$. Since both $\mathsf{S}'_{\mathsf{Blue}}$ and $\mathsf{S}'_{\mathsf{Red}}$ are cubic

S-boxes this proves that the cubic depth of z_0 is also 2. It is easy to count that there are 5 cubic terms in the modified expression of z_0 and one cubic term in the expression for u_8, which implies that the cubic order of z_0 is 6. Since we also have that $\mathsf{S} = \mathsf{S}'_{\mathsf{Red}} \circ \mathsf{S}'_{\mathsf{Blue}}$, this also gives us the cubic decomposition required to construct a first order TI using 4 input/output shares that can evaluate the shared S-box in just 2 cycles. In the supplementary material [8], we present explicit algebraic expressions for all 4 shares of the S-boxes $\mathsf{S}'_{\mathsf{Red}}, \mathsf{S}'_{\mathsf{Blue}}$, where we additionally argue uniformity of our construction too.

4 Implementation

After decomposing the S-box into quadratic and cubic component functions, we use the direct sharing approach to obtain the algebraic expressions for each of the individual shares of the masked S-box. In all cases, except for S_{2222}, correction terms were required to ensure uniform sharing (all the algebraic expressions for the individual shares can be found in [8]).

All the investigated schemes in this work have been synthesized on both ASIC and FPGA platforms. In particular, we used Synopsys Design Vision v2019.03 to synthesize the hardware description into a netlist via the `compile_ultra -no_autoungroup` directive that respects entity boundaries and thus prevents the optimizer from potentially interfering with the threshold properties of the circuit. Additionally, the power figures were obtained using back annotation of the switching activity onto the netlist performed by the Synopsys Power Compiler. In order to obtain a comprehensive set of measurements, our circuits were synthesized using three standard cell libraries of different sizes, namely the low-leakage TSMC 28 nm and UMC 65 nm libraries and the high-leakage NanGate 45 nm process.

In Table 1, we detail the measurements for the investigated S-box circuits and note that both in latency and power, S_{222} as well as S_{33} eclipse the other variants. This trend is amplified when the entire SKINNY circuit is implemented as shown in Table 2. We denote by $\mathsf{SKINNY}_{i_1 \dots i_k}$ the full SKINNY circuit using the S-box $\mathsf{S}_{i_1 \dots i_k}$.

The schemes have also been implemented on a 65 nm Xilinx Virtex-5 FPGA and a 45 nm Xilinx Spartan-6 FPGA using the Xilinx ISE synthesis and implementation tool. To prevent optimisations that might break the masking scheme, `DONT_TOUCH`, `KEEP`, and `KEEP_HIERARCHY` constraints have been added to the HDL source files. The resulting measurements are tabulated in Table 3.

5 Leakage Assessment

SILVER [16] is a formal verification tool for masking countermeasures. For a given security property [9], the tool exhaustive evaluates the input netlist using reduced-ordered binary decision diagrams. We compile the netlist for the S_{222} and S_{33} S-boxes using the NanGate 45nm standard cell library and verified that both netlists satisfied first-order probing security in the standard and robust

Table 1. ASIC synthesis measurements for the investigated substitution boxes.

Scheme	Library	Latency (Cycles)	Area (GE)	Timing (ns)	Power (μW)	
					10 MHz	100 MHz
S_{2222}	TSMC 28 nm	4	550.3	0.20	4.880	45.32
	NanGate 45 nm	4	584.3	0.24	43.81	157.1
	UMC 65 nm	4	597.9	1.15	5.735	56.14
S_{232}	TSMC 28 nm	3	922.0	0.50	6.490	59.17
	NanGate 45 nm	3	915.3	1.11	86.15	166.2
	UMC 65 nm	3	941.3	3.82	7.986	77.86
S_{222}	TSMC 28 nm	3	598.9	0.24	4.561	42.03
	NanGate 45 nm	3	600.6	0.31	46.77	154.4
	UMC 65 nm	3	616.5	1.73	5.395	52.63
S_{33}	TSMC 28 nm	2	1995	0.72	11.12	99.49
	NanGate 45 nm	2	1906	1.21	159.7	553.7
	UMC 65	2	1924	4.79	14.35	139.1

probing models as well as uniformity. A script together and the corresponding netlist files are given in the auxiliary repository [8].

