An Optimal Universal Construction for the Threshold Implementation of Bijective S-boxes

Enrico Piccione[1], Samuele Andreoli[1], Lilya Budaghyan[1], Claude Carlet[1,2], Siemen Dhooghe[3], Svetla Nikova[1,3], and George Petrides[4], Vincent Rijmen[1,3]

[1] University of Bergen, Bergen, Norway, {name.surname}@uib.no
[2] University of Paris 8, Saint-Denis, France, claude.carlet@gmail.com
[3] KU Leuven, Leuven, Belgium, {name.surname}@esat.kuleuven.be
[4] Univeristy of Cyprus, Nicosia, Cyprus g.petrides@yahoo.com

Abstract. Threshold implementation is a method based on secret sharing to secure cryptographic ciphers (and in particular S-boxes) against differential power analysis. Until now, threshold implementations were only constructed for specific types of functions and some small S-boxes, but no general construction for all S-boxes was ever presented. The lower bound for the number of shares of threshold implementation is $t+1$, where t is the algebraic degree of the S-box. Since the smallest number of shares $t+1$ is not possible for all S-Boxes, as proven by Bilgin et al. in 2015, then there does not exist a universal construction with $t+1$ shares. Hence, if there is a universal construction working for all permutations then it should work with at least $t+2$ shares. In this paper, we present the first optimal universal construction with $t+2$ shares. This construction enables low latency hardware implementations without the need for randomness. In particular, we apply this result to find the first two uniform sharings of the AES S-box. Area and performance figures for hardware implementations are provided.

Keywords: AES, DPA, Glitches, Masking, Permutation Polynomials, Sharing, Threshold Implementations, Vectorial Boolean Functions

1 Introduction

In 1999, Kocher et al. [KJJ99] introduced Differential Power Analysis (DPA), an attack which uses a physical device's properties such as its power consumption to retrieve the secret keys of the embedded cryptographic algorithms, originally demonstrated on DES. The weakness is not in the design of DES, but rather in its implementation. As a result, side-channel countermeasures were developed, aiming for the secure implementation of symmetric primitives, including standards such as the Advanced Encryption Standard (AES) [DR01]. Since then, a lot of research has been done to improve the side-channel attacks and to develop protection mechanisms against them. The most prominent countermeasure in both academia and industry is called *sharing* (or also called *masking*) and was introduced by Goubin and Patarin [GP99] and Chari et al. [CJRR99] independently in the same year. In s-share Boolean sharing, a variable $x \in \mathbb{F}_2$ is shared as s values $(x_1, ..., x_s) \in \mathbb{F}_2^s$, such that $\sum_{i=1}^{s} x_i = x$. While masking helps to protect algorithms against formally defined adversary models, the method's protection is not evident when translated to a physical device. For example, the transient behaviour of values on hardware (also called glitches) can undermine a naive sharing's security. This leads to several attacks on shared AES implementations in hardware by Mangard et al. [MPO05] in CHES 2005.

The next year, in 2006, Nikova, Rechberger and Rijmen [NRR06] published a methodology to build a countermeasure, called *Threshold Implementations* (TI), which addresses

this physical behaviour on hardware. The TI method adds two important properties over regular sharing. First, it requires that a sharing is *non-complete*. In its simplest form, non-completeness is satisfied if each coordinate function characterising the combinatorial logic between registers does not operate on all shares of a secret or, in other words, each share is absent from at least one coordinate function. Second, a sharing has to be *uniform* which, if the underlying function is a permutation, means that the shared function is a bijection. Given the above properties, a sharing ensures protection against *first-order attacks* where the mean of the power traces is used as a distinguisher. To help mitigate higher-order attacks (which use higher-order moments or mixed-order moments), Bilgin et al. [BGN+14] in Asiacrypt 2014 introduced an extension of threshold implementation properties, namely higher-order non-completeness. A lower bound on the number of shares $s \geq \deg(F) \times d + 1$ (where d is the security order) to achieve a threshold implementation is given in [NRR06, BGN+14, Pet19]. In Crypto 2015, Reparaz et al.[RBN+15] introduced another flavour of the TI method which requires only $d + 1$ shares, called CMS. For the rest of the paper, we focus only on the initial one with $s \geq \deg(F) \times d + 1$ shares.

The TI approach has been applied to many symmetric primitives. For example, on the AES by De Cnudde et al. [CRB+16] and by Moradi et al. [MPL+11]; on PRESENT by Poschmann et al. [PMK+11]; on Keccak by Groß et al. [GSM17]; etc. Each of these papers have implemented the countermeasure and verified its order of security in practice. The trust in the TI method as a countermeasure against side-channel analysis is based on the significant quantity of works with proven practical results.

Threshold implementations have proven to be an effective countermeasure, but finding a sharing which is both non-complete and uniform is non-trivial. There are three known methods to achieve such a sharing. The first is using *direct sharing* by Bilgin et al. [BNN+12] to guarantee the non-completeness of the sharing and apply *correction terms* [NRR06] for uniformity. However, this method does not guarantee success for any arbitrary permutation. The second method uses direct sharing, but adds fresh randomness as a re-sharing step to guarantee the uniformity. The third method is introduced by Daemen [Dae17] and is called *the changing of the guards*. It embeds the non-complete sharing in a Feistel construction to guarantee the uniformity.

There is currently no known universal construction of a threshold implementation for arbitrary permutations of any size. Instead, research on threshold implementations focuses on finding solutions for small sizes. It has been proven by Bilgin et al. [BNN+12] in CHES 2012 that threshold implementation is invariant up to affine equivalence. Then, using the classification of the affine equivalent classes for 3 and 4 bit permutations, Bilgin et al. provided threshold implementations for all of these classes. Later, Bozilov et al. [BBS17] in FSE 2017 and De Meyer and Bilgin [MB19] in FSE 2019 provided threshold implementations of 5 and 6 bits quadratic permutations. In Table 1.1 we summarise the results on threshold implementations found in [BNN+12] and [BBS17]. For each size, we report the number of S-boxes for which a threshold implementation with s shares is known. If a s-shares threshold implementation was not found for an S-box, then we report the length of the decomposition used to achieve a uniform sharing of the S-box. Note that for 5-bit only quadratic S-boxes are presented here.

There are several ways to achieve a TI sharing of the AES S-box, but the most relevant for our considerations is the one started by Wegener and Moradi [WM18] who gave a decomposition of the AES S-box into two cubic power functions, namely $x \mapsto x^{26}$ and $x \mapsto x^{49}$. The following year, the list of all possible decompositions on quadratic and cubic power functions for the inversion over any binary field up to $n = 16$ was given by Nikova et al. [NNR19]. The list has recently been extended up to $n = 32$ by Petrides [Pet22]. In particular, for the inversion in $\mathbb{F}_{2^8}$, the decomposition in power functions up to algebraic degree three was presented.

Starting from n-bit bijective S-boxes, Varici et al. [VNNR19] construct new $(n + 1)$-bit

Table 1.1: Known threshold implementations of S-boxes of 3, 4, and 5 bit.

size	degree	3 shares				4 shares			5 shares
		1	2	3	4	1	2	3	1
3	2	2	2	1		3			3
4	2	5	1			6			6
	3		28	115	1	4	52	239	295
5	2	30				75			

and $(n+2)$-bit bijective S-boxes. The authors show that, if a threshold implementation for the n-bit bijective S-boxes exist, then the constructed $(n+1)$-bit and $(n+2)$-bit bijective S-boxes also have a threshold implementation.

