IACR Transactions on Cryptographic Hardware and Embedded Systems
ISSN 2569-2925, Vol. 2023, No. 2, pp. 482–518. DOI:10.46586/tches.v2023.i2.482-518

Prime-Field Masking in Hardware and its Soundness against Low-Noise SCA Attacks

Gaëtan Cassiers[1,2,3], Loïc Masure[3], Charles Momin[3],
Thorben Moos[3], François-Xavier Standaert[3]

[1] Graz University of Technology, Graz, Austria
[2] Lamarr Security Research, Graz, Austria
[3] Crypto Group, ICTEAM Institute, UCLouvain, Louvain-la-Neuve, Belgium.
firstname.lastname@uclouvain.be

Abstract. A recent study suggests that arithmetic masking in prime fields leads to stronger security guarantees against passive physical adversaries than Boolean masking. Indeed, it is a common observation that the desired security amplification of Boolean masking collapses when the noise level in the measurements is too low. Arithmetic encodings in prime fields can help to maintain an exponential increase of the attack complexity in the number of shares even in such a challenging context. In this work, we contribute to this emerging topic in two main directions. First, we propose novel masked hardware gadgets for secure squaring in prime fields (since squaring is non-linear in non-binary fields) which prove to be significantly more resource-friendly than corresponding masked multiplications. We then formally show their local and compositional security for arbitrary orders. Second, we attempt to experimentally evaluate the performance vs. security tradeoff of prime-field masking. In order to enable a first comparative case study in this regard, we exemplarily consider masked implementations of the AES as well as the recently proposed AES-prime. AES-prime is a block cipher partially resembling the standard AES, but based on arithmetic operations modulo a small Mersenne prime. We present cost and performance figures for masked AES and AES-prime implementations, and experimentally evaluate their susceptibility to low-noise side-channel attacks. We consider both the dynamic and the static power consumption for our low-noise analyses and emulate strong adversaries. Static power attacks are indeed known as a threat for side-channel countermeasures that require a certain noise level to be effective because of the adversary's ability to reduce the noise through intra-trace averaging. Our results show consistently that for the noise levels in our practical experiments, the masked prime-field implementations provide much higher security for the same number of shares. This compensates for the overheads prime computations lead to and remains true even if / despite leaking each share with a similar Signal-to-Noise Ratio (SNR) as their binary equivalents. We hope our results open the way towards new cipher designs tailored to best exploit the advantages of prime-field masking.

Keywords: Side-channel attacks · masking · prime ciphers · low-noise leakages

1 Introduction

For decades now, symmetric cryptography has been mostly focused on cipher designs taking advantage of computations in binary fields. The rationale behind this choice is that binary operations are in general very efficiently implemented in hardware and in software. Standards like the AES Rijndael are a good illustration of this trend and ciphers designed for lightweight contexts seem to benefit even more from this choice: see, for example, the

Received: 2022-10-15 Accepted: 2022-12-15 Published: 2023-03-06

candidates to the ongoing NIST lightweight cryptography competition.[1]

So far, this intuition was maintained even in the case where Side-Channel Analysis (SCA) is a legitimate threat. Indeed, the most efficient masking schemes currently known are nicely compatible with binary operations, whether being implemented in (bitslice) software [GR17] or in hardware [GMK17]. Under some (now well understood) noise and independence conditions (see details in Section 2), Boolean masking can amplify the leakage noise exponentially in the number of shares, and is naturally suited to binary ciphers.

In this paper, we aim to challenge this mainstream view and are interested by the provocative question whether masking in prime fields could nevertheless gain interest over Boolean masking when considering the risk of certain types of powerful SCA attacks. While this may sound surprising at first sight (since operations in prime fields generally come with overheads), we argue that there is at least conceptual motivation for this purpose. For example, say an adversary is able to observe the noise-free Hamming weight of Boolean shares (whether processed in serial or in parallel). Then, just observing whether this leakage is even or odd leaks information about the secret, regardless of the number of shares [Sta18]. This is typically reflected by the "plateau" regions of information theoretic evaluations of masked implementations [SVO+10]. Intuitively, such a weakness seems to be related to the "algebraic compatibility" between the Hamming weight function and operations in binary fields – a phenomenon that is also known to create weaknesses in the context of fresh-rekeying (e.g., see [MSGR10, BFG14]). Since prime fields contribute to security in the context of fresh re-keying, by making the cipher operations and their leakage less compatible [DMMS21], it appears natural to investigate whether such a tweak can help in the context of additive masking as well. Interestingly, this idea also benefits from strong theoretical support, as Dziembowski et al. showed that masking in prime fields can be used to amplify arbitrarily low noise levels, while masking in fields of composite order always carries the risk that no noise amplification takes place [DFS16].

Admittedly, the question whether masking in prime fields can contribute to improving the security vs. efficiency tradeoff of embedded cryptographic implementations is a long-term one. First results towards confirming the interest of this idea have been recently presented in [MMMS22], but they also highlight that concrete usability will require solving many challenges spanning different research areas. More precisely, the authors showed that the security gains of prime-field masking over Boolean masking (and binary encodings in general) can reach orders of magnitude in case of low-noise high-precision attacks. They additionally showed that a portion of these gains is maintained even in case of noisier settings. However, the arithmetic encoding of intermediate values in prime-order fields is not directly applicable to most commonly used symmetric cryptographic primitives, especially considering implementation-friendly block ciphers. The authors therefore proposed a cipher with similarities to the AES operating in prime fields – denoted as `AES-prime` – as a useful research object (that is clearly not ready for practical use). This choice enables first comparisons between the AES and the `AES-prime` ciphers[2], which share a similar (though not identical) structure and operate in fields of similar sizes – something that would neither be feasible with large-field prime ciphers from the literature [GRR+16, AGR+16, GKR+21] nor with state-of-the-art bitslice ciphers operating in characteristic-2 fields [MMMS22].

Our main contributions to this emerging state of the art are twofold.

First, we observe that in contrast with binary masking, the squaring operation is not linear in prime fields and therefore cannot be trivially implemented by squaring each share independently. We therefore propose a novel arbitrary-order squaring gadget in prime fields, which significantly improves over a naive implementation using a secure multiplication algorithm "à la ISW" [ISW03], and demonstrate its composability guarantees in the robust

[1] https://csrc.nist.gov/Projects/lightweight-cryptography

[2] A cautionary note regarding the fairness of this comparison can be found in Section 2.4

probing model [FGP+18]. We additionally present two optimized versions of the squaring gadget for 2 and 3 shares with a reduced latency of only one clock cycle.

Second, we investigate the security vs. performance tradeoff of prime masking in comparison with Boolean masking. For this purpose, we use the previously proposed `AES-prime` operating in $\mathbb{F}_p$ with $p = 2^7 - 1$, implement it in a provably secure manner (thanks to compositional reasoning) for 1, 2, 3 and 4 shares, and evaluate its performance and side-channel security on FPGA, considering both attacks exploiting the dynamic and static power consumption, and its performance only on ASIC. For the dynamic attacks, we further emulate strong adversaries by allowing to repeat traces with the same randomness in order to average out physical noise. For static attacks, we consider the standard adversary model with control over the clock to make use of intra-trace averaging (see details and reference list in Section 2.2). Besides, we perform the same analysis for the standard (binary) AES, using the state-of-the-art implementation proposed in [MCS22].

Despite specific to the selected algorithms and implementation context, our results show first complete examples where the cost vs. security tradeoff of prime masking is significantly better than the one of Boolean masking for comparable algorithms. So on the one hand, they extend the seed results of [MMMS22] towards full cipher comparisons, confirming that there are concrete cases where masked prime ciphers require smaller overheads than binary ones to reach a given security level. And on the other hand, they open the way towards the generalization of these seed observations. The design of new ciphers adapted to the specificities of prime masking appears as an important direction in this respect. While the observed physical security benefits of the `AES-prime` over the AES confirm the potential interest of prime masking, which is a necessary first step, we clarify in Section 2.4 that these ciphers are not perfectly comparable nor fully optimized for masking. Hence, the important long-term research question is whether an optimized prime cipher can lead to similarly positive conclusions when compared with the best bitslice ciphers of the literature. Interestingly, our first contribution (i.e., the efficient squaring gadget that can be masked at arbitrary orders) is also a useful step in this direction, as it suggests prime ciphers using power maps that leverage this efficient (non-linear) squaring as interesting candidates for side-channel secure implementations.

2 Background

We next introduce the background necessary for understanding our results and provide a cautionary note about the comparability of `AES-prime` with the standard AES.

2.1 Masking and Glitch-Resistance

Masking (a.k.a. secret sharing) is a popular countermeasure against SCA. It was first proposed in [CJRR99, GP99] and has become a topic of intense research attention since then. Analyzing masking schemes usually works by combining different tools. At the more abstract level, the probing model introduced by Ishai et al. guarantees that an adversary must observe a minimum number of intermediate values in an implementation to break it [ISW03]. It is usually instrumental to evaluate whether the randomness used in masked computations is sufficient and whether these computations are composable [BBD+16]. At the more concrete level, it is expected that probing secure masking schemes lead to implementations secure in the noisy leakage model (i.e., implementations that can only be broken with many measurements) [PR13]. Interestingly, it has been shown that as long as the leakages of the different shares are sufficiently noisy and independent, probing security implies noisy leakage security [DDF14, DFS15]. This provides a principled way to design secure implementations, by first analyzing their probing security, then evaluating

whether leakages are indeed sufficiently independent (by checking their statistical security order [SM16]) and finally analyzing the noise in the implementation [JS17].

Despite this conceptual simplicity, implementing masking schemes securely (i.e., ensuring the noise and independence assumptions are fulfilled) is known to be a tedious task. In particular in hardware, it has been shown that physical defaults such as glitches can completely break the independence assumption and significant efforts have been devoted to the design of schemes that can cope with such defaults [NRR06, NRS11, RBN+15, GMK16, GMK17, GM18]. Combining heuristic resistance against physical defaults with composition guarantees turned out to be challenging as well [MMSS19], and as a result, it was proposed to integrate physical defaults in a robust version of the probing model [FGP+18]. Eventually, it was shown that ensuring composition with Probe Isolating Non-interference (PINI) [CS20] was quite suitable for the analysis of masked implementations with physical defaults [CGLS21]. A couple of recent works build on such tools to design efficient higher-order masked implementations and integrate other physical defaults such as registers' transitions in their analysis [CS21, KMMS22, KSM22, MKSM22, MCS22].

In this work, we analyze the security of our implementations in the glitch-robust probing model [FGP+18], where the adversary can place t *glitch-extended* probes in the masked circuit. Each glitch-extended probe placed on a wire in the circuit produces as leakage the value carried by all the wires in the combinatorial circuit that computes the probed wire. In this model, a set of glitch-extended probes P in a masked gadget can be simulated by a set of input shares A of G if there exists a randomized simulator algorithm S taking as input the values of the wires in A such that the joint distribution of the leakage of the probes is equal to the distribution of the output of S, for any fixed value of the input shares of G [BBD+16, CS20]. This simulation notion is at the core of the Non-Interference (NI), Strong Non-Interference (SNI) and Probe-Isolating Non-Interference (PINI) definitions that characterize gadgets and enable compositional reasoning about masking.

Definition 1 (Glitch-robust (strong) non-interference [CS20])**.** A gadget G is glitch-robust t non-interferent (resp., strongly non-interferent) if any set of t_1 glitch-extended probes in G and t_2 glitch-extended probes on the output shares of G can be simulated by a set of at most t (resp., t_1) of its input shares, for any t_1 and t_2 such that $t_1 + t_2 \leq t$.

Definition 2 (Glitch-robust probe-isolating non-interference [CS20])**.** Let P be a set of t_1 glitch-extended probes in a gadget G and B be a set of share indices (the index of a share is its position in a sharing, starting at 0). The gadget G is glitch-robust t probe-isolating non-interferent if, for any P and B such that $|P| + |B| \leq t$, there is a set A of at most $|P|$ share indices such that the probes in P and glitch-extended probes on the output shares of G with index in B can be simulated by the input shares of G with index in $A \cup B$.