5.1 *t*-tests

The TVLA methodology [12,24] provides a set of best-practice guidelines for performing non-invasive leakage detection on a device under test (DUT). To verify the security of our designs, we follow this approach using Welch's t-test and the min-p strategy for null hypothesis rejection. In particular, we perform non-specific fixed versus random t-tests, where we aim to determine the validity of the null hypothesis that *encryptions with a fixed and uniformly sampled plaintext admit the same mean power consumption* (i.e., are indistinguishable under first-order statistical analysis). Following the state of the art [1,24,30], we set a threshold $|t| > 4.5$ for any t-value to reject the null hypothesis.

To perform t-tests, power traces of SKINNY_{222} and SKINNY_{33} were measured using the Sakura-X and Sasebo-GII power side-channel leakage evaluation boards. These boards contain a core FPGA target on which a cryptographic circuit can be programmed, allowing the evaluation of custom hardware implementations of cryptographic primitives. To reduce noise, the boards contain an additional FPGA for communication with the host PC, which is used to send keys and plaintexts and read ciphertexts. Moreover, these boards contain direct connectors for oscilloscope probes, facilitating the acquisition of the power supply voltage traces for the side-channel evaluation. The encryption FPGA has

Table 2. ASIC synthesis figures for all investigated schemes for three cell libraries.

Scheme	Library	Latency	Area	Critical Path	Power (μW)		Energy (nJ/128 bits)	
		(Cycles)	(GE)	(ns)	10 MHz	100 MHz	10 MHz	100 MHz
SKINNY-128$_{2222}$	TSMC 28 nm	872	4461	0.31	21.91	186.2	1.911	1.623
Byte-Serial	NanGate 45 nm	872	5039	0.51	100.6	343.5	8.772	2.995
	UMC 65 nm	872	4989	1.59	25.82	244.5	2.251	2.132
SKINNY-256$_{2222}$	TSMC 28 nm	1040	5280	0.33	25.90	219.6	2.694	2.284
Byte-Serial	NanGate 45 nm	1040	5993	0.52	120.7	420.8	12.55	4.376
	UMC 65 nm	1040	5876	1.64	30.33	287.3	3.154	2.988
SKINNY-384$_{2222}$	TSMC 28 nm	1208	6122	0.35	26.97	222.5	3.258	2.688
Byte-Serial	NanGate 45 nm	1208	6949	0.57	140.3	496.4	16.94	5.993
	UMC 65 nm	1208	6782	1.69	34.98	333.1	4.226	4.024
SKINNY-128$_{2222}$	TSMC 28 nm	160	13671	0.35	80.01	707.0	1.280	1.131
	NanGate 45 nm	160	14637	0.47	917.3	2199	14.68	3.518
	UMC 65 nm	160	15116	2.03	93.57	898.7	1.497	1.438
SKINNY-256$_{2222}$	TSMC 28 nm	192	15197	0.36	88.13	776.9	1.692	1.491
	NanGate 45 nm	192	16315	0.47	1041	2490	19.98	4.781
	UMC 65 nm	192	16735	2.12	103.1	990.3	1.979	1.901
SKINNY-384$_{2222}$	TSMC 28 nm	224	16641	0.38	95.98	844.8	2.149	1.892
	NanGate 45 nm	224	17991	0.47	1166	2774	26.12	6.213
	UMC 65 nm	224	18357	2.12	113.4	1088	2.538	2.437
SKINNY-128$_{222}$	TSMC 28 nm	120	14452	0.44	77.58	683.3	0.931	0.819
	NanGate 45 nm	120	14899	0.66	474.9	1890	5.699	2.268
	UMC 65 nm	120	15413	3.40	93.05	892.7	1.156	1.071
SKINNY-256$_{222}$	TSMC 28 nm	144	15975	0.44	86.74	761.1	1.249	1.095
	NanGate 45 nm	144	16576	0.66	501.5	2010	7.222	2.894
	UMC 65 nm	144	17031	3.51	104.0	997.1	1.497	1.436
SKINNY-384$_{222}$	TSMC 28 nm	168	17484	0.44	95.51	838.7	1.604	1.410
	NanGate 45 nm	168	18253	0.66	632.1	2298	10.62	3.861
	UMC 65 nm	168	18654	3.51	115.6	1109	1.942	1.863
SKINNY-128$_{33}$	TSMC 28 nm	80	24375	0.66	114.7	988.5	0.917	0.791
	NanGate 45 nm	80	23954	0.88	980.1	3200	7.841	2.560
	UMC 65 nm	80	24923	4.13	139.1	1391	1.113	1.113
SKINNY-256$_{33}$	TSMC28 nm	96	26192	0.66	126.3	1090	1.212	1.046
	NanGate 45 nm	96	25888	0.87	1109	3678	10.64	3.531
	UMC 65 nm	96	26767	4.23	159.3	1542	1.529	1.480
SKINNY-384$_{33}$	TSMC 28 nm	112	27964	0.66	137.5	1190	1.540	1.333
	NanGate 45 nm	112	27820	0.87	1382	4001	15.48	4.481
	UMC 65 nm	112	28621	4.24	147.7	1636	1.654	1.832