Contributions. In the present paper, we construct a first-order threshold implementation with $t+2$ shares for every bijective S-box of any algebraic degree $t \geq 2$. Since the theoretically smallest number of shares $t+1$ is not possible for all S-boxes, as proven in [BNN+15], then there does not exist a universal construction with $t+1$ shares. Hence, our construction is the first optimal universal TI construction and it enables low latency hardware implementations without the need for randomness. We demonstrate this by providing the first threshold implementations (direct and by using decomposition) of the AES S-box that are uniform by construction, comparing them to the state of the art.
A secondary contribution of our result is to the theory of permutation polynomials, since we are able to construct a new infinite family of permutations.
In the end, we conjecture that power permutations and APN permutations (functions with optimal resistance to differential cryptanalysis) do not admit a threshold implementation with the theoretically smallest number of shares. Hence, our construction would be optimal in those relevant cases.

Paper Outline. Section 2 introduces the notations and the main concepts used in the paper, such as Boolean functions and threshold implementations. Section 3 covers the main result of the paper, which is an optimal universal construction for the threshold implementation of bijective S-boxes. Section 4 applies the construction to achieve two uniform threshold implementations of the AES S-box, where performance results in hardware are given. Finally, Section 5 concludes the paper and presents conjectures on non-existence of threshold implementations with $t+1$ shares for power and APN permutations.

2 Preliminaries

In this section, we are going to provide a prelude about Boolean functions, secret sharing, and threshold implementation.

2.1 Boolean Functions

Let n and s be positive integers. We denote by $\mathbb{F}_2$, respectively, by $\mathbb{F}_{2^n}$ the finite field with 2, respectively, 2^n elements and by $\mathbb{F}_2^s$, respectively, by $\mathbb{F}_{2^n}^s$ the s-dimensional vector space over $\mathbb{F}_2$, respectively, over $\mathbb{F}_{2^n}$.

A *Boolean function* f in n variables is an $\mathbb{F}_2$-valued function on $\mathbb{F}_2^n$. The unique

representation of f as a polynomial over $\mathbb{F}_2$ in n variables of the form

$$f(x_1, \ldots, x_n) = \sum_{u \in \mathbb{F}_2^n} c(u) \left(\prod_{i=1}^{n} x_i^{u_i} \right), \quad c(u) \in \mathbb{F}_2$$

is called the *algebraic normal form* of f. The global degree of the algebraic normal form of f is denoted by $\deg(f)$ and is called the *algebraic degree* of the function f [Car20]. A Boolean function f is *affine*, *quadratic*, or *cubic* if its algebraic degree is respectively less than or equal to 1, 2, or 3. Moreover, f is *linear* if it is affine and $f(0) = 0$. The *Hamming weight* $wt(f)$ of a Boolean function f is the size of its *support* $\{x \in \mathbb{F}_2^n \colon f(x) \neq 0\}$. A Boolean function F is called *balanced* if $wt(f) = 2^{n-1}$.

Any function F from $\mathbb{F}_2^n$ into $\mathbb{F}_2^m$ can be considered as a *vectorial Boolean function*, i.e. F can be presented in the form

$$F(x_1, \ldots, x_n) = (F_1(x_1, \ldots, x_n), \ldots, F_m(x_1, \ldots, x_n)),$$

where the Boolean functions $F_1, \ldots, F_m$ are called the *coordinate functions*. A *component function* of F is any linear combination of its coordinate functions. The algebraic degree of F is equal to the maximum algebraic degree of the coordinate functions of F (see [Car20]). A vectorial Boolean function F is *affine*, *quadratic*, or *cubic* if its algebraic degree is respectively less than or equal to 1, 2, or 3. We define the derivative of F in the direction of the vector $a \in \mathbb{F}_2^n$ as $D_a F(x) = F(x) + F(x + a)$.

If we identify $\mathbb{F}_2^n$ with the finite field $\mathbb{F}_{2^n}$, then any function $F \colon \mathbb{F}_{2^n} \to \mathbb{F}_{2^n}$ is also uniquely represented as a univariate polynomial over $\mathbb{F}_{2^n}$ of degree smaller than 2^n

$$F(x) = \sum_{i=0}^{2^n - 1} c_i x^i, \quad c_i \in \mathbb{F}_{2^n}.$$

For any integer k, $0 \leq k \leq 2^n - 1$, the number $w_2(k)$ of non-zero coefficients k_s, $0 \leq k_s \leq 1$, in the binary expansion $\sum_{s=0}^{n-1} 2^s k_s$ of k is called the 2-weight of k. The algebraic degree of a function $F \colon \mathbb{F}_{2^n} \to \mathbb{F}_{2^n}$ is equal to the maximum 2-weight of the exponents i of the polynomial $F(x)$ such that $c_i \neq 0$ (see [CCZ98]), that is

$$\deg(F) = \max_{\substack{0 \leq i \leq 2^n - 1 \\ c_i \neq 0}} w_2(i).$$

In particular, F is linear if and only if $F(x)$ is a *linearized polynomial* over $\mathbb{F}_{2^n}$ that is of the form:

$$\sum_{i=0}^{n-1} c_i x^{2^i}, \quad c_i \in \mathbb{F}_{2^n}.$$

Let F be a function from $\mathbb{F}_2^n$ to itself. Then, F is called a *permutation* if it is bijective. In this case, the transformation

$$F \mapsto F^{-1}$$

is called the *inverse transformation* (or simply inversion). A function $F \colon \mathbb{F}_2^n \to \mathbb{F}_2^m$ is called *balanced* if $n \geq m$ and F takes every value of $\mathbb{F}_2^n$ the same number 2^{n-m} of times. The balanced functions from $\mathbb{F}_2^n$ to itself are the permutations of $\mathbb{F}_2^n$. A function F is balanced if and only if all non-zero component functions of F are balanced. That is, if and only if the Boolean function $v \cdot F$ is balanced for every non-zero $v \in \mathbb{F}_2^n$ (see [Car20]). Let A_1, $A_2 \colon \mathbb{F}_2^n \to \mathbb{F}_2^n$ be affine permutations, then the functions F and $A_1 \circ F \circ A_2$ are called *affine equivalent*. We have that the two affine equivalent functions have the same algebraic degree. The algebraic degree of a function is not invariant under the inverse transformation.

Let $N = ns_1$ and $M = ms_2$, then we can represent a function $\mathcal{F}$ from $\mathbb{F}_2^N$ to $\mathbb{F}_2^M$ as a function from $(\mathbb{F}_2^n)^{s_1}$ to $(\mathbb{F}_2^m)^{s_2}$ in the following way

$$\mathcal{F}(x_1, \ldots, x_{s_1}) = (\mathcal{F}_1(x_1, \ldots, x_{s_1}), \ldots, \mathcal{F}_{s_2}(x_1, \ldots, x_{s_1})),$$

where the functions $\mathcal{F}_1, \ldots, \mathcal{F}_{s_2} \colon (\mathbb{F}_2^n)^{s_1} \to \mathbb{F}_2^m$ are called the *coordinate functions* of the function $\mathcal{F}$.

A function $\mathcal{F} \colon \mathbb{F}_2^N \to \mathbb{F}_2^n$ where $N = ns$ can be represented as a function from $\mathbb{F}_{2^n}^s$ to $\mathbb{F}_{2^n}$ as a multivariate polynomial over $\mathbb{F}_{2^n}$ of the following form:

$$\mathcal{F}(x_1, \ldots, x_s) = \sum_{u \in \{0, \ldots, 2^n - 1\}^s} c(u) \left(\prod_{i=1}^{s} x_i^{u_i} \right), \quad c(u) \in \mathbb{F}_{2^n}.$$

2.2 Threshold Implementations

A *Boolean s-sharing* of $x \in \mathbb{F}_2^n$ is a tuple $\underline{x} = (x_1, \ldots, x_s) \in (\mathbb{F}_2^n)^s$ over $\mathbb{F}_2^n$ such that

$$x = \sum_{i=1}^{s} x_i.$$

The value x can be interpreted as the secret, and $x_1, \ldots, x_s$ as the shares.