The gadgets that satisfy these definitions can be composed together. In the following, we will mainly use the trivial composition of PINI gadgets: any gadget that can be written as a composition of PINI gadgets is itself PINI [CS20]. In Subsection 3.2, we will also use the NI and SNI properties for optimization purposes, e.g., the fact that connecting a single-input SNI gadget to the output of an NI gadget is SNI [BBD+16].

2.1.1 Masking with Insufficient Noise

Besides the (well investigated) difficulties raised by the independence condition, it is also known that Boolean masking without sufficient noise level cannot provide an exponential increase of the attack complexity in the number of shares. This issue already takes place in case of single-target attacks [SVO+10, Sta18, Moo19] and is naturally amplified when considering multi-target (a.k.a. horizontal) attacks [BCPZ16]. While low-noise attacks have first been regarded as a less critical problem, since a sufficient amount of noise is

usually inherent to real-world SCA settings, advances towards more powerful equipment and stronger statistical approaches targeting the noise of masked implementations make it more critical. This is especially true in the context of low-end embedded devices [BS21] and in the context of hardware implementations when static leakage is exploitable (see the following subsection). Interestingly, this issue has also been considered from the theoretical viewpoint by Dziembowski et al., who studied the amplification of noisy leakages and conclude that prime-order fields are better suited for low-noise masking [DFS16]. A recent report confirms the practical impact of this observation by providing first information theoretic evaluations of a concrete cryptographic primitive additively masked with a prime-field encoding [MMMS22]. It additionally shows that the interest of prime encodings goes beyond the low-noise context (i.e., is maintained for higher noise levels), raising the question whether the overheads of prime encodings (for suitable prime ciphers) could not be compensated by the higher physical security levels they provide.

2.2 Static Power Side-Channel Analysis

Static power side-channel attacks became a relevant threat for the physical security of cryptographic hardware when modern integrated circuits started to leak observable currents even during idle times, i.e., in the absence of active computation. Such leakage currents occur when electrical energy is transferred across a boundary which is supposed to act as an insulator. In the constant pursuit of performance gains along with area and cost savings, leakage currents, as a byproduct of size and voltage scaling, have manifested themselves as a permanent burden for the power budget of modern integrated circuits. Due to the structure of complementary logic gates and the concrete arrangement of field effect transistors to realize common logic operations, leakage currents in computing devices depend heavily on the data that is currently processed inside the circuit. In the late 2000s, researchers have first proposed and simulated attacks on cryptographic implementations based on the exploitation of this static power consumption [GSST07, AGST09]. A few years later, the first practical demonstration of such attacks has been presented [Mor14, PSKM15]. Since then, a few dozen articles on the topic have been published in academic literature. Notably, they (almost) exclusively target hardware implementations of cryptographic primitives. The sequential nature and short life cycle of the data processed by the arithmetic logic unit in a microcontroller makes this platform a less attractive target for such attacks. The publications focusing on hardware attacks on the other hand can be sub-divided into case studies on custom ASIC designs [PSKM15, MMR17, Moo19, KMM19, MMR20, Moo20, MM21] and on reconfigurable FPGAs [Mor14, BCS+17, BST18, BBOS20, BSS21]. One interesting aspect of this side-channel, which was first mentioned in [Mor14, PSKM15] and later demonstrated by [MMR17, Moo19, MM21], is the threat it poses for masking schemes and other countermeasures that require a certain noise level to be effective. This is mainly caused by the attackers' ability to reduce the noise in the measurements through intra-trace averaging. The persistence of the leakage currents in an idle state, as opposed to the dynamic power consumption of a temporary transition, allows adversaries to obtain high-precision measurements (which may even be enhanced in case voltage and temperature can be controlled [Moo19]). Unless the cryptographic module under attack keeps sensitive intermediate data in the circuit when entering an idle state, the adversary typically requires control over the clock signal to perform such an analysis efficiently. Similar to traditional power side-channel analysis, template attacks have been suggested as a powerful strategy to extract the underlying key material [BDST17, BST18, XH19]. Finally, there have also been countermeasures proposed to thwart the exploitation of this side-channel. Standard-cell based solutions are for example presented and compared in [BBOS20, BSS21, MM21].

2.3 AES-prime

The `AES-prime` has been proposed in [MMMS22] as a variant of the standard AES cipher that works in prime fields. In short, it is based on the same round processing steps (i.e., `AddRoundKey`, `SubBytes`, `ShiftRows` and `MixColumns`) adapted so that operations are performed in $\mathbb{F}_p$, where $p = 2^n - 1$ is a Mersenne prime.

Our following experiments will be based on an instance with $n = 7$. Concretely, `AddRoundKey` and `ShiftRows` are direct adaptations of their standard counterpart while `SubBytes` and `MixColumns` have been modified in the following manner:

For `SubBytes`, rather than using the inverse function in $\mathbb{F}_{2^8}$, the power map $f(x) = x^5 + 2$ is used. It is the cheapest bijection — i.e. of least degree — without fixed point. Please note that in contrast to the standard AES, the `AES-prime`'s inverse S-box is less efficient than its S-box (i.e., it has a larger multiplicative depth), making the cipher primarily suitable for inverse-free modes of operation.

For `MixColumns`, a slightly modified Vandermonde matrix is used to minimize the number of different elements while keeping the Maximum Distance Separable (MDS) property:

$$M = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 2 & 4 & 16 \\ 1 & 4 & 16 & 2 \\ 1 & 16 & 2 & 4 \end{bmatrix}.$$

We refer to [MMMS22] for evidence that such an `AES-prime` design can be secure with a similar number of rounds as the standard AES. Precisely, this reference suggests that 11 rounds should be enough to prevent known attacks and advocate for a security margin of a few rounds given the novelty of the design. Our next evaluations use the recommended 14 rounds. We note that other instances of `AES-prime` could be defined (e.g., with larger primes) and it is for example an interesting open problem to study whether increasing p (e.g., up to 32-bit primes) can lead to significant security improvements.

2.4 Cautionary Note: AES-prime vs. AES

We stress that the efficiency and security comparisons between the `AES-prime` and the AES presented in Sections 4 and 5 need to be interpreted with caution. We use the `AES-prime` as a first target to study the potential benefits of prime masking, due to a number of similarities it shares with the standard AES. Yet, on the one hand, the amount of external cryptanalysis these ciphers received is incomparable. On the other hand, despite the `AES-prime` being the most suitable currently available primitive for our investigation, there remain crucial differences to the standard AES cipher that we highlight next:

1. AES-128 has a key and block size of 128 bits and claims 128-bit security (attack complexity $\geq 2^{8 \cdot 16}$) with 10 cipher rounds. `AES-prime` with the parameters proposed for $p = 2^7 - 1$ in [MMMS22] has a key and block size of 112 bits, and aims to provide 112-bit security (attack complexity $\geq 2^{7 \cdot 16}$) with 14 rounds.

2. The S-box and inverse S-box of the AES share the same complexity, since both are based on finite field inversion and only differ in the associated affine mapping. The `AES-prime` S-box is based on a small power map which can be computed efficiently, but whose inverse is significantly less efficient especially when masked, due to a larger multiplicative depth, making the decryption routine unattractive to implement. Hence, as detailed in [MMMS22], it would mostly be interesting for inverse-free modes of operation. In more detail, the inverse AES-prime S-box can be computed as $(x-2)^{101}$, which has a multiplicative depth of 7 and can be implemented with 6 squarings and 3 multiplications (roughly 3 × more expensive than the regular Sbox

with depth 3, 2 squarings and 1 multiplication). In this respect, we recall that the multiplicative depth of a function is the main factor driving the latency of masked hardware implementations, as consecutive non-linear operations generally need to be separated with register stages [NRS11, GPS14, BMD+20].

In general, we acknowledge that a completely fair comparison is not possible between these two primitives, because of both, differences in their design and differences in the understanding of their black-box security. However, we also insist that for our results to be relevant, we only require that comparable ciphers working in prime or binary fields of similar sizes can be secure with a similar number of rounds. For now we have not discovered any technical caveat that invalidates this view and the emergence of prime-based cipher designs also points in that direction [GRR+16, AGR+16, DEG+18, DGGK21, CHK+21, GKR+21]. So more than the specific results obtained for the `AES-prime`, which is anyway not fully optimized for prime-field masking, our investigations should be understood as an encouragement to design new prime ciphers that can compensate for the implementation overheads that operations in prime fields imply with strong side-channel security gains.

3 Hardware Gadgets and Composition

While secure (masked) multiplication algorithms are generally defined for arbitrary fields, working in a prime field comes with the specificity (compared to Boolean masking) that squaring operations are non-linear. In this section, we therefore study how to perform secure squaring operations in prime fields more efficiently than by naively using a secure multiplication algorithm. We then show how to integrate them securely into an `AES-prime` S-box thanks to compositional reasoning. We complete these new results by referring to the standard AES masked design that we will use in our comparisons.

3.1 PINI Squaring Gadgets

Algorithm 1 Masked squaring (glitch-robust PINI) with d shares.

Input: Sharing $\mathbf{a}$.
Output: Sharing $\mathbf{b}$ such that $b = a^2$.

1: **for** $i = 0$ to $d-1$ **do**
2: **for** $j = i+1$ to $d-1$ **do**
3: $r_{ij} \xleftarrow{\$} \mathbb{F}_p$
4: $r'_{ij} \xleftarrow{\$} \mathbb{F}_p$
5: $\alpha_{ij} \leftarrow 2\mathbf{a}_j + r_{ij}$
6: $\beta_{ij} \leftarrow \mathbf{a}_i \operatorname{Reg}(\alpha_{ij}) + r'_{ij}$
7: **for** $i = 0$ to $d-1$ **do**
8: $\gamma_i \leftarrow \mathbf{a}_i \left(\mathbf{a}_i - \sum_{j=i+1}^{d-1} r_{ij}\right)$
9: $\mathbf{b}_i = \operatorname{Reg}(\gamma_i) + \sum_{j=i+1}^{d-1} \operatorname{Reg}(\beta_{ij}) - \sum_{j=0}^{i-1} r'_{ji}$ ▷ Blue Reg not needed for PINI.

We introduce in Algorithm 1 a glitch-robust PINI squaring gadget with a latency of 2 cycles, using $d(d+1)/2$-1 multiplications, 1 squaring and $d(d-1)$ random elements.[3] This algorithm is correct: every term $\mathbf{a}_i r_{ij}$ in α_{ij} is canceled by the opposite term in γ_i, and every term r'_{ij} in β_{ij} is canceled by the opposite term in $\mathbf{b}_j$, which leaves only the terms $\mathbf{a}_i^2 + 2\mathbf{a}_i \sum_{j=i+1}^{d-1} \mathbf{a}_j$ in the output share $\mathbf{b}_i$ not canceled.

[3] The registers given in Algorithm 1 are the ones that are needed to ensure security against glitches. Other registers may be added freely (e.g., in order to allow pipelining for performance reasons).

Let us now prove the security of this new secure squaring gadget.

Proposition 1. *The masked squaring gadget (Algorithm 1) with d shares is $(d-1)$-glitch-robust PINI (even when the blue register of Line 9 is removed).*

Proof. Without loss of generality, we consider glitch-extended probes only on $\mathbf{b}_i$, α_{ij} and β_{ij}: all the other extended probes are strictly less powerful than these probes (i.e., they expand to a subset of the expansion of one of these probes). Given a set of extended probes P and probed output indices B, the PINI simulator proceeds as follows. Starting with $I \leftarrow B$, for every $\mathbf{b}_i$ probe in P, it adds i to I. Next, for every α_{ij} probe, if $j \notin I$, it adds j to I, otherwise it adds i to I, and then, for every β_{ij} probe, if $i \notin I$, it adds i to I, otherwise it adds j to I. Let $A = I \setminus B$, we observe that the set of input shares required for the simulation A (in addition to the indices in B) satisfy the PINI definition.