direct connections to header pins on the board, allowing easy synchronisation using a dedicated trigger signal.

The Sakura-X board contains a more recent FPGA from the Xilinx 7-Series (Kintex-7, XC7K160T), while the Sasebo-GII board contains an older FPGA from the 5-Series (Virtex-5, XC5VLX30) architecture. To prevent unwanted optimizations during the FPGA toolchain synthesis and implementation, `DONT_TOUCH`, `KEEP_HIERARCHY`, and `KEEP` constraints are added. The clock frequency of our designs is constrained to a low 3 MHz on both boards. All power measure-

Table 3. Xilinx Virtex-5 and Spartan-6 substitution and cipher synthesis results.

Scheme	Target	Slices	Flip-Flops	Lookup Tables	Max. Frequency (MHz)
S_{2222}	Virtex-5	35	72	24	600
	Spartan-6	41	72	32	375
S_{222}	Virtex-5	30	51	46	472
	Spartan-6	42	51	52	316
S_{33}	Virtex-5	106	36	296	278
	Spartan-6	178	36	300	202
$\mathsf{SKINNY\text{-}128}_{2222}$	Virtex-5	1348	1672	1514	280
	Spartan-6	2204	1672	928	194
$\mathsf{SKINNY\text{-}128}_{222}$	Virtex-5	956	1337	1689	250
	Spartan-6	791	1328	1619	105
$\mathsf{SKINNY\text{-}128}_{33}$	Virtex-5	2883	1224	5834	180
	Spartan-6	2640	1216	5641	110

ments are performed using a Tektronix MDO3104 oscilloscope with a sampling rate of 1 GS/s and AC coupling; we take 10000 sample points per trace with 1 microsecond horizontal graduations.

To perform non-specific t-tests, all encryptions were performed with a fixed key. The cryptographic primitive was reset before every encryption to ensure identical initial conditions for both the fixed and random traces. Consequently, this allowed us to record traces for t-tests in a deterministic interleaving fashion, where a random plaintext preceded a fixed plaintext and vice-versa, reducing bias in any one dataset from potential variation in noise and environmental conditions over time. To avoid leakage arising from generating random masks on the DUT itself, we sent pre-masked plaintext shares to the FPGA.

In order to verify the soundness of our experimental setup, we first ran t-tests in the *masks off* setting by setting all but one share of the plaintext to the zero vector. We perform the masks off t-tests on 10000 traces for each design. Figures 10a and 10b plot a sample trace for the two designs. Note that we take traces corresponding to 10 rounds of an encryption operation in each experiment. Recall that executing a round of SKINNY with S_{33} uses two cycles, rather than three like with S_{222}. The encryption operation for the SKINNY_{33} experiments only begins after a few thousand data points, whereas we record from the beginning of an encryption for the SKINNY_{222} experiments.