For every $x \in \mathbb{F}_2^n$, we define the set of *s-sharings* of x as

$$\mathrm{Sh}_s(x) \coloneqq \left\{ \underline{x} = (x_1, \ldots, x_s) \in (\mathbb{F}_2^n)^s \mid \sum_{i=1}^{s} x_i = x \right\}.$$

It follows directly from the definition that $|\,\mathrm{Sh}_s(x)| = |\,(\mathbb{F}_2^n)^{s-1}\,| = 2^{n(s-1)}$ for every $x \in \mathbb{F}_2^n$.

From now on, we will use simply s-sharing instead of Boolean s-sharing for the sake of brevity.

Let s_x, s_y be positive integers. We refer to a function $\mathcal{F} \colon (\mathbb{F}_2^n)^{s_x} \to (\mathbb{F}_2^m)^{s_y}$ as an s_x *to* s_y *sharing*.

We say that $\mathcal{F}$ is *correct*, or equivalently that it has the *correctness* property, with respect to $F : \mathbb{F}_2^n \to \mathbb{F}_2^m$, if it maps any s_x-sharing of x over $\mathbb{F}_2^n$, to an s_y-sharing y over $\mathbb{F}_2^m$, i.e. for all $x \in \mathbb{F}_2^n$, and $\underline{x} \in \mathrm{Sh}_{s_x}(x)$ we have

$$\mathcal{F}(\underline{x}) \in \mathrm{Sh}_{s_y}(F(x)). \tag{1}$$

Note that the notion of *Correctness* given in [Bil15], i.e. that for all $x \in \mathbb{F}_2^n$, we have that $\underline{x} \in \mathrm{Sh}_{s_x}(x)$ and $\sum_{j=1}^{s} \mathcal{F}_j(\underline{x}) = F(\sum_{i=1}^{s} x_i)$, immediately follows from Equality (1) and the definition of s-sharing.

For $i \in \{1, \ldots, s_x\}$, $j \in \{1, \ldots, s_y\}$, and $a \in \mathbb{F}_2^n$ we denote the derivative of $\mathcal{F}$ over coordinate i in direction a as

$$D_a^{(i)} \mathcal{F}_j(\underline{x}) = \mathcal{F}_j(x_1, \ldots, x_{i-1}, x_i + a, x_{i+1}, \ldots, x_{s_x}) + \mathcal{F}_j(\underline{x}).$$

We note that if $D_a^{(i)} \mathcal{F}_j = 0$ for all $a \in \mathbb{F}_2^n$, then the value of $\mathcal{F}_j$ is unaffected by the variable x_i and we say that $\mathcal{F}_j$ does not depend on x_i, or equivalently that $\mathcal{F}_j$ is independent of x_i. Otherwise, we say that $\mathcal{F}_j$ depends on x_i. We can now define the set of all i such that $\mathcal{F}_j$ does not depend on x_i as

$$S_j = \{ i \in \{1, \ldots, s_x\} \mid \forall a \in \mathbb{F}_2^n,\, D_a^{(i)} \mathcal{F}_j = 0 \}.$$

Finally, we can say that $\mathcal{F}$ is d^{th} *order non-complete*, or equivalently that it has the d^{th} *order non-completeness* property, if for all $I \subseteq \{1, \ldots, s_y\}$ with $|I| \leq d$, we have that

$$\bigcap_{j \in I} S_j \neq \emptyset,$$

i.e. there exists at least one i such that none of the considered $\mathcal{F}_j$ depends on x_i.

Let $\mathcal{F}$ be correct with respect to $F\colon \mathbb{F}_2^n \to \mathbb{F}_2^m$. We say that $\mathcal{F}$ is *uniform*, or equivalently that it has the *uniformity* property, if for all $x \in \mathbb{F}_2^n$ and $\underline{y} \in \mathrm{Sh}_{s_y}(F(x))$, the following holds:

$$\left|\{\underline{x} \in \mathrm{Sh}_{s_x}(x) \mid \mathcal{F}(\underline{x}) = \underline{y}\}\right| = \frac{2^{n(s_x-1)}}{2^{m(s_y-1)}}.$$

We say that $\mathcal{F}$ is a d^{th} order *threshold implementation* of $F\colon \mathbb{F}_2^n \to \mathbb{F}_2^m$ if $\mathcal{F}$ is correct with respect to F, d^{th} order non-complete, and uniform. We call $\mathcal{F}$ symmetric if $s_x = s_y$. We note that according to [Bil15], if $\mathcal{F}$ is symmetric and F is a permutation, we have that $\mathcal{F}$ being uniform is equivalent to $\mathcal{F}$ being a permutation. This statement is not proven in literature to the best of our knowledge. Thus, we give a brief proof of it in Appendix A.

We know that for a d^{th} order threshold implementation of a vectorial Boolean function F of algebraic degree t to exist, s_x must satisfy $s_x \geq td + 1$ [NRR06, BGN+14, Pet19].

In this paper, we focus on symmetric 1^{st} order threshold implementations of permutations. In this setting, we have that the lower bound on the number of shares becomes $s \geq t + 1$. From the definition, we have that $\mathcal{F}$ is non-complete if and only if, for each coordinate function $\mathcal{F}_j$, there exists a component x_i such that $\mathcal{F}_j$ does not depend on x_i. For the sake of brevity, we will only write threshold implementation instead of symmetric 1^{st} order threshold implementation from now on.

3 A Universal Construction for the Threshold Implementation with $t + 2$ Shares

In this section, we construct a threshold implementation $\mathcal{F}$ with $t + 2$ shares for every permutation F over $\mathbb{F}_2^n$ of algebraic degree at most t, for $t \geq 2$. This construction does not depend on the dimension n of the vector space $\mathbb{F}_2^n$ that is permuted by F.

Let $\mathcal{F}\colon (\mathbb{F}_2^n)^{t+2} \to (\mathbb{F}_2^n)^{t+2}$ be defined for every $\underline{x} = (x_1, \ldots, x_{t+2}) \in (\mathbb{F}_2^n)^{t+2}$ as

$$\mathcal{F}(\underline{x}) = \begin{pmatrix} \mathcal{F}_1(\underline{x}) & = & x_1 \\ \mathcal{F}_2(\underline{x}) & = & \sum_{i=3}^{t+2} x_i + F\left(\sum_{i=2}^{t+2} x_i\right) \\ \mathcal{F}_j(\underline{x}) & = & x_j + \sum_{I \in \mathcal{I}_{j-2}} F\left(\sum_{i \in I} x_i + \sum_{i=j}^{t+2} x_i\right) \\ & & j = 3, \ldots, t+1 \\ \mathcal{F}_{t+2}(\underline{x}) & = & x_{t+2} + x_1 + \sum_{I \in \mathcal{I}_t} F\left(\sum_{i \in I} x_i\right) \end{pmatrix}^{\mathrm{T}}, \tag{2}$$

where for any $k \geq 1$ we denote by $\mathcal{I}_k$ as the set of all subsets of $\{1, \ldots, k\}$ (including $\emptyset$). We will also denote $\mathcal{I}_k^* = \mathcal{I}_k \setminus \{\{1, \ldots, k\}\}$. Moreover, we use the convention that $\sum_{i \in \emptyset} x_i = 0$.

Our construction is done for $t \geq 2$ and can not be defined for $t = 1$. However, it is already known how to construct threshold implementations of affine permutations.