Let us now show how to simulate the probes. First, for every α_{ij} probe, sample uniformly at random a value for r_{ij} and save its value, then output r_{ij} and $\mathbf{a}_j$. Next, for every β_{ij} probe, if $j \in I$, sample uniformly at random a value for r_{ij} and r'_{ij} and save their values (if the value is not saved yet, otherwise use the saved value) then output $\mathbf{a}_i$, $\alpha_{ij} = \mathbf{a}_j + r_{ij}$ and r'_{ij}. Otherwise, sample and output r'_{ij} as before, but simulate α_{ij} as a fresh uniformly random element, and output $\mathbf{a}_i$. Finally, for every $\mathbf{b}_i$ probe and for every $i \in B$, output $\mathbf{a}_i$ and sample uniformly at random (if it is not saved yet) and output r_{ij} for $j = 0, \ldots, i-1$. Then, for every $j = i+1, \ldots, d-1$, if $j \in I$, sample uniformly at random (if it is not saved yet) and output r'_{ij} and $\mathbf{a}_j$, otherwise simulate β_{ij} as a fresh uniformly random element. Let us remark than in all cases where $\mathbf{a}_i$ or $\mathbf{a}_j$ is needed for the simulation, the corresponding index belongs to I. Moreover, the values produced by the simulator allow to reconstruct trivially the full values of the extended probes.

Let us now show that the simulation is correct, that is, that the distribution of the values computed by the simulator is the same as the one of the real gadget. This is obvious since the simulator behaves in the same way as the gadget, except for two cases: when it simulates (i) α_{ij} or (ii) β_{ij} as a fresh random. In the case (i), $j \notin I$ and β_{ij} is probed, which implies that $\mathbf{b}_i$ and α_{ij} are not probed. Therefore, r_{ij} is only observed by the adversary through the probe β_{ij} in the form of $\alpha_{ij} = \mathbf{a}_j + r_{ij}$. Therefore, simulating α_{ij} as a fresh random is correct. In the other case, $j \notin I$ and $\mathbf{b}_i$ is probed (or equivalently, $j \in B$), therefore none of $\mathbf{b}_j$, β_{ij} and β_{ji} are probed, hence r'_{ij} is only observed through $\beta_{ij} = \mathbf{a}_i\alpha_{ij}$, which implies that simulating β_{ij} as a fresh random is correct. □

Let us now introduce optimized gadgets performing the same operation more efficiently in the cases where $d = 2$ and $d = 3$. Concretely, these optimized gadgets (given in Algorithm 2 and Algorithm 3) have a latency of only 1 clock cycle.

Algorithm 2 Masked squaring (glitch-robust PINI) with $d = 2$ shares.

Input: Sharing $\mathbf{a}$.
Output: Sharing $\mathbf{b}$ such that $b = a^2$.

1: $r \xleftarrow{\$} \mathbb{F}_p$
2: $r' \xleftarrow{\$} \mathbb{F}_p$
3: $\mathbf{b}_0 \leftarrow \mathbf{a}_0 \operatorname{Reg}(2\mathbf{a}_1 + r) + \operatorname{Reg}(\mathbf{a}_0(\mathbf{a}_0 - r) + r')$
4: $\mathbf{b}_1 \leftarrow \operatorname{Reg}(\mathbf{a}_1^2 - r')$ ▷ Blue Reg not needed for PINI.

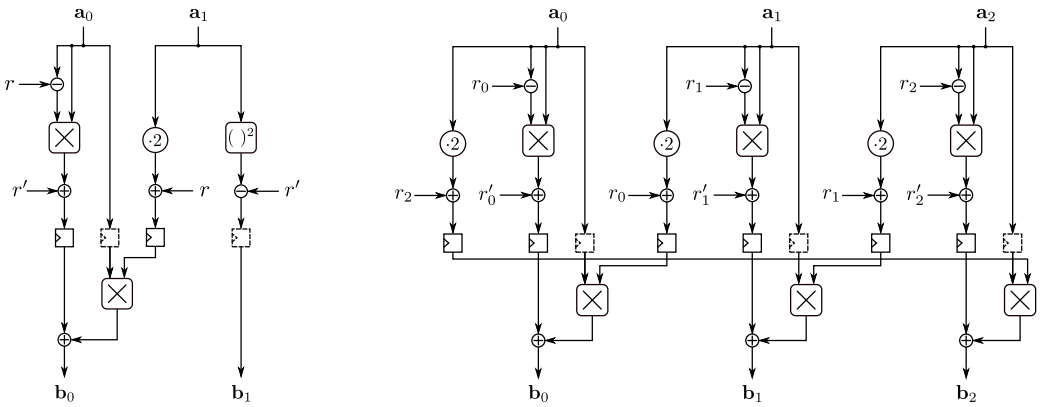
Proposition 2. *The squaring gadget (Algorithm 2) with 2 shares is 1-glitch-robust PINI.*

Proof. A probe on $\mathbf{b}_0$ is independent of $\mathbf{a}_1$, since $\mathbf{a}_0 \operatorname{Reg}(2\mathbf{a}_1 + r)$ is independent of $\mathbf{a}_1$ thanks to r, which is a fresh random, since $\operatorname{Reg}(\mathbf{a}_0(\mathbf{a}_0 - r) + r')$ is independent of r thanks

Algorithm 3 Masked squaring (glitch-robust PINI) with $d = 3$ shares.

Input: Sharing $\mathbf{a}$.
Output: Sharing $\mathbf{b}$ such that $b = a^2$.

1: **for** $i = 0$ to 2 **do**
2: $\quad r_i \xleftarrow{\$} \mathbb{F}_p$
3: $r'_0 \xleftarrow{\$} \mathbb{F}_p$
4: $r'_1 \xleftarrow{\$} \mathbb{F}_p$
5: $r'_2 \leftarrow -\mathrm{Reg}(r'_0 + r'_1)$ ▷ Blue Reg not needed for PINI.
6: **for** $i = 0$ to 2 **do**
7: $\quad \alpha_i \leftarrow 2\mathbf{a}_{(i+1) \mod 3} + r_i$
8: $\quad \beta_i \leftarrow \mathbf{a}_i(\mathbf{a}_i - r_i) + r'_i$
9: $\quad \mathbf{b}_i \leftarrow \mathbf{a}_i \,\mathrm{Reg}\,(\alpha_i) + \mathrm{Reg}\,(\beta_i)$



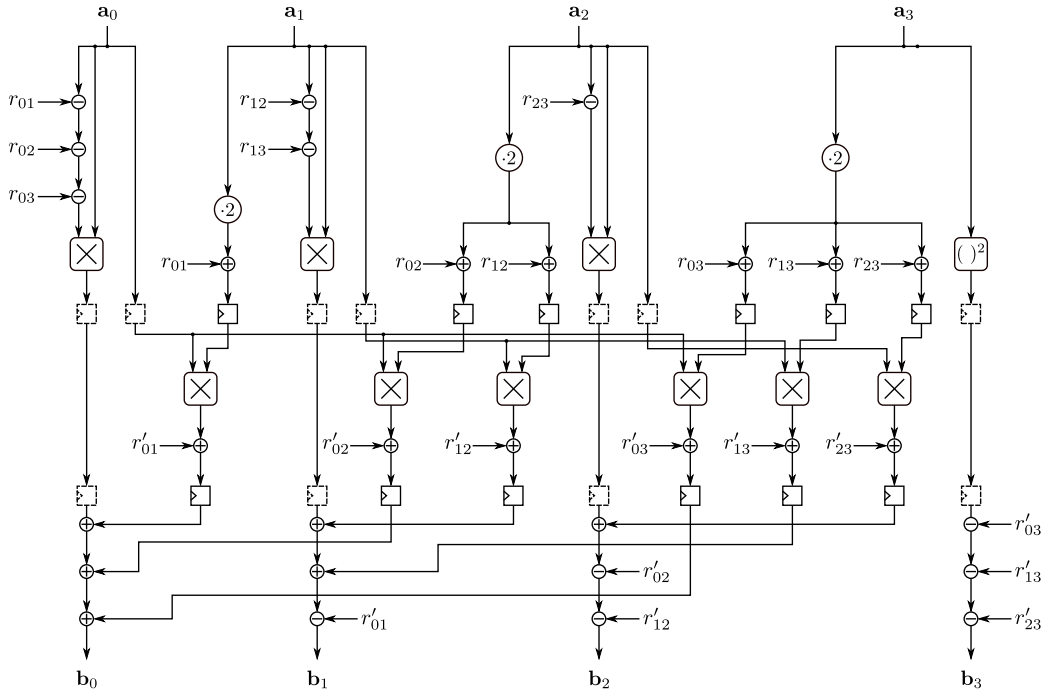
(a) 1-PINI SQ Gadget (b) 2-PINI SQ Gadget

Figure 1: Schematics of the optimized glitch-robust PINI gadgets for $d = 2$ and 3 shares in hardware. Optional registers (not needed for PINI conformity) are in dashed lines.

to the fresh r'. The other probes either lexically depend on only one of the input shares, or their expansion is a subset of the expansion of $\mathbf{b}_0$. □

Proposition 3. *The squaring gadget (Algorithm 3) with 3 shares is 2-glitch-robust PINI.*

Proof. Without loss of generality, we consider glitch-extended probes only on $\mathbf{b}_i$, α_i and β_i: all the other extended probes are strictly less powerful that these probes. If there is a single probe $\mathbf{b}_i$, it can be simulated while knowing only the input share $\mathbf{a}_i$, since β_i can be simulated as a uniform random (r'_i appears as a fresh random), and α_i can be simulated (r_i then appears as a fresh random). Any other single probe lexically depends on at most a single input share, hence it is trivial to simulate. Then, any set of two probes that does not contain any $\mathbf{b}_i$ lexically depends on at most two input shares: it is therefore trivial to simulate. Finally, let us assume that $\mathbf{b}_i$ is probed. If the other probe does not lexically depend on $\mathbf{a}_{(i+2) \mod 3}$, the simulation is trivial. Otherwise, r_i is not observed through the second, hence α_i can be simulated as a fresh random. Therefore, if the second probe does not lexically depend on both $\mathbf{a}_{(i+1) \mod 3}$ and $\mathbf{a}_{(i+2) \mod 3}$, simulation is trivial, knowing $\mathbf{a}_i$ and either $\mathbf{a}_{(i+1) \mod 3}$ or $\mathbf{a}_{(i+2) \mod 3}$ (depending on the dependency of the second probe). The only probe that depends lexically on both shares is $\mathbf{b}_{(i+1) \mod 3}$, but $\alpha_{(i+1) \mod 3}$ can be simulated as a fresh random since r_i is fresh. □



(a) 3-PINI SQ Gadget

Figure 2: Schematics of the glitch-robust PINI gadget for $d = 4$ shares in hardware. Optional registers (not needed for PINI conformity) are in dashed lines.

Figure 1 shows the optimized glitch-robust PINI squaring gadgets for $d = 2$ and 3 shares corresponding to Algorithms 2 and 3, respectively. Figure 2 shows an instantiation of the arbitrary-order glitch-robust PINI squaring gadget (Algorithm 1) for $d = 4$ shares. Please note that the $\cdot 2$ operation is merely a bit-rotation in Mersenne-prime-fields and that an unmasked squaring operation $( )^2$ is about half as expensive in hardware (in terms of circuit size) as an unmasked multiplication (unless executed by a pre-existing multiplier of course). We conclude this section with Table 1, showing that our (optimized) squaring gadgets require fewer partial multiplications than a generic multiplication gadget (asymptotically half), at the potential expense of a higher randomness usage (only for a large number of shares). As expected, for lower numbers of shares (i.e., $d = 2$ or 3 shares), the optimized squaring gadgets additionally reduce the number of clock cycles needed.