The results in Figures 7a and 8a indicate that there is potentially exploitable leakage with just 10000 traces, even with measurements with low SNR taken on the Sakura-X board. We then record 1 million traces with randomly generated masks to assess the first-order security of our designs (Figures 7b and 8b). Our results indicate that the threshold of 4.5 is not crossed in any of the trace samples, and that no leakage is detected with this number of traces. Since Threshold Implementations are well-studied, we expect these results to hold with a larger number of traces also.

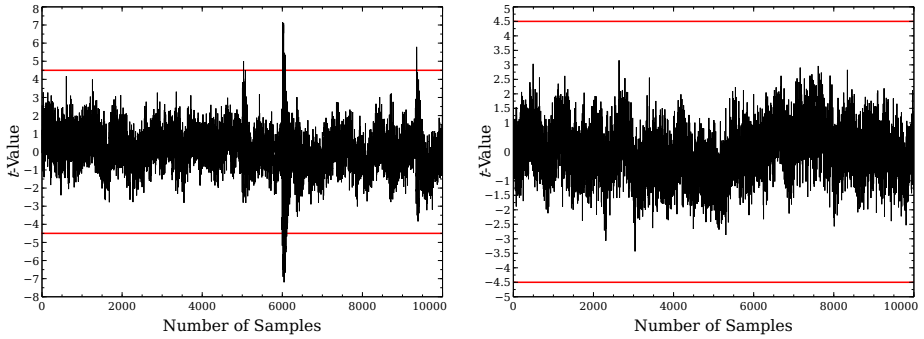

Fig. 7. t-test results for SKINNY_{222} on the Sakura-X with (a) 10000 traces and masks off and (b) one million traces and masks on.

To demonstrate that our Threshold Implementation of SKINNY_{222} is secure even on a smaller FPGA with a higher SNR (lower noise), we also performed t-tests with both randomly generated and zero masks using the Sasebo-II side-channel evaluation board. Figure 10c shows a sample trace taken during the experiments, where the power consumption from the encryption operation in each clock cycle is clearly visible. Figure 9 shows the t-values obtained for the power traces. As before, with 10000 traces in the masks off setting, we note substantial leakage. With one million traces and masks on, we find no evidence of leakage.

6 Conclusion and Future Work

In this work, we re-envision first-order TI for the SKINNY family of tweakable block ciphers in the round-based setting. More specifically, we propose different decompositions of the 8-bit S-box which enable significantly more efficient implementations of a protected SKINNY circuit in terms of latency and energy consumption, which we demonstrate through an extensive suite of synthesis benchmarks. We conclude by assessing the security of our designs via leveraging existing leakage detection and formal verification techniques. In terms of future work, we identify the following problems as of particular interest:

- *Higher-Order Schemes.* This paper covers first-order realizations but against a more capable adversary, security against higher-order attacks is required. As TI schemes become increasingly expensive in this setting, a suitable candidate approach is $d+1$ sharing e.g., using Domain-Oriented Masking [13] to reduce the number of required shares.
- *Area Optimizations.* Although S_{222} and S_{33} optimize for latency and energy consumption in comparison to S_{2222}, their circuit area is roughly the same

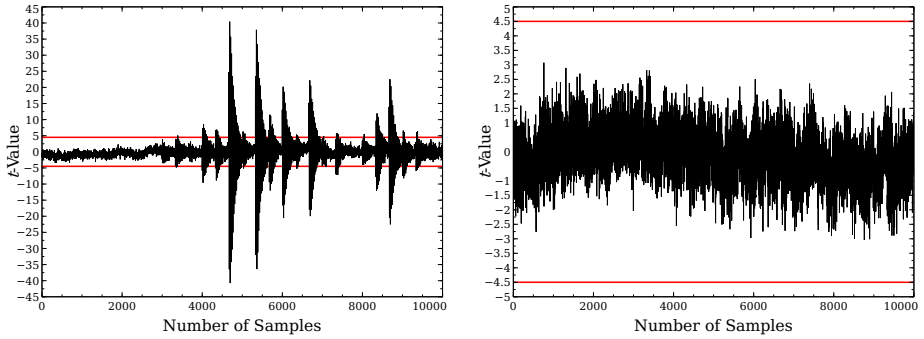

Fig. 8. t-test results for SKINNY_{33} on the Sakura-X with (a) 10000 traces and masks off and (b) one million traces and masks on.

or moderately larger. It is thus an interesting exercise to determine whether the area footprint can be reduced as well.