First, we observe that $\mathcal{F}$ is non-complete. Indeed, $\mathcal{F}_1$ is independent of $x_2, \ldots, x_{t+2}$, and functions $\mathcal{F}_j(\underline{x})$ are independent of x_{j-1} for any $j = 2, \ldots, t+2$. Furthermore, we will prove that $\mathcal{F}$ is correct with respect to F (Theorem 2) and uniform (Theorem 3). With this, we will conclude that $\mathcal{F}$ is a threshold implementation of F. As a consequence, we can state the following theorem that sums up the main result of this section.

Theorem 1. *Let F be a permutation over $\mathbb{F}_2^n$ with algebraic degree t. Then for every $s \geq t + 2$, function F admits a threshold implementation with s shares.*

Note that, according to this theorem, it is sufficient for F to have algebraic degree at most t, making the construction viable even if F has algebraic degree strictly lower than t.

There are instances where the flexibility with the number of shares might be useful. For instance, when composing functions with different algebraic degrees, one might choose the construction necessary for the highest algebraic degree and apply it to all functions to achieve a seamless composition.

3.1 Proving the Correctness Property

We prove the following theorem.

Theorem 2. *Let $F\colon \mathbb{F}_2^n \to \mathbb{F}_2^m$ be of algebraic degree at most $t \geq 2$. Then for every $x_1, x_2, \ldots, x_{t+2} \in \mathbb{F}_2^n$ we have that $F\left(\sum_{i=1}^{t+2} x_i\right)$ is equal to*

$$F\left(\sum_{i=2}^{t+2} x_i\right) + \sum_{j=3}^{t+1} \sum_{I \in \mathcal{I}_{j-2}} F\left(\sum_{i \in I} x_i + \sum_{i=j}^{t+2} x_i\right) + \sum_{I \in \mathcal{I}_t} F\left(\sum_{i \in I} x_i\right). \tag{3}$$

It is easy to note that, for every $\underline{x} = (x_1, \ldots, x_{t+2}) \in (\mathbb{F}_2^n)^{t+2}$, the sum $\sum_{i=1}^{t+2} \mathcal{F}_i(\underline{x})$ is equal to Expression (3). Hence, using Theorem 2 we can prove that the function $\mathcal{F}$ defined in (2) satisfies the correctness property with respect to F.

We can see that the correctness property does not depend on the condition that F is a permutation.

To prove Theorem 2 we need the following lemmas.

Lemma 1 ([CPRR15, Corollary 1])**.** *Let $F\colon \mathbb{F}_2^n \to \mathbb{F}_2^m$ be of algebraic degree at most $t \geq 1$ and let $s > t$. Then for every $x_1, x_2, \ldots, x_s \in \mathbb{F}_2^n$ we have that*

$$F\left(\sum_{i=1}^{s} x_i\right) = \sum_{j=0}^{t} \mu_{s,t}(j) \sum_{I \in \mathcal{I}_s,\, |I|=j} F\left(\sum_{i \in I} x_i\right)$$

where $\mu_{s,t}(j) = \binom{s-j-1}{t-j} \mod 2$ for every $j = 0, \ldots, t$.

Lemma 2. *Let $F\colon \mathbb{F}_2^n \to \mathbb{F}_2^m$. Let $k \geq 1$, then for every $x_1, x_2, \ldots, x_{k+1}, z \in \mathbb{F}_2^n$ we have that*

$$\sum_{I \in \mathcal{I}_{k+1}^*} F\left(\sum_{i \in I} x_i + z\right) = \sum_{I \in \mathcal{I}_k} F\left(\sum_{i \in I} x_i + z\right) + \sum_{I \in \mathcal{I}_k^*} F\left(\sum_{i \in I} x_i + x_{k+1} + z\right).$$

Proof. It follows from the following disjoint union $\mathcal{I}_{k+1}^* = \mathcal{I}_k \cup \{I \cup \{k+1\} \colon I \in \mathcal{I}_k^*\}$. □

Lemma 3. *Let $F\colon \mathbb{F}_2^n \to \mathbb{F}_2^m$ and $t \geq 2$. Then for every $x_1, x_2, \ldots, x_{t+2} \in \mathbb{F}_2^n$ and for every $3 \leq k \leq t+1$ we have that*

$$\begin{aligned}
&\sum_{I \in \mathcal{I}_t^*} F\left(\sum_{i \in I} x_i + \sum_{j=t+1}^{t+2} x_j\right) = \\
&= \sum_{j=k}^{t+1} \sum_{I \in \mathcal{I}_{j-2}} F\left(\sum_{i \in I} x_i + \sum_{i=j}^{t+2} x_i\right) + \sum_{I \in \mathcal{I}_{k-2}^*} F\left(\sum_{i \in I} x_i + \sum_{i=k-1}^{t+2} x_i\right).
\end{aligned}$$

Proof. We use induction from $k = t+1$ to $k = 3$. Set $z_k = \sum_{j=k}^{t+2} x_j$. By using Lemma 2, we can prove the base case $k = t+1$:

$$\begin{aligned}
\sum_{I \in \mathcal{I}_t^*} F\left(\sum_{i \in I} x_i + z_{t+1}\right) &= \sum_{I \in \mathcal{I}_{t-1}} F\left(\sum_{i \in I} x_i + z_{t+1}\right) + \sum_{I \in \mathcal{I}_{t-1}^*} F\left(\sum_{i \in I} x_i + x_t + z_{t+1}\right) = \\
&= \sum_{I \in \mathcal{I}_{t-1}} F\left(\sum_{i \in I} x_i + z_{t+1}\right) + \sum_{I \in \mathcal{I}_{t-1}^*} F\left(\sum_{i \in I} x_i + z_t\right).
\end{aligned}$$

Now we prove it for $k-1$ assuming it is true for k. For k we have

$$\sum_{I\in\mathcal{I}_t^*} F\left(\sum_{i\in I} x_i + z_{t+1}\right) = \sum_{j=k}^{t+1} \sum_{I\in\mathcal{I}_{j-2}} F\left(\sum_{i\in I} x_i + z_j\right) + \sum_{I\in\mathcal{I}_{k-2}^*} F\left(\sum_{i\in I} x_i + z_{k-1}\right). \quad (4)$$

If $t=2$ we have that $t+1=3$, so there is nothing more to prove. Assuming $t\geq 3$, we consider the second term on the right side of Equality (4) and using again Lemma 2 we get

$$\begin{aligned}
&\sum_{I\in\mathcal{I}_{k-2}^*} F\left(\sum_{i\in I} x_i + z_{k-1}\right) = \\
&= \sum_{I\in\mathcal{I}_{k-3}} F\left(\sum_{i\in I} x_i + z_{k-1}\right) + \sum_{I\in\mathcal{I}_{k-3}^*} F\left(\sum_{i\in I} x_i + x_{k-2} + z_{k-1}\right) = \\
&= \sum_{I\in\mathcal{I}_{k-3}} F\left(\sum_{i\in I} x_i + z_{k-1}\right) + \sum_{I\in\mathcal{I}_{k-3}^*} F\left(\sum_{i\in I} x_i + z_{k-2}\right).
\end{aligned}$$

Hence, inserting the obtained expression into Equality (4) we get

$$\sum_{I\in\mathcal{I}_t^*} F\left(\sum_{i\in I} x_i + z_{t+1}\right) = \sum_{j=k-1}^{t+1} \sum_{I\in\mathcal{I}_{j-2}} F\left(\sum_{i\in I} x_i + z_j\right) + \sum_{I\in\mathcal{I}_{k-3}^*} F\left(\sum_{i\in I} x_i + z_{k-2}\right).$$

□

Now we can prove Theorem 2.