Table 1: High-level cost figures for a masked squaring operation using either the generic multiplication gadget HPC1 [CGLS21], or our optimized squaring gadgets.

Gadget	d	Multiplications	Squarings	Randomness	Reg. stages
HPC1	2	4	0	2	2
HPC1	3	9	0	5	2
HPC1	d	d^2	0	$d^2/2 + \mathcal{O}(d \log d)$	2
Algorithm 2	2	2	1	2	1
Algorithm 3	3	6	0	5	1
Algorithm 1	d	$d(d+1)/2 - 1$	1	$d(d-1)$	2

3.2 PINI `AES-prime` S-box

We compute a^5 as $a \cdot (a^2)^2$ using two squaring gadgets and a multiplication gadget. For the squaring, we use Algorithm 1 or optimized versions for low orders, and for the multiplication we can use a PINI gadget such as HPC1 [CGLS21]. Then, thanks to the glitch-robust composability of PINI gadgets, we know that the whole S-box is PINI, and it is therefore trivially composable when integrated in a full `AES-prime` design.

Algorithm 4 DOM-indep multiplication with d shares.

Input: Sharings $\mathbf{a}$ and $\mathbf{b}$.
Output: Sharing $\mathbf{c}$ such that $c = ab$.

for $i = 0$ to $d-1$ **do**
  for $j = i+1$ to $d-1$ **do**
    $r_{ij} \xleftarrow{\$} \mathbb{F}_p$
    $r_{ji} \leftarrow -r_{ij}$
for $i = 0$ to $d-1$ **do**
  $\mathbf{c}_i \leftarrow \mathbf{a}_i\mathbf{b}_i + \sum_{j=0, j\neq i}^{d-1} \mathrm{Reg}\left(\mathbf{a}_i\mathbf{b}_j + r_{ij}\right)$

Algorithm 5 `AES-prime` optimized S-box with d shares.

Input: Sharing $\mathbf{a}$.
Output: Sharing $\mathbf{d}$ such that $d = a^5$.

$\mathbf{b} \leftarrow \mathtt{Sq}(\mathbf{a})$ ▷ Algorithm 1.
$\mathbf{c} \leftarrow \mathtt{Sq}(\mathbf{b})$ ▷ Algorithm 1.
$\mathbf{d} \leftarrow \mathtt{Mul}(\mathbf{a}, \mathbf{c})$ ▷ DOM-indep (Algorithm 4).

This construction is however not optimal, and we next show that we can use the DOM-indep multiplication gadget from [GMK16], which is only glitch-robust NI and not PINI, instead of the HPC1 gadget, while still achieving PINI for the S-box (see Algorithm 5). The trivial composition based on the HPC1 gadget is compared to the optimized one described in the following with respect to its cost and performance in Appendix B. We will next prove that the sequential composition of two of our squaring gadgets (i.e., raising an element to the fourth using two instances of Algorithm 1) is a glitch-robust SNI gadget. It then follows (from Lemma 2 and Proposition 1 of [CGLS21]) that the S-box is glitch-robust PINI. Moreover, since we build the S-box as a pipeline and use it in a round-based implementation of `AES-prime`, the full cipher implementation is also transition-robust (thanks to [CS21, Corollary 2]). To prove that the composition of two squarings is glitch-robust SNI, we show two other properties for Algorithm 1: the first one is that it is slightly stronger than glitch-robust NI, as it does not allow to propagate glitches through the gadget, as stated and proved hereafter.

Lemma 1. *For any set P of glitch-extended probes in Algorithm 1 (with the blue Reg) and any set $B \subset \{0, \ldots, d-1\}$, there exists sets $A, A' \subset \{0, \ldots, d-1\}$ with $|A| \leq |P|$ and $|A'| \leq |B|$ such that the glitch-extended probes in P and glitch-extended probes on the output shares with index in B can be simulated by knowing glitch-extended probes on the input shares with index in A and (non-extended) probes on the input shares with index in A'.*

Proof. Without loss of generality, we consider glitch-extended probes only on α_{ij}, β_{ij} and γ_i. The simulator computes the required input shares as follows: starting with $A \leftarrow \emptyset$, for every γ_i probe, add i to A, then for every α_{ij} probe, if $j \notin A$, add j to I, otherwise add i to A, and finally, for every β_{ij} probe, if $i \notin A$, add i to A, otherwise add j to A. Next, for

every $i \in B$, if $i \notin A$, add i to A', otherwise if there is a β_{ij} in P such that $j \notin A$, add j to A'. (Let us remark that there is at most one such j by construction of A.)

The simulation works as follows: simulate all the probes and output shares with index in B by computing them (and the intermediate values involved), following Algorithm 1 and using the known input shares (and the known glitches on them). The only variable for which this cannot be done is β_{ij} when $j \notin A$. This variable has to be simulated when either β_{ij} itself is probed, or when $i \in B$ (hence $\mathbf{b}_i$ must be simulated). In such a case, since $j \notin A$, neither γ_i nor α_{ij} are probed. If $i \notin B$, then r_{ij} is only observed through β_{ij}, hence $\mathrm{Reg}\,(\alpha_{ij})$ can be simulated as a fresh random. Otherwise, if β_{ij} belongs to P, then by construction $j \in A \cup A'$, therefore $\mathrm{Reg}\,(\alpha_{ij})$ can be correctly simulated. The last case is when β_{ij} is not probed. Then, if j does not belong to $A \cup A'$, r'_{ij} is not observed except through β_{ij}, hence $\mathrm{Reg}\,(\beta_{ij})$ can be simulated as a fresh random. □

The second property is a weaker version of glitch-robust SNI, where the probes on the outputs cannot observe glitches, as stated and proved hereafter.

Lemma 2. *For any set P of glitch-extended probes in Algorithm 1 (even without the blue Reg) and any set $B \subset \{0, \ldots, d-1\}$ such that $|P| + |B| \leq d - 1$, there exists a set $A \subset \{0, \ldots, d-1\}$ with $|A| \leq |P|$ such that the glitch-extended probes in P and (non-extended) probes on the output shares with index in B can be simulated by knowing glitch-extended probes on the input shares with index in A.*

Proof. Without loss of generality, we consider that probes in P are only on α_{ij}, β_{ij} and $\mathbf{b}_i$. The simulator computes the required input shares as follows: starting with $A \leftarrow \emptyset$, for every $\mathbf{b}_i$ probe, add i to A, then for every α_{ij} probe, if $j \notin A$, add j to I, otherwise add i to A, and finally, for every β_{ij} probe, if $i \notin A$, add i to A, otherwise add j to A.

Again, the simulation can be done following Algorithm 1, except for the simulation of β_{ij} when $j \notin A$, and $\mathbf{b}_i$ when $i \notin A$. When $i \in B$ and $i \notin A$, then there exists some $j \in \{0, \ldots, d-1\}$ such that $j \neq i$ and r'_{ij} (if $j > i$) or r'_{ji} (if $j < i$) is not observed by the adversary, except through $\mathbf{b}_i$ (on which no glitch is observed). Therefore, $\mathbf{b}_i$ can be simulated as a fresh random. Next, if $\mathbf{b}_i$ belongs to P, we have $i \in A$, and we can simulate every term in the sum following Algorithm 1, except for $\mathrm{Reg}\,(\beta_{ij})$ where $j \notin A$, for which we know that they can be simulated as fresh randoms since r_{ij} is not observed through another probe. This last observation also applies to β_{ij} probes for which $j \notin A$. □

Proposition 4. *The fourth-power gadget G with d shares that, given an input sharing $\mathbf{a}$, computes $\mathbf{b} \leftarrow \mathtt{Sq}_1(\mathbf{a})$ then $\mathbf{c} \leftarrow \mathtt{Sq}_2(\mathbf{b})$ and outputs $\mathbf{c}$, where $\mathtt{Sq}_1$ and $\mathtt{Sq}_2$ are masked squaring gadgets (Algorithm 1, with the blue Reg) is glitch-robust $(d-1)$-SNI.*

Proof. Let P_i be the set of glitch-extended probes in $\mathtt{Sq}_i$ for $i = 1, 2$, and let $B \subset \{0, \ldots, d\}$ such that $|P_1| + |P_2| + |B| \leq d - 1$. Using Lemma 1 for $\mathtt{Sq}_2$, we have a simulator that simulates the glitch-extended probes P_2 and glitch-extended on the shares of $\mathbf{c}$ whose index belong to B. This simulator requires knowledge of glitch-extended probes on the shares of $\mathbf{b}$ with index in A (with $|A| \leq |P_2|$), and non-extended probes on the shares with index in A' (with $|A'| \leq |B|$). Using Lemma 2 for $\mathtt{Sq}_1$, we can simulate these values, as well as the glitch-extended probes in P_1 using glitch-extended probes on shares of $\mathbf{a}$ with index in A'', with $|A''| \leq |A| + |P_1|$ (we can apply the lemma since $|A| + |A'| + |P_1| \leq d - 1$). As a result, we can simulate all the required values using at most $|A''| \leq |P_1| + |P_2|$ input shares. □

Corollary 1. *The masked `AES-prime` S-box (Algorithm 5) is glitch-robust PINI.*

Proof. This is a direct consequence of [CGLS21, Corollary 1] (where the refresh restriction on the SNI gadget is lifted, since it is not used in the proof), [CGLS21, Proposition 1] and the previous Proposition 4. □

Let us now consider S-box implementations based on the squaring gadgets optimized at low orders. For $d = 2$ shares, we can use the same construction as in the arbitrary-order case, but we swap the shares of $\mathbf{b}$ to improve the logic depth balance in the datapath (see Algorithm 6). The security proof is the same as the one of Algorithm 5, except for the proof that the composition of the squaring gadgets is glitch-robust SNI.

Algorithm 6 `AES-prime` optimized S-box with $d = 2$ shares.

Input: Sharing $\mathbf{a}$.
Output: Sharing $\mathbf{d}$ such that $d = a^5$.

1: $\mathbf{b} \leftarrow \mathtt{Sq}(\mathbf{a})$ ▷ Algorithm 2.
2: $\mathbf{c} \leftarrow \mathtt{Sq}((\mathbf{b}_1, \mathbf{b}_0))$ ▷ Algorithm 2.
3: $\mathbf{d} \leftarrow \mathtt{Mul}(\mathbf{a}, \mathbf{c})$ ▷ DOM-indep (Algorithm 4).

Proposition 5. *The fourth-power gadget G that, given a 2-share input sharing $\mathbf{a}$, computes $\mathbf{b} \leftarrow \mathtt{Sq}_1(\mathbf{a})$ then $\mathbf{c} \leftarrow \mathtt{Sq}_2((\mathbf{b}_1, \mathbf{b}_0))$ and outputs $\mathbf{c}$, where $\mathtt{Sq}_1$ and $\mathtt{Sq}_2$ are masked squaring gadgets (Algorithm 2, with the blue Reg) is glitch-robust 1-SNI.*

Proof. Any internal probe in G can be simulated using one input share, since G is a sequential composition of single-input glitch-robust PINI gadgets. For output probes: $\mathbf{c}_1$ can be simulated as a fresh random (since r' is a uniform random), and it carries no glitches. For $\mathbf{c}_0$, we can simulate $\mathrm{Reg}\,(\mathbf{b}_1(\mathbf{b}_1 - r) + r')$ as a fresh random, thanks to r', which leaves $\mathbf{b}_1\,\mathrm{Reg}\,(2\mathbf{b}_0 + r)$. The latter value can be simulated by knowing only $\mathbf{b}_1$, thanks to r. Since $\mathbf{b}_1$ is the output of the first execution of Algorithm 2, it can be simulated without knowing any input share, following the same argument as for $\mathbf{c}_1$. □

Finally, let us consider the optimized implementation for $d = 3$ in Algorithm 7. The structure is similar to the previous S-box implementations, except that there is a register layer after the squarings. The proof is again the same, except for the SNI part.