Acknowledgements. We wish to thank the anonymous reviewers whose comments helped improve this work. Subhadeep Banik is supported by the Swiss National Science Foundation (SNSF) through the Ambizione Grant `PZ00P2_-179921`.

A Algebraic expressions for SKINNY S-box S

$$
\begin{aligned}
z_0 = {} & x_7x_6x_3x_2x_1x_0 + x_7x_6x_3x_2x_0 + x_7x_6x_3x_2 + x_7x_6x_3x_1x_0 + x_7x_6x_2x_1 + \\
& x_7x_6x_1x_0 + x_7x_3x_2x_1x_0 + x_7x_3x_2x_0 + x_7x_3x_1x_0 + x_7x_3x_0 + x_7x_2x_1 + \\
& x_7x_2 + x_7x_1x_0 + x_7x_1 + x_7x_0 + x_7 + x_6x_5x_3x_2 + x_6x_5x_3x_0 + x_6x_5x_2 + \\
& x_6x_5x_1 + x_6x_5x_0 + x_6x_5 + x_6x_4x_3x_2x_1 + x_6x_4x_3x_1 + x_6x_4x_3x_0 + \\
& x_6x_4x_3 + x_6x_4x_2x_1 + x_6x_4x_1x_0 + x_6x_3x_2x_1x_0 + x_6x_3x_2x_1 + x_6x_3x_2x_0 + \\
& x_6x_3x_1x_0 + x_6x_3x_1 + x_6x_3 + x_6x_2 + x_6x_1 + x_6x_0 + x_6 + x_5x_3x_2x_1x_0 + \\
& x_5x_3x_2x_0 + x_5x_3x_2 + x_5x_3x_1x_0 + x_5x_2x_1x_0 + x_5x_2x_1 + x_5x_2x_0 + \\
& x_5x_1x_0 + x_4x_3x_2x_1x_0 + x_4x_3x_2x_0 + x_4x_3x_2 + x_4x_3x_1x_0 + x_4x_2x_1 + \\
& x_4x_1x_0 + x_3x_2x_1x_0 + x_3x_2x_0 + x_3x_1x_0 + x_3x_0 + x_2x_1x_0 + x_2x_1 + \\
& x_2x_0 + x_1x_0 + x_1 + x_0 + 1 \\
z_1 = {} & x_7x_6x_3x_2x_1 + x_7x_6x_3x_1 + x_7x_6x_2x_1x_0 + x_7x_6x_2x_0 + x_7x_6x_2 + \\
& x_7x_6x_1x_0 + x_7x_3x_2x_1 + x_7x_3x_1 + x_7x_2x_1x_0 + x_7x_2x_0 + x_7x_2 + \\
& x_7x_1x_0 + x_7 + x_6x_5 + x_6x_4x_3x_2 + x_6x_4x_3 + x_6x_4x_2 + x_6x_4x_0 + \\
& x_6x_3x_2x_1 + x_6x_3x_2 + x_6x_3x_1 + x_6x_3 + x_6x_2x_1x_0 + x_6x_2x_0 +
\end{aligned}
$$

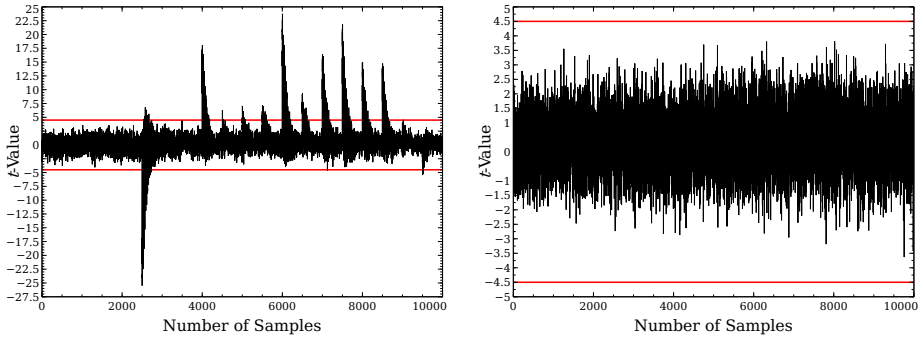

Fig. 9. t-test results for SKINNY_{222} on the Sasebo-II with (a) 10000 traces and masks off and (b) one million traces and masks on.