of Theorem 2. Set $z_{t+1} = \sum_{j=t+1}^{t+2} x_j$, $z_i = x_i$ for $i = 1,\ldots,t$. By using Lemma 1 for the case $s = t+1$ on $z_1,\ldots,z_{t+1}$, we obtain

$$F\left(\sum_{i=1}^{t+2} x_i\right) = F\left(\sum_{i=1}^{t+1} z_i\right) = \sum_{I\in\mathcal{I}_{t+1}^*} F\left(\sum_{i\in I} z_i\right).$$

In fact, $\mu_{t+1,t}(j) = \binom{t-j}{t-j} \mod 2 = 1$ for all $j = 0,\ldots,t$.
By using Lemma 2 we have that

$$\begin{aligned}
\sum_{I\in\mathcal{I}_{t+1}^*} F\left(\sum_{i\in I} z_i\right) &= \sum_{I\in\mathcal{I}_t} F\left(\sum_{i\in I} z_i\right) + \sum_{I\in\mathcal{I}_t^*} F\left(\sum_{i\in I} z_i + z_{t+1}\right) = \\
&= \sum_{I\in\mathcal{I}_t} F\left(\sum_{i\in I} x_i\right) + \sum_{I\in\mathcal{I}_t^*} F\left(\sum_{i\in I} x_i + \sum_{j=t+1}^{t+2} x_j\right).
\end{aligned}$$

We use the case $k=3$ of Lemma 3 to conclude that

$$\begin{aligned}
&\sum_{I\in\mathcal{I}_t^*} F\left(\sum_{i\in I} x_i + \sum_{j=t+1}^{t+2} x_j\right) = \\
&= \sum_{j=3}^{t+1} \sum_{I\in\mathcal{I}_{j-2}} F\left(\sum_{i\in I} x_i + \sum_{i=j}^{t+2} x_i\right) + \sum_{I\in\mathcal{I}_1^*} F\left(\sum_{i\in I} x_i + \sum_{i=2}^{t+2} x_i\right) = \\
&= \sum_{j=3}^{t+1} \sum_{I\in\mathcal{I}_{j-2}} F\left(\sum_{i\in I} x_i + \sum_{i=j}^{t+2} x_i\right) + F\left(\sum_{i=2}^{t+2} x_i\right).
\end{aligned}$$

□

3.2 Proving the Uniformity Property

We are going to prove the following theorem.

Theorem 3. *Let F be a permutation over $\mathbb{F}_2^n$ of algebraic degree at most $t \geq 2$. Then the function $\mathcal{F}$ as defined in (2) is a permutation over $(\mathbb{F}_2^n)^{t+2}$.*

Proof. Let $\mathcal{F}\colon (\mathbb{F}_2^n)^{t+2} \to (\mathbb{F}_2^n)^{t+2}$ be as defined in (2). We have that $\mathcal{F}$ is correct with respect to F because of Theorem 2. Let $\underline{x} = (x_1, \ldots, x_{t+2}) \in (\mathbb{F}_2^n)^{t+2}$. We introduce variables $y_i = \mathcal{F}_i(\underline{x})$ over $\mathbb{F}_2^n$ for $i = 1, \ldots, t+2$ to define a system of equations:

$$\begin{cases} y_1 = x_1 \\ y_2 = \sum_{i=3}^{t+2} x_i + F\left(\sum_{i=2}^{t+2} x_i\right) \\ y_j = x_j + \sum_{I \in \mathcal{I}_{j-2}} F\left(\sum_{i \in I} x_i + \sum_{i=j}^{t+2} x_i\right) \quad j = 3, \ldots, t+1 \\ y_{t+2} = x_{t+2} + x_1 + \sum_{I \in \mathcal{I}_t} F\left(\sum_{i \in I} x_i\right). \end{cases} \tag{5}$$

Let $\underline{y} = (y_1, \ldots, y_{t+2})$. Then $\mathcal{F}$ is a permutation if and only if for every $i = 1, \ldots, t+2$ there exists a function $\mathcal{G}_i\colon (\mathbb{F}_2^n)^{t+2} \to \mathbb{F}_2^n$ such that $x_i = \mathcal{G}_i(\underline{y})$. We are going to use the fact that since $F\left(\sum_{i=1}^{t+2} x_i\right) = \sum_{i=1}^{t+2} y_i$, then

$$\sum_{i=1}^{t+2} x_i = F^{-1}\left(\sum_{i=1}^{t+2} y_i\right). \tag{6}$$

We have that $x_1 = y_1 = \mathcal{G}_1(\underline{y})$ using the first equation of System (5). By using the second equation of System (5), we have that

$$\sum_{i=3}^{t+2} x_i = y_2 + F\left(\sum_{i=2}^{t+2} x_i\right) = y_2 + F\left(\mathcal{G}_1(\underline{y}) + F^{-1}\left(\sum_{i=1}^{t+2} y_i\right)\right) = \mathcal{H}_3(\underline{y})$$

for some function $\mathcal{H}_3$. By using Equality (6), we have that

$$x_2 = x_1 + \sum_{i=3}^{t+2} x_i + F^{-1}\left(\sum_{i=1}^{t+2} y_i\right) = \mathcal{G}_1(\underline{y}) + \mathcal{H}_3(\underline{y}) + F^{-1}\left(\sum_{i=1}^{t+2} y_i\right) = \mathcal{G}_2(\underline{y}).$$

We continue by using the induction method. We claim that at step $2 \leq j \leq t+1$ we have that $x_i = \mathcal{G}_i(\underline{y})$ for all $1 \leq i \leq j$ and $\sum_{i=j+1}^{t+2} x_i = \mathcal{H}_{j+1}(\underline{y})$ for some function $\mathcal{H}_{j+1}$. Observe that the case $j = 2$ is true since we have that $x_1 = \mathcal{G}_1(\underline{y})$, $x_2 = \mathcal{G}_2(\underline{y})$, and $\sum_{i=3}^{t+2} x_i = \mathcal{H}_3(\underline{y})$. Now we prove it for $3 \leq j \leq t+1$, assuming it is true for $j-1$. By using the j-th equation of System (5), we have that

$$\begin{aligned} x_j &= y_j + \sum_{I \in \mathcal{I}_{j-2}} F\left(\sum_{i \in I} x_i + \sum_{i=j}^{t+2} x_i\right) = \\ &= y_j + \sum_{I \in \mathcal{I}_{j-2}} F\left(\sum_{i \in I} \mathcal{G}_i(\underline{y}) + \mathcal{H}_j(\underline{y})\right) = \mathcal{G}_j(\underline{y}). \end{aligned}$$

By using Equality (6), we have that

$$\sum_{i=j+1}^{t+2} x_i = \sum_{i=1}^{j} x_i + F^{-1}\left(\sum_{i=1}^{t+2} y_i\right) = \sum_{i=1}^{j} \mathcal{G}_i(\underline{y}) + F^{-1}\left(\sum_{i=1}^{t+2} y_i\right) = \mathcal{H}_{j+1}(\underline{y}).$$

Hence, we get $x_i = \mathcal{G}_i(\underline{y})$ for all $i \leq t+1$ and $x_{t+2} = \sum_{i=t+2}^{t+2} x_i = \mathcal{H}_{t+2}(\underline{y}) = \mathcal{G}_{t+2}(\underline{y})$. □

We observe that we never used the last equation of System (5) in the proof of Theorem 3. The reason being that the last coordinate function can be deduced from the first $t+1$ coordinates using the correctness property and the fact that $\sum_{i=1}^{t+2} \mathcal{F}_i(\underline{x}) = F\left(\sum_{i=1}^{t+2} x_i\right)$.