Algorithm 7 `AES-prime` optimized S-box with $d = 3$ shares.

Input: Sharing $\mathbf{a}$.
Output: Sharing $\mathbf{d}$ such that $d = a^5$.

1: $\mathbf{b} \leftarrow \mathtt{Sq}(\mathbf{a})$ ▷ Algorithm 3.
2: $\mathbf{c} \leftarrow \mathrm{Reg}\,(\mathtt{Sq}(\mathbf{b}))$ ▷ Algorithm 3.
3: $\mathbf{d} \leftarrow \mathtt{Mul}(\mathbf{a}, \mathbf{c})$ ▷ DOM-indep (Algorithm 4).

Proposition 6. *The fourth-power gadget G that, given a 3-share input sharing $\mathbf{a}$, computes $\mathbf{b} \leftarrow \mathtt{Sq}_1(\mathbf{a})$ then $\mathbf{c} \leftarrow \mathtt{Sq}_2((\mathbf{b})$ and outputs $\mathbf{c}$, where $\mathtt{Sq}_1$ and $\mathtt{Sq}_2$ are masked squaring gadgets (Algorithm 3, with the blue Reg) is glitch-robust 2-SNI.*

Proof. Let us show that $\mathtt{Sq}_2$, composed with the output registers, is glitch-robust SNI. Moreover, $\mathtt{Sq}_1$ is glitch-robust NI (since it is glitch-robust PINI), therefore G is glitch-robust SNI. Without loss of generality, we consider glitch-extended probes only on $\mathbf{b}_i$, α_i, β_i and $r'_0 + r'_1$. If two shares of $\mathbf{c}$ are probed, each of these shares is re-masked by one element of the tuple (r'_0, r'_1, r'_2), which is a fresh and uniform sharing of 0. Both shares can therefore be simulated as fresh random values. Next, if $\mathbf{c}_i$ is probed and $\mathbf{a}_i$, β_i, $r'_0 + r'_1$ or $\alpha_{(i-1) \mod 3}$ is probed, then $\mathrm{Reg}\,(\alpha_i)$ can be simulated as a fresh random, and the remaining intermediate values in the probes can be simulated by following Algorithm 3 if $\mathbf{a}_i$ is known. If the other probe is another value, it is independent of r'_i, therefore $\mathbf{b}_i$ can be simulated as a fresh random and the other probe can be simulated with one input share (since the gadget is PINI). Finally, if there are two probes inside the gadget, they can be simulated by knowing two input shares since the gadget is PINI. □

3.3 PINI AES S-box

For the masked AES in binary fields, we rely on the PINI S-box architecture proposed in [MCS22]. It is based on the binary representation with 34 ANDs and 94 XORs presented in [BP12]. The PINI property is achieved by replacing each operation over $\texttt{GF}(2)$ by the corresponding gadget of the HPC2 masking scheme presented in [CGLS21]. The latter is a glitch-robust masking scheme for which the composability has been proven for an arbitrary number of shares. As explained in [MCS22], the input latency asymmetry of the HPC2 multiplication gadget may result in suboptimal implementation result if a naive instantiation strategy is used. Instead, the authors show that significant performance improvements, both in terms of area and latency, can be obtained by switching the input ports of some specific multiplication gadgets. It follows that the S-box is organized as a pipeline of 6 stages and requires $17 \cdot d(d-1)$ bits of fresh randomness per execution.

4 Hardware Cost and Performance

We now evaluate the cost and performance of (masked) `AES-prime` implementations based on the introduced gadgets and S-box constructions. We also compare the results to corresponding implementations of the classical AES[4]. We distinguish between ASIC implementations synthesized using a digital standard cell library and FPGA implementations where the circuits rely on a reconfigurable fabric. As observed in [MMMS22], field arithmetic modulo a Mersenne-prime can be realized with efficiency similar to binary field operations for fields of small size when dedicated multipliers and adders are available. For FPGA implementations, those are usually present in the form of DSP slices. For our ASIC implementations, we assume that all the logic has to be constructed from scratch.

It is important to mention that reduction modulo a Mersenne-prime $p = 2^n - 1$ can be performed efficiently using the well-known implementation trick shown in the following equation, where & denotes the logical *AND* and $\gg$ denotes a bit shift:

$$a = (a \ \& \ p) + (a \gg n).$$

However, when $a = p$ this procedure will not reduce a to 0. Instead $a = p$ remains unchanged. From an implementation perspective there is no reason to treat this special case anywhere in a cipher implementation except for the final output of a ciphertext. Any intermediate calculation can be continued with $a = p$. In our squaring gadgets and S-boxes we have therefore not included a corresponding check (which could lead to leakage), but only verify $0 \leq a \leq p - 1$ at the final output of the full `AES-prime` implementation.

The S-boxes analyzed in the following are built to internally synchronize and time the arrival of the random values to each sub-circuit using register elements. In that way, all fresh random inputs required for a masked execution can be supplied synchronously with the shared data and may be kept stable as long as desired until the next input should be processed. This strategy simplifies the integration of the S-boxes, minimizes the required control logic and is overall well-suited for a comparison as it requires no additional assumptions about the timing of input values. However, the additional registers lead to an area overhead compared to more optimized realizations such as [MCS22].

4.1 ASIC

For the ASIC based cost and performance evaluation we have chosen *Synopsys Design Compiler Version O-2018.06-SP4* as a synthesis tool and mapped our designs using a commercial 65 nm low-power CMOS standard cell library. We run three iterations of the `compile_ultra` command, one regular and two incremental, to optimize the mapping with

[4] A cautionary note regarding the fairness of this comparison can be found in Section 2.4

Table 2: Cost and performance figures for glitch-robust PINI AES and `AES-prime` S-boxes synthesized for minimum area using a 65 nm ASIC library.

S-box	Shares d	Area [GE]	Crit. Path [ns]	Reg. Stages	Randomn. [bits]
AES	1	278.00	4.23	0	0
	2	4577.00	1.26	6	34
	3	10070.00	1.46	6	102
	4	17767.00	1.65	6	204
`AES-prime`	1	790.00	10.64	0	0
	2	4682.75	9.32	3	35
	3	11467.50	10.54	4	91
	4	22335.00	7.66	5	210

Table 3: Cost and performance figures for glitch-robust PINI AES and `AES-prime` S-boxes synthesized for maximum frequency using a 65 nm ASIC library.

S-box	Shares d	Area [GE]	Crit. Path [ns]	Reg. Stages	Randomn. [bits]
AES	1	697.75	0.83	0	0
	2	4980.25	0.40	6	34
	3	10788.00	0.44	6	102
	4	18485.00	0.51	6	204
`AES-prime`	1	2165.75	1.81	0	0
	2	6449.50	1.93	3	35
	3	16493.75	2.21	4	91
	4	31992.75	1.50	5	210

the optimization goal set either to minimum area or to maximum frequency. In order to set these optimization goals we either leave the clock entirely unconstrained (minimum area) or we constrain the clock with a period that is impossible to achieve (maximum frequency). In the latter case, the synthesis tool tries to optimize the critical path delay as much as possible before eventually reporting a negative slack. Running the `compile_ultra` command several times incrementally helps to make sure that the minimum attainable critical path delay is indeed reached. Of course, optimizing a hardware implementation for one metric will typically lead to a significantly worse performance in other metrics that are not constrained. To be precise, when optimizing for minimum area we expect large critical path delays and for maximum operating frequency the reported area will increase considerably. For the masked implementations we have additionally passed the `-no_autoungroup` parameter to preserve all hierarchies. Tables 2 and 3 show the results, comparing the glitch-robust PINI AES and `AES-prime` S-boxes, when synthesized for minimum area and maximum clock frequency respectively (i.e., unconstrained clock vs. maximally constrained clock). The S-boxes are based on the Boyar-Peralta S-box [BP12] for the AES and on the calculation $a \cdot (a^2)^2 + 2$ for the `AES-prime`. Clearly, the unmasked AES S-box ($d = 1$) is significantly smaller and faster. However, the masked variants ($d > 1$) of the `AES-prime` S-boxes are comparable in size when synthesized for minimum area and can additionally be executed in fewer clock cycles due to a lower number of register stages. In the first case, their critical path delay is significantly larger, so the operating frequency is reduced. In the second case, when synthesized for maximum operating frequency (i.e., minimum critical path), the area gap between AES and `AES-prime` increases, but the delay gap decreases, although it is still a few times larger.

Tables 4 and 5 compare full round-based cipher implementations with 20 parallel S-box

Table 4: Cost and performance figures for masked round-based implementations of AES and `AES-prime` synthesized for minimum area using a 65 nm ASIC library.

Cipher	Shares d	Area [GE]	Crit. Path [ns]	Lat. [cycles]	Randomn. [bits]
AES	1	10659.50	5.48	10	0
	2	98376.75	1.88	70	6800
	3	210700.25	2.14	70	20400
	4	363278.00	2.52	70	40800
`AES-prime`	1	21949.50	16.45	14	0
	2	102831.25	14.20	56	9800
	3	233950.00	15.97	70	25480
	4	443702.50	12.64	84	58800

Table 5: Cost and performance figures for masked round-based implementations of AES and `AES-prime` synthesized for maximum clock frequency using a 65 nm ASIC library.

Cipher	Shares d	Area [GE]	Crit. Path [ns]	Lat. [cycles]	Randomn. [bits]
AES	1	17322.00	1.12	10	0
	2	115667.75	0.60	70	6800
	3	268535.00	0.61	70	20400
	4	447540.50	0.75	70	40800
`AES-prime`	1	44856.75	3.04	14	0
	2	201568.00	3.01	56	9800
	3	437586.75	3.62	70	25480
	4	863202.50	2.50	84	58800

instances (16 for the data path, 4 for the key schedule) when synthesized for minimum area and maximum operating frequency respectively. The unmasked `AES-prime` is about twice as large as the unmasked AES when mapped to standard cells. Yet, the differences are again much smaller when comparing the shared implementations. Besides, the latency of the `AES-prime` variant is quite similar to (and can even be lower than) the one of the AES. For example, the $d = 2$ version of the `AES-prime` requires only 56 cycles to execute all 14 rounds. As for the critical path, it follows the S-box trends and remains larger for the prime-field operations, reducing the maximum possible operating frequency. In summary, the minimum area of masked implementations is similar between `AES-prime` and AES on ASIC platforms, while the maximum frequency is several times higher for the AES. Overall, the two algorithms show performances that only differ by single-digit factors.

4.2 FPGA

We conduct a similar analysis of the (masked) S-boxes based on resource utilization reports on a modern FPGA device. We have selected a Xilinx Virtex UltraScale+ FPGA (XCVU3P-FFVC1517-3-E) for the comparison. The synthesis and implementation are performed using Xilinx Vivado v2022.1 (64-bit). For the masked implementations, the synthesis parameter `-flatten_hierarchy` is set to `none` while the `use_dsp` attribute is set to `yes` on all arithmetic multiplications and squarings (unless specified otherwise).

The resulting resource utilization and critical path delays are given in Tables 6 and 7 for minimum area and maximum operating frequency, respectively. Similar to the previous ASIC results, we let the tool optimize for minimum area by not constraining the clock at all, while optimizing for maximum operating frequency by providing an unrealistically

Table 6: Cost and performance figures for glitch-robust PINI AES and `AES-prime` S-boxes synthesized for minimum area using a Virtex UltraScale+ FPGA.