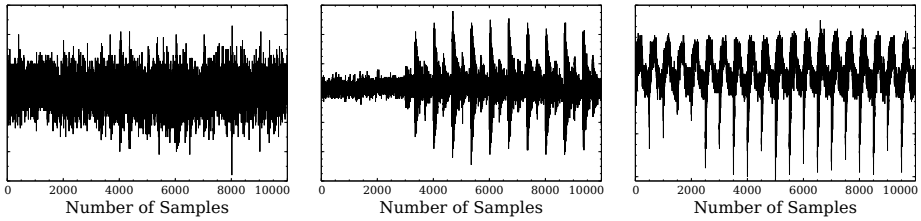

Fig. 10. Sample power traces of encryption operations for (a) SKINNY_{222} on the Sakura-X, (b) SKINNY_{33} on the Sakura-X and (c) SKINNY_{222} on the Sasebo-II.

$$
\begin{aligned}
&x_6x_1x_0 + x_6x_0 + x_6 + x_5x_2x_1 + x_5x_2 + x_5x_1 + x_4x_3x_2x_1 + x_4x_3x_1 +\\
&x_4x_2x_1x_0 + x_4x_2x_0 + x_4x_2 + x_4x_1x_0 + x_2x_1 + x_2 + x_1\\
z_2 = {}& x_6 + x_2x_1 + x_2 + x_1 + 1,\ z_3 = x_3x_2 + x_3x_0 + x_2 + x_1 + x_0\\
z_4 = {}& x_7x_6x_5 + x_7x_6x_3x_2 + x_7x_6x_3 + x_7x_6x_2 + x_7x_6x_0 + x_7x_6 + x_7x_5 +\\
&x_7x_3x_2 + x_7x_3 + x_7x_2 + x_7x_0 + x_7 + x_6x_5 + x_6x_3x_2 + x_6x_3 + x_6x_2 +\\
&x_6x_0 + x_6 + x_5x_4 + x_4x_3x_2 + x_4x_3 + x_4x_2 + x_4x_0 + x_4 + x_3\\
z_5 = {}& x_3x_2 + x_3 + x_2 + x_0 + 1,\ z_6 = x_7x_6 + x_7 + x_6 + x_4 + 1\\
z_7 = {}& x_7x_6x_3x_2 + x_7x_6x_3 + x_7x_6x_2 + x_7x_6x_0 + x_7x_3x_2 + x_7x_3 + x_7x_2 +\\
&x_7x_0 + x_6x_3x_2 + x_6x_3 + x_6x_2 + x_6x_0 + x_5 + x_4x_3x_2 + x_4x_3 +\\
&x_4x_2 + x_4x_0
\end{aligned}
$$