Due to Theorem 3 we have that the function $\mathcal{F}$ is a permutation, and it is correct with respect to F due to Theorem 2. Hence, $\mathcal{F}$ is also uniform.

3.3 Adding Correction Terms without Losing any Property

The use of correction terms has been proposed first in [NRR06, NRS08] as a method to make a direct sharing [BNN+12] uniform. Correction terms are coordinate terms that are added in pairs to more than one share, such that the new function obtained still satisfies the non-completeness property. Our aim is to take the threshold implementation $\mathcal{F}$ as constructed in (2) and add some correction terms so that the new function obtained is still a threshold implementation. This procedure is important to study because it gives the possibility to construct new threshold implementations that may have better cryptographic properties than the one described in (2) e.g., higher algebraic degree.

In the following corollary we present functions $\mathcal{C}$, such that $\mathcal{F}+\mathcal{C}$ is still a threshold implementation with $t+2$ shares.

Proposition 1. *Let F be a permutation over $\mathbb{F}_2^n$ of algebraic degree at most $t \geq 2$ and $\mathcal{F}$ be the function defined in (2). Let $\mathcal{C}\colon (\mathbb{F}_2^n)^{t+2} \to (\mathbb{F}_2^n)^{t+2}$ be defined as*

$$\mathcal{C}(\underline{x}) = \left(\begin{array}{lcl} \mathcal{C}_1(\underline{x}) & = & x_1 + P_1(x_1) \\ \mathcal{C}_2(\underline{x}) & = & \sum_{i=3}^{t+2} x_i + P_2\left(\sum_{i=3}^{t+2} x_i\right) + C_2\left(\sum_{i=2}^{t+2} x_i\right) \\ \mathcal{C}_j(\underline{x}) & = & x_j + P_j(x_j) + C_j(x_1, \ldots, x_{j-2}, \sum_{i=j}^{t+2} x_i) \\ & & j = 3, \ldots, t+1 \\ \mathcal{C}_{t+2}(\underline{x}) & = & C_{t+2}(x_1, \ldots, x_t, x_{t+2}) \end{array} \right)^T,$$

where function P_j is a permutation over $\mathbb{F}_2^n$ for all $j = 1, \ldots, t+1$; C_j is a function from $(\mathbb{F}_2^n)^{j-1}$ to $\mathbb{F}_2^n$ for $j = 2, \ldots, t+2$, such that $\sum_{i=1}^{t+2} \mathcal{C}_i(\underline{x}) = 0$. Then $\mathcal{F}+\mathcal{C}$ is a permutation over $(\mathbb{F}_2^n)^{t+2}$.

Proof. The proof is very similar to the one of Theorem 3, so we will not give too many details. First, we have that

$$\sum_{i=1}^{t+2} \mathcal{F}_i(\underline{x}) = \sum_{i=1}^{t+2} \mathcal{F}_i(\underline{x}) + \sum_{i=1}^{t+2} \mathcal{C}_i(\underline{x}).$$

We introduce variables $y_i = \mathcal{F}_i(\underline{x}) + \mathcal{C}_i(\underline{x})$ for $i = 1, \ldots, t+2$.

Since $\mathcal{F}_1(\underline{x}) + \mathcal{C}_1(\underline{x}) = P_1(x_1)$ and P_1 is a permutation, we can write x_1 and $\sum_{i=2}^{t+2} x_i$ in terms of the y's.

Since $\mathcal{F}_2(\underline{x}) + \mathcal{C}_2(\underline{x}) = P_2\left(\sum_{i=3}^{t+2} x_i\right) + C_2\left(\sum_{i=2}^{t+2} x_i\right) + F\left(\sum_{i=2}^{t+2} x_i\right)$ and P_2 is a permutation, we can write $\sum_{i=3}^{t+2} x_i$ in terms of the y's and consequently x_2 in terms of the y's.

We continue using a similar induction argument as in the proof of Theorem 3. We will prove that, for $2 \leq j \leq t+1$, we can write $x_1, \ldots, x_j$, and $\sum_{i=j+1}^{t+2} x_i$ in terms of the y's. For $j = 2$, it is true. Now, assuming it is true for $j-1$, we prove it for $j \geq 3$. Since $\mathcal{F}_j(\underline{x}) + \mathcal{C}_j(\underline{x}) = P_j(x_j) + C_j(x_1, \ldots, x_{j-2}, \sum_{i=j}^{t+2} x_i)$ and P_j is a permutation, we can write x_j in terms of the y's and consequently $\sum_{i=j+1}^{t+2} x_i$ in terms of the y's. □

4 Two Uniform Implementations of the AES S-box

We use the construction (2) from Section 3 to find the first threshold implementations of the AES S-box without the use of additional randomness or the changing of the guards construction by Daemen [Dae17]. We implement the sharings and provide an area cost in hardware.

The first implementation is a direct application of the construction (2) from Section 3 on the AES S-box. Since this S-box has an algebraic degree of seven, we use nine shares. The result is a sharing with a large area cost, but which requires only one cycle to compute.

The second implementation uses the decomposition given in the work by Wegener and Moradi [WM18]. There, the inversion over $\mathbb{F}_{2^8}$ is represented as a composition of two cubic functions, namely x^{26} and x^{49}. Each of the two cubic functions is then shared using the construction (2) of Section 3. The result is a sharing with a much smaller area overhead compared to the first implementation, but it requires two cycles to compute.

We have estimated the hardware cost of these two uniform masked S-boxes. The area is measured in gate equivalences (GE), i.e., the S-box area normalised to the area of a 2-input `NAND` gate in a given standard cell library. In this work, we use the `NANGATE` 45nm Open Cell Library [NAN] where the synthesis results are obtained with the Synopsis Design Compiler v2021.06 using the `KEEP_HIERACHY` option to prevent optimisation across modules in the synthesis step. The results of the synthesis and its comparison with sharings of the AES S-box in the literature, using the same standard cell library, are shown in Table 4.1. The latency of the masked S-boxes is given in the number of cycles, denoted *cc*, and we provide the total number of random bits which are needed in the computation of the masked AES S-box.

Table 4.1: Hardware cost of the masked AES S-box in the `NANGATE` 45nm library.

Design	Area [kGE]	Latency [cc]	Randomness [$bits$]
This work [without decomposition]	128.27	1	0
This work [with decomposition]	21.86	2	0
Wegener-Moradi [WM18][1]	4.20	16	0
Sugawara [Sug19]	3.50	4	0
Gross et al. [GIB18]	60.76	1	2 048
Gross et al. [GIB18]	6.74	2	416

1. Wegener and Moradi wrote that without serialisation their design costs will be "more than 20 kGE" which is comparable to our design's cost.

The first implementation on the AES S-box provides a sharing costing 128.27 kGE. However, it is the first sharing of the AES S-box in one cycle, which requires no fresh randomness. The implementation of the decomposed shared AES S-box is a direct improvement over the S-box design by Wegener and Moradi who used the changing of the guards method by Daemen to ensure the uniformity of their five-share non-complete sharing of the cubic functions x^{26} and x^{49}. Instead, this work's sharing method provides both non-completeness and uniformity. Note that the difference in implementation results of this work and Wegener and Moradi's is in the architecture. Whereas their work made a highly serialised implementation, we went with a rolled-out design of each cubic function. The result is a low-latency sharing at the cost of an increased area. The sharing requires only two cycles and is randomness-free.

We note that by using the correction terms from Section 3.3, one could get improved area costs. However, we did not pursue this direction and leave this possible optimisation for future investigation.