S-box	Shares d	FFs	LUTs	Slic.	DSPs	Crit. Path [ns]	Reg. Stag.
AES	1	0	87	15	0	3.69	0
	2	616	531	87	0	5.13	6
	3	1362	1259	207	0	5.82	6
	4	2400	2162	366	0	7.66	6
`AES-prime`	1	0	53	11	3	9.51	0
	2	133	379	87	10	9.15	3
	3	364	927	219	21	10.12	4
	4	1022	1885	444	36	9.71	5
`AES-prime`*	1	0	189	31	0	11.09	0
	2	133	963	184	0	10.00	3
	3	364	2271	432	0	11.92	4
	4	1022	4133	803	0	10.41	5

*Use of DSP slices prohibited

small clock period. As expected, the prime-field AES utilizes the multipliers offered by the DSP48E2 slices, while the binary one cannot since no support for the required polynomial Galois-field multiplication exists in this case. Since the costly multiplications are outsourced to the DSP slices, the remaining soft logic utilization is smaller than, or similar to, the classical masked AES. When prohibiting the use of DSP slices for the `AES-prime` implementations, the required amount of resources can be up to twice as high as for the AES (see lower parts of Tables 6 and 7). Yet, hardware support for arithmetic multiplications is available on all mainstream FPGA families and can easily be utilized for cryptographic operations in prime-fields on such devices.

In addition to the S-boxes, we implemented round-based versions of the ciphers on the FPGA. The results are listed in Tables 8 and 9. The comparison between AES and `AES-prime` leads to similar conclusions as for the S-box implementations. It is noteworthy that the smallest package of the Virtex UltraScale+ family (XCVU3P) provides 2280 DSP slices and therefore more than enough to support round-based `AES-prime` implementations with 20 parallel S-boxes and 4 shares (720 needed). The larger devices of the same FPGA family provide up to 12288 DSP slices (XCVU13P) and can even incorporate highly parallel higher-order masked implementations of prime ciphers with full DSP support.

It is worth mentioning that an FPGA-based cost and performance comparison using the concrete device mounted on the Sakura-G measurement board used for the experimental case study in Section 5 can be found in Appendix A. Appendix B additionally contains a comparison between the trivial composition of the `AES-prime` S-box using an HPC1 gadget as multiplier and the optimized composition that has been proven secure in Section 3. The differences in terms of area are small, but the critical path is increased more evidently when using the HPC1 gadget, especially for the FPGA results. Also, the trivial composition requires more randomness. However, for the concrete case of $d = 3$ the HPC1-based composition actually requires only 3 cycles. The HPC1 gadget has a two-cycle latency only with respect to one of the inputs and in the `AES-prime` S-box one input to the multiplication is ready before the other anyway, which allows efficient integration. Therefore, in some cases the trivial composition may be preferable over the optimized one, even despite higher randomness usage and larger critical path, due to a reduced latency.

In summary, like in the ASIC case, implementing the `AES-prime` on FPGAs admittedly comes with a price tag. Therefore, we next question whether this additional area and

Table 7: Cost and performance figures for glitch-robust PINI AES and `AES-prime` S-boxes synthesized for maximum operating frequency using a Virtex UltraScale+ FPGA.

S-box	Shares d	FFs	LUTs	Slic.	DSPs	Crit. Path [ns]	Reg. Stag.
AES	1	0	87	19	0	3.63	0
	2	616	513	101	0	5.08	6
	3	1362	1217	212	0	5.53	6
	4	2400	2203	523	0	6.79	6
`AES-prime`	1	0	72	24	3	6.99	0
	2	133	495	123	10	5.95	3
	3	364	1207	246	21	6.69	4
	4	1022	2462	487	36	7.02	5
`AES-prime`*	1	0	204	39	0	5.76	0
	2	133	1095	202	0	5.55	3
	3	364	2589	464	0	6.26	4
	4	1022	4774	834	0	7.13	5

*Use of DSP slices prohibited

reduced frequencies can be compensated by physical security gains. Optimally, if a smaller number of shares is sufficient to achieve the same practical security level, masked prime ciphers, despite coming with implementation overheads, can become preferable over classical (binary) approaches for applications where physical security matters.

5 Experimental Results

So far we have discussed the implementation cost of prime vs. binary masking in Section 4. We may now wonder about the side-channel security that can be reached for this cost. The following case study serves two main purposes. First, we want to verify experimentally whether the gadgets and the masked S-boxes proposed in Section 3 provide the desired security levels. This is also useful as a confirmation that the implementations analyzed for their cost and performance in Section 4 correspond to practically secure circuits (since common verification tools for masked implementations are not yet capable of dealing with prime-field arithmetic masking). Secondly, we want to quantify whether the price to pay for choosing `AES-prime` over AES is justified by the enhanced security level. We are strictly limiting ourselves to low-noise attacks in this work. It is already well-established that any masking scheme which is able to guarantee a statistical security order can be effective against passive physical attacks when independence and noise are satisfied to a sufficient degree. Especially with respect to hardware implementations, the latter requirement is typically expected to be fulfilled. This is mainly justified by the parallel processing of the individual shares of secrets, which makes it more difficult to obtain a high Signal-to-Noise Ratio (SNR) for individual intermediate values processed by the underlying implementation. However, as shown by [MMMS22], parallelism without noise is not sufficient to provide security amplification when leakage functions like Hamming weight or bit models are applicable. Thus, in order to identify whether a violation of the (physical) noise assumption can lead to problems for masked hardware implementations in practice, we consider strong adversary models and perform low noise attacks against our masked targets.

In particular, we are not considering the round-based circuits with 20 parallel S-boxes in this section, but only target circuits with a single S-box instance (in case of more parallel S-boxes the noise from the non-targeted instances could be significant and oppose our goal of low-noise analysis). We perform both dynamic and static power analysis attacks in the

Table 8: Cost and performance figures for unprotected round-based implementations of AES and `AES-prime` synthesized for minimum area using a Virtex UltraScale+ FPGA.

Cipher	Shares d	FFs	LUTs	Slic.	DSPs	Crit. Path [ns]
AES	1	269	2075	354	0	3.49
	2	12592	12207	1958	0	5.95
	3	27640	26527	4269	0	7.83
	4	48528	45785	7302	0	8.95
`AES-prime`	1	236	2446	427	60	18.10
	2	2898	11137	2182	200	16.26
	3	7630	23857	4598	420	17.92
	4	20903	44778	8925	720	14.61
`AES-prime`*	1	236	4770	839	0	16.91
	2	2898	22819	4179	0	16.96
	3	7630	51153	9119	0	18.60
	4	20903	90417	16552	0	16.92

*Use of DSP slices prohibited

Figure 3: Dynamic power setup including a Sakura-G board and a PicoScope 5224D.

following. In the former case, we allow the adversary to repeatedly measure traces with the same randomness, which would not be possible in a real-world attack setting due to a fresh selection of the random values for each iteration. We give the adversary this power to emulate a high-precision, low-noise measurement setup. Therefore the main source of noise in the measurements is of algorithmic nature (or related to limited informativeness of the underlying leakage function) and not due to a suboptimal measurement facility. For the static power side-channel attacks, we consider the standard adversary model using control over the clock signal to pause the computation when desired and to measure the leakage of data values that are currently applied to gates in the circuit. In both cases, we perform profiled Soft Analytical Side-Channel Attacks (SASCA) [VGS14] to extract secrets.

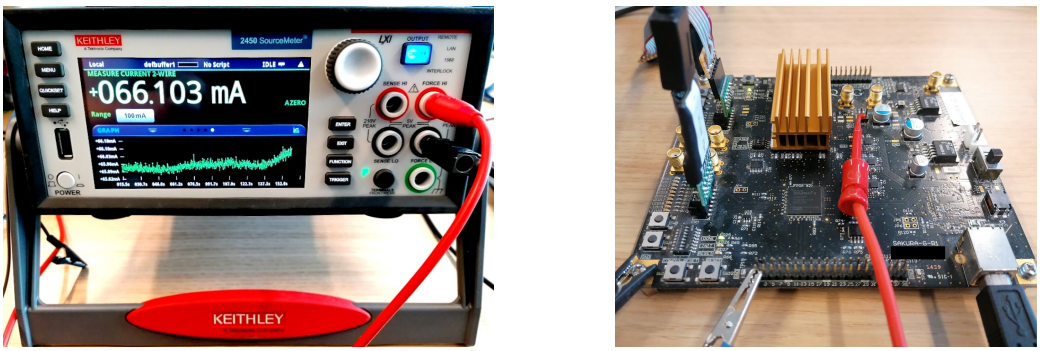
5.1 Setup

Our dynamic power setup is shown is Figure 3. We rely on the Sakura-G evaluation board [sak] and use the embedded Xilinx Spartan-6 FPGA with package XC6SLX75-2CSG484C as the target device. For each analyzed case, the design under evaluation was operated at 6MHz. The measurements were conducted using a Tektronix CT-1 current

Table 9: Cost and performance figures for unprotected round-based implementations of AES and `AES-prime` synthesized for max. frequency using a Virtex UltraScale+ FPGA.

Cipher	Shares d	FFs	LUTs	Slic.	DSPs	Crit. Path [ns]
AES	1	269	2221	371	0	1.89
	2	12598	12625	2202	0	3.17
	3	27642	27341	4611	0	3.87
	4	48529	46839	7908	0	4.05
`AES-prime`	1	271	3009	556	60	12.32
	2	2975	14731	2437	200	9.91
	3	7919	31544	5694	420	11.25
	4	21204	58688	10169	720	8.85
`AES-prime`*	1	236	5432	915	0	10.51
	2	2929	26824	4406	0	9.54
	3	7657	59476	9704	0	11.21
	4	20922	105080	16936	0	8.50

*Use of DSP slices prohibited

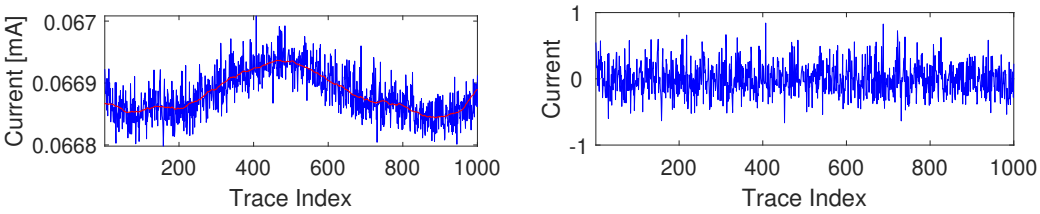


(a) Sourcemeter (b) Sakura-G Board

Figure 4: Static power setup showing the measurement instrument (Keithley 2450 SMU) on the left and the connected evaluation board (Sakura-G) on the right.

probe (with up to 1GHz bandwidth) directly connected to the power supply path of the target FPGA. For the acquisitions, we used a PicoScope 5244D digital oscilloscope sampling at 250 MS/s (i.e., 41.67 samples per clock cycle) with a vertical resolution of 12 bits. Using this setup we were able to record about 15 000 traces per second (250 time samples). The same tool flow as for the cost evaluation in Appendix A (i.e., Xilinx ISE v14.7 with `-keep_hierarchy` set to `yes`) was used to implement the evaluated designs.

The static power measurement setup is shown in Figure 4. The Keithley 2450 Sourcemeter on the left was used to power the target FPGA and subsequently to measure the static power consumption of the device. We have applied an increased core voltage of 1.6 V instead of the nominal 1.2 V in order to increase the leakage of the target [Moo19]. At this voltage, the device showed an idle-current of roughly 66 mA. Unlike previous works we have not controlled (or raised) the environmental temperature to increase the leakage. Yet, in order to dissipate the heat caused by the computation faster to the environment we have put passive cooling elements onto the FPGA package. This small detail had a noticeable positive impact on the stability of measurements. Additionally, as suggested in [MMR20], we have employed a moving average filter to post-process the traces in order



(a) raw trace set (blue), moving average (red)

(b) normalized trace set

Figure 5: Static power sample set before (left) and after (right) moving average correction.

to remove noise caused by low-frequency temperature changes. The effect of such a filtering is illustrated exemplarily in Figure 5. Using this setup we were able to record about 5.85 traces per second (approximately 171 milliseconds per measurement).