References

1. Arribas, V., Bilgin, B., Petrides, G., Nikova, S., Rijmen, V.: Rhythmic Keccak: SCA security and low latency in HW. IACR Transactions on Cryptographic Hardware and Embedded Systems **2018**(1), 269–290 (2018). https://doi.org/10.13154/tches.v2018.i1.269-290, https://tches.iacr.org/index.php/TCHES/article/view/840
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.) Advances in Cryptology – CRYPTO 2016, Part II. Lecture Notes in Computer Science, vol. 9815, pp. 123–153. Springer, Heidelberg, Germany, Santa Barbara, CA, USA (Aug 14–18, 2016). https://doi.org/10.1007/978-3-662-53008-5_5
3. Bilgin, B.: Threshold implementations: as countermeasure against higher-order differential power analysis. Ph.D. thesis, University of Twente, Netherlands (May 2015). https://doi.org/10.3990/1.9789036538916, cum laude
4. Bilgin, B., Gierlichs, B., Nikova, S., Nikov, V., Rijmen, V.: Higher-order threshold implementations. In: Sarkar, P., Iwata, T. (eds.) Advances in Cryptology – ASIACRYPT 2014, Part II. Lecture Notes in Computer Science, vol. 8874, pp. 326–343. Springer, Heidelberg, Germany, Kaoshiung, Taiwan, R.O.C. (Dec 7–11, 2014). https://doi.org/10.1007/978-3-662-45608-8_18
5. Bilgin, B., Nikova, S., Nikov, V., Rijmen, V., Stütz, G.: Threshold implementations of all 3×3 and 4×4 S-boxes. In: Prouff, E., Schaumont, P. (eds.) Cryptographic Hardware and Embedded Systems – CHES 2012. Lecture Notes in Computer Science, vol. 7428, pp. 76–91. Springer, Heidelberg, Germany, Leuven, Belgium (Sep 9–12, 2012). https://doi.org/10.1007/978-3-642-33027-8_5
6. Brier, E., Clavier, C., Olivier, F.: Correlation power analysis with a leakage model. In: Joye, M., Quisquater, J.J. (eds.) Cryptographic Hardware and Embedded Systems – CHES 2004. Lecture Notes in Computer Science, vol. 3156, pp. 16–29. Springer, Heidelberg, Germany, Cambridge, Massachusetts, USA (Aug 11–13, 2004). https://doi.org/10.1007/978-3-540-28632-5_2
7. Caforio, A., Balli, F., Banik, S.: Energy analysis of lightweight AEAD circuits. In: Krenn, S., Shulman, H., Vaudenay, S. (eds.) CANS 20: 19th International Conference on Cryptology and Network Security. Lecture Notes in Computer Science, vol. 12579, pp. 23–42. Springer, Heidelberg, Germany, Vienna, Austria (Dec 14–16, 2020). https://doi.org/10.1007/978-3-030-65411-5_2
8. Caforio, A., Collins, D., Glamocanin, O., Banik, S.: Improving First-Order Threshold Implementations of SKINNY (Repository) (10 2021), https://github.com/qantik/skinny-dipping
9. De Meyer, L., Bilgin, B., Reparaz, O.: Consolidating security notions in hardware masking. IACR Transactions on Cryptographic Hardware and Embedded Systems **2019**(3), 119–147 (2019). https://doi.org/10.13154/tches.v2019.i3.119-147, https://tches.iacr.org/index.php/TCHES/article/view/8291
10. 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
11. 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 Transactions on Cryptographic Hardware and Embedded Systems **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
12. Gilbert Goodwill, B.J., Jaffe, J., Rohatgi, P., et al.: A testing methodology for side-channel resistance validation. In: NIST non-invasive attack testing workshop. vol. 7, pp. 115–136 (2011)
13. 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
14. Guo, C., Iwata, T., Khairallah, M., Minematsu, K., Peyrin, T.: Romulus v1.3. Tech. rep. (2021)
15. Ishai, Y., Sahai, A., Wagner, D.: Private circuits: Securing hardware against probing attacks. In: Boneh, D. (ed.) Advances in Cryptology – CRYPTO 2003. Lecture Notes in Computer Science, vol. 2729, pp. 463–481. Springer, Heidelberg, Germany, Santa Barbara, CA, USA (Aug 17–21, 2003). https://doi.org/10.1007/978-3-540-45146-4_27
16. Knichel, D., Sasdrich, P., Moradi, A.: SILVER - statistical independence and leakage verification. In: Moriai, S., Wang, H. (eds.) Advances in Cryptology – ASIACRYPT 2020, Part I. Lecture Notes in Computer Science, vol. 12491, pp. 787–816. Springer, Heidelberg, Germany, Daejeon, South Korea (Dec 7–11, 2020). https://doi.org/10.1007/978-3-030-64837-4_26
17. Kocher, P.C., Jaffe, J., Jun, B.: Differential power analysis. In: Wiener, M.J. (ed.) Advances in Cryptology – CRYPTO'99. Lecture Notes in Computer Science, vol. 1666, pp. 388–397. Springer, Heidelberg, Germany, Santa Barbara, CA, USA (Aug 15–19, 1999). https://doi.org/10.1007/3-540-48405-1_25
18. Moradi, A., Standaert, F.X.: Moments-correlating dpa. In: Proceedings of the 2016 ACM Workshop on Theory of Implementation Security. pp. 5–15 (2016)
19. Nikova, S., Rechberger, C., Rijmen, V.: Threshold implementations against side-channel attacks and glitches. In: Ning, P., Qing, S., Li, N. (eds.) ICICS 06: 8th International Conference on Information and Communication Security. Lecture Notes in Computer Science, vol. 4307, pp. 529–545. Springer, Heidelberg, Germany, Raleigh, NC, USA (Dec 4–7, 2006)
20. Nikova, S., Rijmen, V., Schläffer, M.: Secure hardware implementation of nonlinear functions in the presence of glitches. Journal of Cryptology **24**(2), 292–321 (Apr 2011). https://doi.org/10.1007/s00145-010-9085-7
21. Poschmann, A., Moradi, A., Khoo, K., Lim, C.W., Wang, H., Ling, S.: Side-channel resistant crypto for less than 2,300 GE. Journal of Cryptology **24**(2), 322–345 (Apr 2011). https://doi.org/10.1007/s00145-010-9086-6
22. Reparaz, O.: A note on the security of higher-order threshold implementations. Cryptology ePrint Archive, Report 2015/001 (2015), https://eprint.iacr.org/2015/001
23. Reparaz, O., Bilgin, B., Nikova, S., Gierlichs, B., Verbauwhede, I.: Consolidating masking schemes. In: Gennaro, R., Robshaw, M.J.B. (eds.) Advances in Cryptology – CRYPTO 2015, Part I. Lecture Notes in Computer Science, vol. 9215, pp. 764–783. Springer, Heidelberg, Germany, Santa Barbara, CA, USA (Aug 16–20, 2015). https://doi.org/10.1007/978-3-662-47989-6_37
24. Schneider, T., Moradi, A.: Leakage assessment methodology - A clear roadmap for side-channel evaluations. In: Güneysu, T., Handschuh, H. (eds.) Cryptographic