5 Conclusions

In this paper, we have presented a universal construction for a threshold implementation with $t+2$ shares, where t is the algebraic degree of the S-box. This is the first construction that applies to all bijective S-boxes of any size. This result is a significant advance with respect to the state of the art on the construction of threshold implementations, which were either computational searches for small sizes using direct sharing [BNN+12], or techniques to restore uniformity of a non-complete sharing. For instance, the use of correction terms [NRR06], fresh randomness, or the changing of the guards [Dae17]. It was also noted that this construction yields a threshold implementation with $t+2$ shares in the case of the 3-bit inversion. Such an implementation was proven to be optimal for this S-box by Bilgin et al. [BNN+15]. This means that this construction is optimal for some S-boxes, even if it does not achieve the theoretical lower bound on the number of shares.

We observed that the construction is rather flexible, allowing to change the form of the constructed threshold implementation using correction terms and providing a description of the terms that can be used for this purpose.

We applied this construction to obtain the first uniform sharing of the AES S-box. We have analysed the cost of the implementation using this construction, both directly to the AES S-box and to a decomposition of the S-box using cubic power permutations. The results include the first design of a randomness-free AES S-box in one cycle and a direct improvement on the S-box design by Wegener and Moradi [WM18]. We noted that it might be possible to improve the presented implementation using correction terms. However, we leave this investigation as future work.

The result is also of importance for the research on permutation polynomials, as it provides a method for the construction of new infinite families of permutations.

This result is a very important advance in the understanding of the general theory of threshold implementation. Regarding other aspects of this topic, very little is known. In some cases, it is very hard to find even computational results. For instance, we know very few constructions of 2^{nd} order threshold implementations. Achieving the uniformity property for non-permutations is very challenging, since there is no characterisation that is computationally faster to verify than the definition. Moreover, we do not know the reasons why some permutations do not admit a threshold implementation with $t+1$ shares. In fact, we have very little data to study the non-existence because it is only feasible to run the exhaustive search of correction terms [BNN+12] for 3-bit S-boxes. However, we could still observe that the only examples of permutations that admit a threshold implementation with $t+1$ shares are ones with bad cryptographic properties.

We believe that the fact that the 3-bit inversion (which in this specific case is equivalent to the Gold function x^3) does not admit such a threshold implementation is not a coincidence. There have been many examples in the vectorial Boolean function theory where properties of the Gold functions reflected general results [Car20]. The fact that the function x^3 over $\mathbb{F}_{2^3}$, being the simplest non-linear power permutation and the simplest case of APN functions does not admit $t+1$ shares, leads us to the conjectures presented below.

Conjecture 1. *No power permutation of algebraic degree $t \geq 2$ admits a threshold implementation with $t+1$ shares.*

Conjecture 2. *No APN permutation of algebraic degree t admits a threshold implementation with $t+1$ shares.*

These conjectures are supported by all computational data available nowadays [BNN+12, BBS17, MB19].

Acknowledgements We thank Ventzislav Nikov for useful discussions. The research of this paper is supported by the Norwegian Research Council.

References

[BBS17] Dusan Bozilov, Begül Bilgin, and Haci Ali Sahin. A note on 5-bit quadratic permutations' classification. *IACR Trans. Symmetric Cryptol.*, 2017(1):398–404, 2017.

[BGN+14] Begül Bilgin, Benedikt Gierlichs, Svetla Nikova, Ventzislav Nikov, and Vincent Rijmen. Higher-order threshold implementations. In Palash Sarkar and Tetsu Iwata, editors, *Advances in Cryptology - ASIACRYPT 2014 - 20th International Conference on the Theory and Application of Cryptology and Information Security, Kaoshiung, Taiwan, R.O.C., December 7-11, 2014, Proceedings, Part II*, volume 8874 of *Lecture Notes in Computer Science*, pages 326–343. Springer, 2014.

[Bil15] Begül Bilgin. *Threshold implementations : as countermeasure against higher-order differential power analysis.* PhD thesis, University of Twente, Enschede, Netherlands, 2015.

[BNN+12] Begül Bilgin, Svetla Nikova, Ventzislav Nikov, Vincent Rijmen, and Georg Stütz. Threshold implementations of all 3 ×3 and 4 ×4 s-boxes. In Emmanuel Prouff and Patrick Schaumont, editors, *Cryptographic Hardware and Embedded Systems - CHES 2012 - 14th International Workshop, Leuven, Belgium, September 9-12, 2012. Proceedings*, volume 7428 of *Lecture Notes in Computer Science*, pages 76–91. Springer, 2012.

[BNN+15] Begül Bilgin, Svetla Nikova, Ventzislav Nikov, Vincent Rijmen, Natalia N. Tokareva, and Valeriya Vitkup. Threshold implementations of small s-boxes. *Cryptogr. Commun.*, 7(1):3–33, 2015.

[Car20] Claude Carlet, editor. *Boolean Functions for Cryptography and Coding Theory.* Cambridge University Press, 2020.

[CCZ98] Claude Carlet, Pascale Charpin, and Victor A. Zinoviev. Codes, bent functions and permutations suitable for des-like cryptosystems. *Des. Codes Cryptogr.*, 15(2):125–156, 1998.

[CJRR99] Suresh Chari, Charanjit S. Jutla, Josyula R. Rao, and Pankaj Rohatgi. Towards sound approaches to counteract power-analysis attacks. In Michael J. Wiener, editor, *Advances in Cryptology - CRYPTO '99, 19th Annual International Cryptology Conference, Santa Barbara, California, USA, August 15-19, 1999, Proceedings*, volume 1666 of *Lecture Notes in Computer Science*, pages 398–412. Springer, 1999.

[CPRR15] Claude Carlet, Emmanuel Prouff, Matthieu Rivain, and Thomas Roche. Algebraic decomposition for probing security. In Rosario Gennaro and Matthew Robshaw, editors, *Advances in Cryptology - CRYPTO 2015 - 35th Annual Cryptology Conference, Santa Barbara, CA, USA, August 16-20, 2015, Proceedings, Part I*, volume 9215 of *Lecture Notes in Computer Science*, pages 742–763. Springer, 2015.

[CRB+16] Thomas De Cnudde, Oscar Reparaz, Begül Bilgin, Svetla Nikova, Ventzislav Nikov, and Vincent Rijmen. Masking AES with d+1 shares in hardware. In Benedikt Gierlichs and Axel Y. Poschmann, editors, *Cryptographic Hardware and Embedded Systems - CHES 2016 - 18th International Conference, Santa Barbara, CA, USA, August 17-19, 2016, Proceedings*, volume 9813 of *Lecture Notes in Computer Science*, pages 194–212. Springer, 2016.

[Dae17] Joan Daemen. Changing of the guards: A simple and efficient method for achieving uniformity in threshold sharing. In *Cryptographic Hardware and Embedded Systems - CHES 2017 Proceedings*, pages 137–153, 2017.

[DR01] Joan Daemen and Vincent Rijmen. Advanced Encryption Standard (AES). National Institute of Standards and Technology (NIST), FIPS PUB 197, U.S. Department of Commerce, Nov. 2001.

[GIB18] Hannes Groß, Rinat Iusupov, and Roderick Bloem. Generic low-latency masking in hardware. *IACR Trans. Cryptogr. Hardw. Embed. Syst.*, 2018(2):1–21, 2018.

[GP99] Louis Goubin and Jacques Patarin. DES and differential power analysis (the "duplication" method). In Çetin Kaya Koç and Christof Paar, editors, *Cryptographic Hardware and Embedded Systems, First International Workshop, CHES'99, Worcester, MA, USA, August 12-13, 1999, Proceedings*, volume 1717 of *Lecture Notes in Computer Science*, pages 158–172. Springer, 1999.