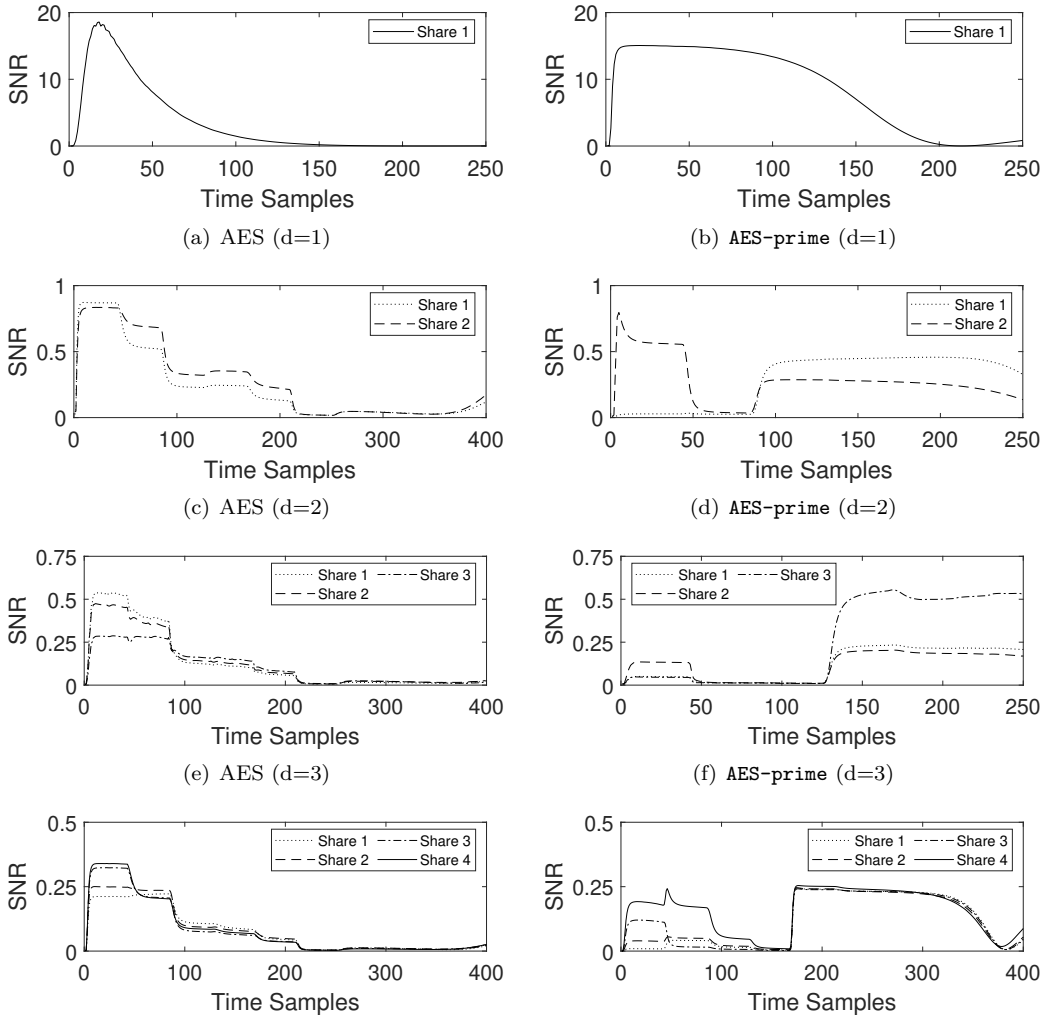
5.2 Dynamic Power Attacks

Method. We have measured our FPGA implementations during the execution of one (masked) S-box and repeated each trace (with the same randomness) 10 times before averaging them to one low-noise trace. In the following, we present our results and always compare the classical AES circuits to `AES-prime` ones. The attack results are obtained by selecting 1 000 000 traces for random inputs from the fixed-vs-random measurement sets for the profiling, and then performing a pooled template attack. In this process, the dimensionality of the measurement data is reduced from 250 or 400, respectively, to 5 dimensions using Linear Discriminant Analysis (LDA) [SA08]. As shown in Figure 6, the SNRs are not sparse, so a preliminary selection of points of interest would not have been relevant here. Once the probabilities for each share are computed from the LDA, we compute the discrete probability distribution of the secret value. Since the latter one is encoded by a linear sharing (with respect to the underlying Boolean/arithmetic scheme), the probability distribution to compute is the convolution product of the distributions of the corresponding shares. This can be implemented as a simple SASCA tree graph inside the SCALib library.[5] Once the probability distribution of the secret value for each of the validation traces are computed, we estimate the average rank of the correct key thanks to a re-sampling method, where the attacks are averaged 1 000 times (which can be done without bias since our number of attack traces is far from the size of the validation set).[6]

Results. For the case of unmasked S-boxes the results are depicted in Figure 7, the corresponding illustrations for masked implementations with 2, 3 and 4 shares are shown in Figures 8, 9 and 10, respectively. In all figures, the top row shows a comparison of sample traces, the middle row presents results of a non-specific fixed-vs-random t-test [SM16], while the bottom row shows the rank of the correct key candidate over the number of traces as a result of a profiled SASCA. The TVLA results are merely used to confirm that all circuits indeed deliver the targeted statistical security order. The resistance of targets is then judged by the attack performance. When comparing the physical security properties of AES and `AES-prime` implementations based on the presented data, it is clear that the prime-field masked AES provides a significantly higher security for the same number of shares (1-2 orders of magnitude). It can for example be observed that the 3-share `AES-prime` S-box shows similar (TVLA) and even better (SASCA) security in our metrics than the 4-share AES S-box. Thus, we may conclude that considering low-noise attacks,

[5] https://scalib.readthedocs.io/en/latest/index.html

[6] Please note that the axes of the related plots are shown in logarithmic scale. Therefore, even small differences on the x-axis can already constitute an order of magnitude difference in the attack complexity.



(a) AES (d=1)
(b) `AES-prime` (d=1)
(c) AES (d=2)
(d) `AES-prime` (d=2)
(e) AES (d=3)
(f) `AES-prime` (d=3)
(g) AES (d=4)
(h) `AES-prime` (d=4)

Figure 6: SNRs of all individual shares in the dynamic power attacks.

prime-field masking comes with security benefits that can sometimes cover the performance losses discussed in the previous section. We insist that, as illustrated by Figure 6, these benefits are not caused by a reduced Signal-to-Noise Ratio (SNR) in the prime-field case, but indeed by the algebraic incompatibility of leakage function and prime-field operations (since the SNRs per share are similar for both encodings). We also note that the theoretical security analysis of prime masking in [DFS16] only requires the leakage function to be non-injective (which then leads to trivial attacks anyway) for security amplification. So the interest of prime masking should be quite technology-independent. The only thing that could decrease its comparative advantage over binary masking is if leakage functions became much more complex (e.g., close to random) so that their compatibility with Boolean masking strongly decreases. As a result, the security of Boolean masking would increase in low-noise settings (while the security of prime masking would remain unchanged). We deem this quite unlikely given the current trends in device leakage.

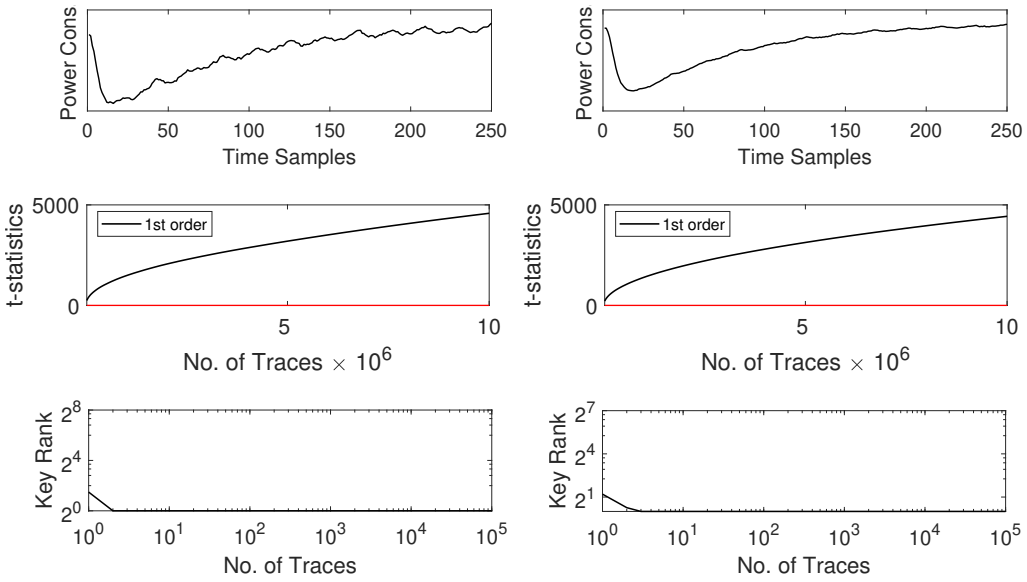


Figure 7: Comparison of unmasked S-boxes (dynamic power). Top to bottom: sample traces, TVLA results, profiled attack results. Left to right: AES, `AES-prime`.

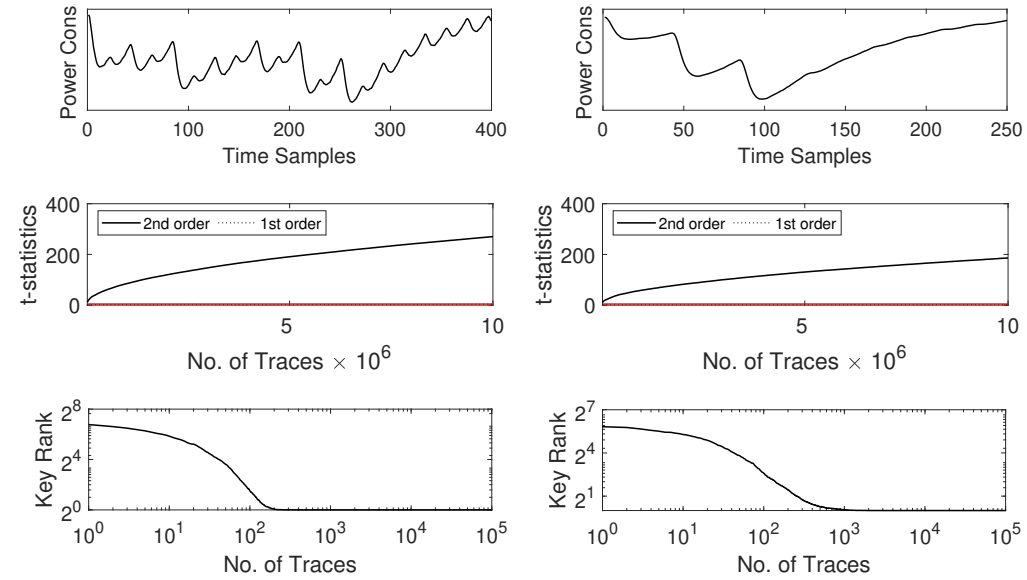


Figure 8: Comparison of first-order masked S-boxes (dynamic power). Top to bottom: sample traces, TVLA results, profiled attack results. Left to right: AES, `AES-prime`.