Hardware and Embedded Systems – CHES 2015. Lecture Notes in Computer Science, vol. 9293, pp. 495–513. Springer, Heidelberg, Germany, Saint-Malo, France (Sep 13–16, 2015). https://doi.org/10.1007/978-3-662-48324-4_25
25. Shahmirzadi, A.R., Božilov, D., Moradi, A.: New first-order secure AES performance records. IACR Transactions on Cryptographic Hardware and Embedded Systems **2021**(2), 304–327 (2021). https://doi.org/10.46586/tches.v2021.i2.304-327, https://tches.iacr.org/index.php/TCHES/article/view/8796
26. Sönmez Turan, M., McKay, K., Chang, D., Çalık, Ç., Bassham, L., Kang, J., Kelsey, J.: Status report on the second round of the nist lightweight cryptography standardization process. Tech. rep., National Institute of Standards and Technology (2021)
27. Sugawara, T.: 3-share threshold implementation of AES s-box without fresh randomness. IACR Transactions on Cryptographic Hardware and Embedded Systems **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
28. Wegener, F., Baiker, C., Moradi, A.: Shuffle and mix: On the diffusion of randomness in threshold implementations of Keccak. In: Polian, I., Stöttinger, M. (eds.) COSADE 2019: 10th International Workshop on Constructive Side-Channel Analysis and Secure Design. Lecture Notes in Computer Science, vol. 11421, pp. 270–284. Springer, Heidelberg, Germany, Darmstadt, Germany (Apr 3–5, 2019). https://doi.org/10.1007/978-3-030-16350-1_15
29. Wegener, F., De Meyer, L., Moradi, A.: Spin me right round rotational symmetry for FPGA-specific AES: Extended version. Journal of Cryptology **33**(3), 1114–1155 (Jul 2020). https://doi.org/10.1007/s00145-019-09342-y
30. Zarei, S., Shahmirzadi, A.R., Soleimany, H., Salarifard, R., Moradi, A.: Low-latency keccak at any arbitrary order. IACR Transactions on Cryptographic Hardware and Embedded Systems pp. 388–411 (2021)