[GSM17] Hannes Groß, David Schaffenrath, and Stefan Mangard. Higher-order side-channel protected implementations of KECCAK. In Hana Kubátová, Martin Novotný, and Amund Skavhaug, editors, *Euromicro Conference on Digital System Design, DSD 2017, Vienna, Austria, August 30 - Sept. 1, 2017*, pages 205–212. IEEE Computer Society, 2017.

[KJJ99] Paul C. Kocher, Joshua Jaffe, and Benjamin Jun. Differential power analysis. In *Advances in Cryptology - CRYPTO '99 Proceedings*, pages 388–397, 1999.

[MB19] Lauren De Meyer and Begül Bilgin. Classification of balanced quadratic functions. *IACR Trans. Symmetric Cryptol.*, 2019(2):169–192, 2019.

[MPL+11] Amir Moradi, Axel Poschmann, San Ling, Christof Paar, and Huaxiong Wang. Pushing the limits: A very compact and a threshold implementation of AES. In Kenneth G. Paterson, editor, *Advances in Cryptology - EUROCRYPT 2011 - 30th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Tallinn, Estonia, May 15-19, 2011. Proceedings*, volume 6632 of *Lecture Notes in Computer Science*, pages 69–88. Springer, 2011.

[MPO05] Stefan Mangard, Norbert Pramstaller, and Elisabeth Oswald. Successfully attacking masked AES hardware implementations. In Josyula R. Rao and Berk Sunar, editors, *Cryptographic Hardware and Embedded Systems - CHES 2005, 7th International Workshop, Edinburgh, UK, August 29 - September 1, 2005, Proceedings*, volume 3659 of *Lecture Notes in Computer Science*, pages 157–171. Springer, 2005.

[NAN] NANGATE. The NanGate 45nm Open Cell Library. Version: PDKv1.3 v2010 12.Apache.CCL, https://github.com/The-OpenROAD-Project/OpenROAD-flow-scripts/tree/master/flow/platforms/nangate45.

[NNR19] Svetla Nikova, Ventzislav Nikov, and Vincent Rijmen. Decomposition of permutations in a finite field. *Cryptogr. Commun.*, 11(3):379–384, 2019.

[NRR06] Svetla Nikova, Christian Rechberger, and Vincent Rijmen. Threshold implementations against side-channel attacks and glitches. In Peng Ning, Sihan Qing, and Ninghui Li, editors, *Information and Communications Security, 8th International Conference, ICICS 2006, Raleigh, NC, USA, December 4-7, 2006, Proceedings*, volume 4307 of *Lecture Notes in Computer Science*, pages 529–545. Springer, 2006.

[NRS08] Svetla Nikova, Vincent Rijmen, and Martin Schläffer. Secure hardware implementation of non-linear functions in the presence of glitches. In Pil Joong Lee and Jung Hee Cheon, editors, *Information Security and Cryptology - ICISC 2008, 11th International Conference, Seoul, Korea, December 3-5, 2008, Revised Selected Papers*, volume 5461 of *Lecture Notes in Computer Science*, pages 218–234. Springer, 2008.

[Pet19] George Petrides. On non-completeness in threshold implementations. In Begül Bilgin, Svetla Petkova-Nikova, and Vincent Rijmen, editors, *Proceedings of ACM Workshop on Theory of Implementation Security, TIS@CCS 2019, London, UK, November 11, 2019*, pages 24–28. ACM, 2019.

[Pet22] George Petrides. On decompositions of permutation polynomials into quadratic and cubic power permutations. *Cryptography and Communications*, 2022.

[PMK+11] Axel Poschmann, Amir Moradi, Khoongming Khoo, Chu-Wee Lim, Huaxiong Wang, and San Ling. Side-channel resistant crypto for less than 2, 300 GE. *J. Cryptol.*, 24(2):322–345, 2011.

[RBN+15] Oscar Reparaz, Begül Bilgin, Svetla Nikova, Benedikt Gierlichs, and Ingrid Verbauwhede. Consolidating masking schemes. In Rosario Gennaro and Matthew Robshaw, editors, *Advances in Cryptology - CRYPTO 2015 - 35th Annual Cryptology Conference, Santa Barbara, CA, USA, August 16-20, 2015, Proceedings, Part I*, volume 9215 of *Lecture Notes in Computer Science*, pages 764–783. Springer, 2015.

[Sug19] Takeshi Sugawara. 3-share threshold implementation of AES s-box without fresh randomness. *IACR Trans. Cryptogr. Hardw. Embed. Syst.*, 2019(1):123–145, 2019.

[VNNR19] Kerem Varici, Svetla Nikova, Ventzislav Nikov, and Vincent Rijmen. Constructions of s-boxes with uniform sharing. *Cryptogr. Commun.*, 11(3):385–398, 2019.

[WM18] Felix Wegener and Amir Moradi. A first-order SCA resistant AES without fresh randomness. In Junfeng Fan and Benedikt Gierlichs, editors, *Constructive Side-Channel Analysis and Secure Design - 9th International Workshop, COSADE 2018, Singapore, April 23-24, 2018, Proceedings*, volume 10815 of *Lecture Notes in Computer Science*, pages 245–262. Springer, 2018.

A Characterisation of the uniformity property

Let $F\colon \mathbb{F}_2^n \to \mathbb{F}_2^n$ and let $\mathcal{F}$ be an s to s sharing that is correct with respect to F. In Section 2.2, we claimed that if F is a permutation, then $\mathcal{F}$ is a permutation if and only if $\mathcal{F}$ is uniform.

We are going to prove the following proposition. It is not exactly what we claim, but it is indeed stronger.
$\mathcal{F}$ is a permutation if and only if $\mathcal{F}$ is uniform and F is a permutation.

Proof. Let $\mathcal{F}$ be uniform and F be a permutation. We claim that $\mathcal{F}$ is surjective and since the domain of $\mathcal{F}$ is equal to its codomain, this implies that $\mathcal{F}$ is a permutation. Let $y \in \mathbb{F}_2^n$ and $\underline{y} \in \mathrm{Sh}_s(y)$. Since F is surjective, there exists an $\underline{x} \in (\mathbb{F}_2^n)^s$ such that $\mathcal{F}(\underline{x}) = \underline{y}$. Because of correctness, there exists an $x \in \mathbb{F}_2^n$ such that $\underline{x} \in \mathrm{Sh}_s(x)$ and $F(x) = y$.

Let $\mathcal{F}$ be a permutation. We claim that F is surjective and since the domain of F is equal to its codomain, this implies that F is a permutation. Let $\underline{y} \in (\mathbb{F}_2^n)^s$ and $y \in \mathbb{F}_2^n$ be such that $\underline{y} \in \mathrm{Sh}_s(y)$. Since $\mathcal{F}$ is surjective, there exists an $\underline{x} \in (\mathbb{F}_2^n)^s$ such that $\mathcal{F}(\underline{x}) = \underline{y}$. Since $\mathcal{F}$ is correct, then there exists $x \in \mathbb{F}_2^n$ such that $\underline{x} \in \mathrm{Sh}_s(x)$ and $F(x) = y$. So we conclude that F is surjective. We claim that $\mathcal{F}$ is uniform. Let $x \in \mathbb{F}_2^n$ and $\underline{y} \in \mathrm{Sh}_s(F(x))$. Since F is a permutation, exists a unique $\underline{x} \in (\mathbb{F}_2^n)^s$ such that $\mathcal{F}(\underline{x}) = \underline{y}$. Because of correctness, there exists an $x' \in \mathbb{F}_2^n$ such that $\underline{x} \in \mathrm{Sh}_s(x')$ and $F(x') = F(x)$. Since F is a permutation, we have that $x' = x$. This implies that $\mathcal{F}$ is uniform. □