5.3 Static Power Attacks

Method. In addition to the dynamic power analysis of the previous section, we also performed static power attacks on the same targets. Static power side-channel analysis is known as an attack vector that typically requires a stronger adversary model since a certain degree of control over the execution is (most often) needed. Yet, if this requirement is satisfied the capabilities of the adversaries are quite impressive. In particular, the persistence of the leakage currents in idle states can be exploited to measure the static power

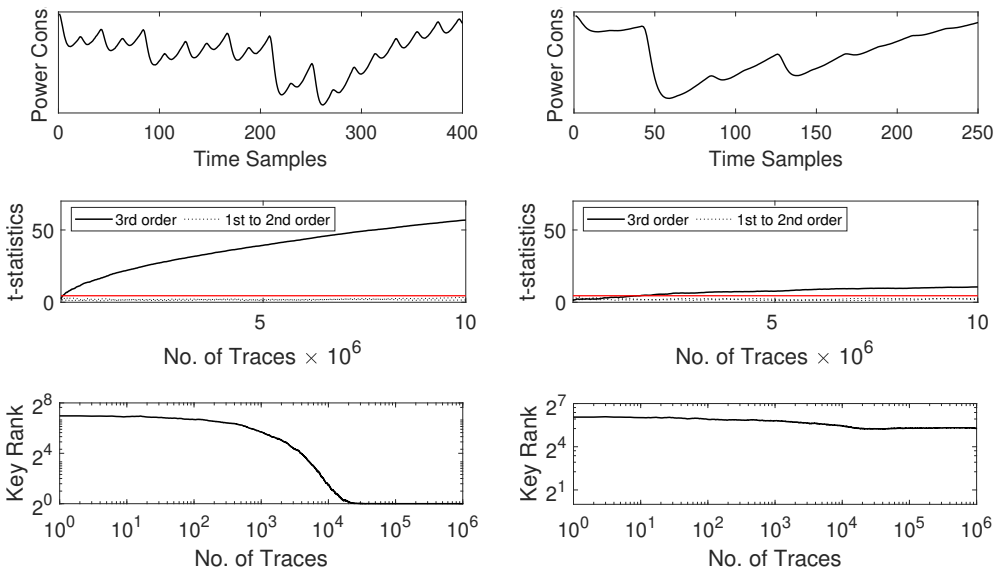


Figure 9: Comparison of second-order masked S-boxes (dynamic power). Top to bottom: sample traces, TVLA results, profiled attack results. Left to right: AES, `AES-prime`.

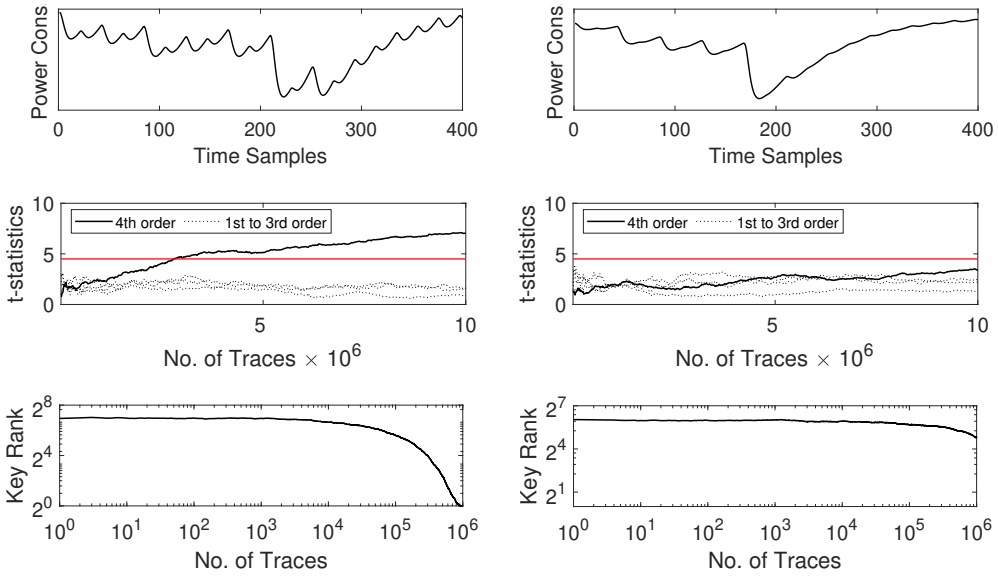


Figure 10: Comparison of third-order masked S-boxes (dynamic power). Top to bottom: sample traces, TVLA results, profiled attack results. Left to right: AES, `AES-prime`.

consumption with almost arbitrary precision. If control over the operating environment is given, the temperature and voltage levels can be influenced to increase the leakage currents exponentially (up to a certain point) in order to measure them even more precisely [Moo19]. As detailed before we have assumed control over the clock and the supply voltage of the device under test in order to obtain the experimental results presented in the following. We stop the computation during the targeted clock cycles and measure the static power consumption of the device in this idle state, while powering the target FPGA at an over-voltage of 33.3%. Afterwards the traces are post-processed using a moving average

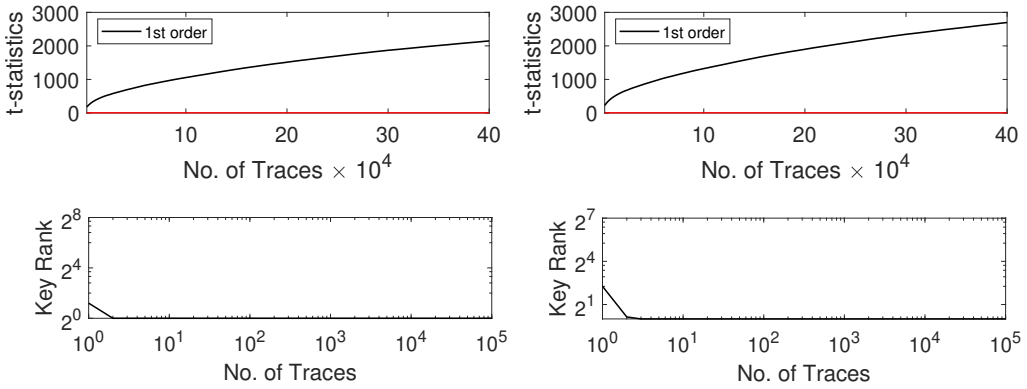


Figure 11: Comparison of unmasked S-boxes (static power). Top to bottom: TVLA results, profiled attack results. Left to right: AES, `AES-prime`.

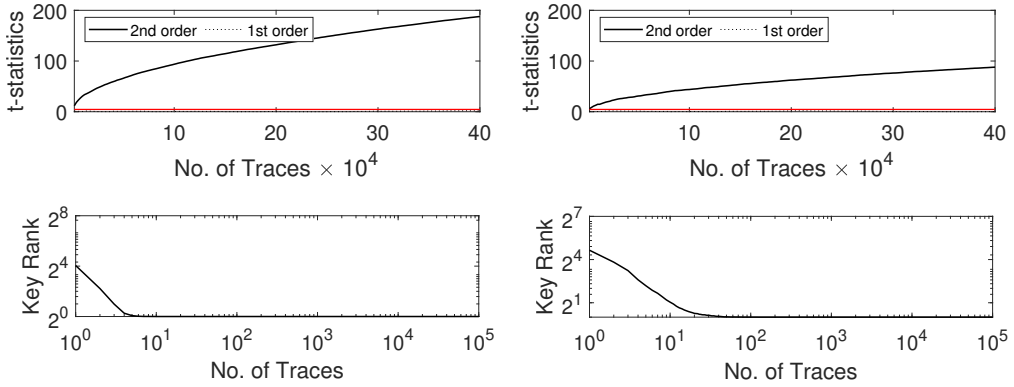


Figure 12: Comparison of first-order masked S-boxes (static power). Top to bottom: TVLA results, profiled attack results. Left to right: AES, `AES-prime`.

filter. The attack results presented in the following are obtained by selecting 100 000 traces for random inputs from the fixed-vs-random measurement set for the profiling, and then similar to the dynamic power results performing a pooled template attack.

Results. Figure 11 shows the comparison of the unmasked S-boxes, while Figures 12, 13 and 14 illustrate the corresponding results for the 2-, 3- and 4-share versions, respectively. Clearly, compared to the dynamic power results the data complexity of leakage detection and attacks is significantly reduced (despite the averaging with repeated randomness in case of the dynamic power analysis), confirming the low noise characteristic of our experimental data. The difference between the complexity of attacks on AES and `AES-prime` S-boxes reaches again 1-2 orders of magnitude. Additionally, the experiments show that the 3-share prime S-box delivers a similar practical security level as the 4-share binary one.

These experimental results once more confirm the security gains of masking in prime-order fields over masking in $\mathbb{F}_2$ for the selected (encryption and masking) algorithms. We summarize the complexities of all performed attacks in Table 10. For $d = 3$, the attacks on prime-field AES require about 30 and 75 times more traces in dynamic power and static power attacks, respectively. This is indeed a large security increase. Furthermore, the factor of security gain increases with the number of shares in our experiments.

Together with the observation that the masked implementations using our novel gadgets

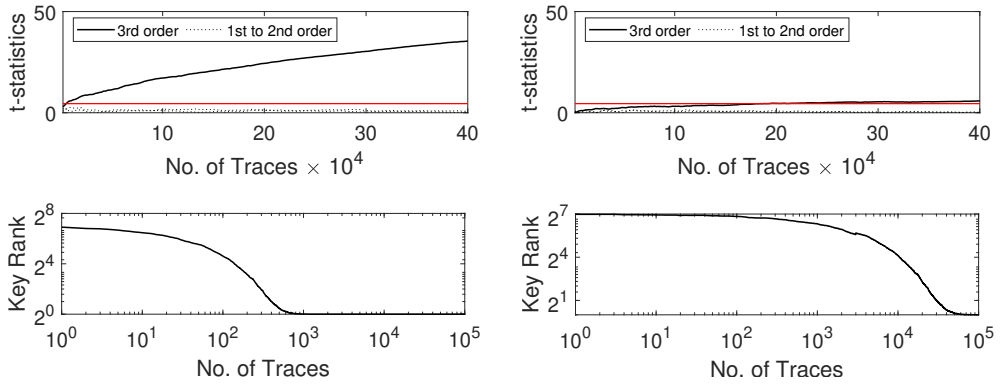


Figure 13: Comparison of second-order masked S-boxes (static power). Top to bottom: TVLA results, profiled attack results. Left to right: AES, `AES-prime`.

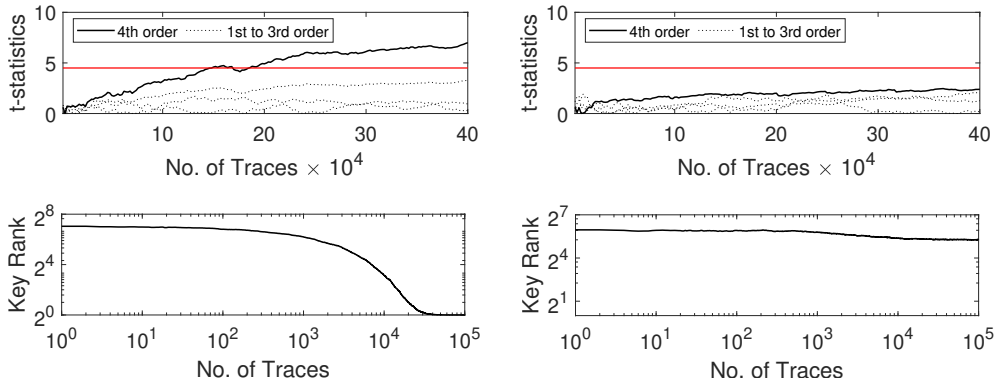


Figure 14: Comparison of third-order masked S-boxes (static power). Top to bottom: TVLA results, profiled attack results. Left to right: AES, `AES-prime`.

and constructions are only slightly more expensive for the same security order, we are confident that prime-field masking is indeed a promising research direction.

Table 10: Data complexity of key recovery of all attacks (to reach rank 1).

Cipher	Shares d	Dynamic Power		Static Power	
		Absolute	Factor	Absolute	Factor
AES	1	2		2	
	2	385		10	
	3	35 084		1 186	
	4	944 390		59 281	
`AES-prime`	1	3	1.50	3	1.50
	2	2 992	7.77	123	12.30
	3	$> 1\,000\,000$	> 28.50	88 921	74.98
	4	$> 1\,000\,000$	-	$> 100\,000$	-

6 Conclusion and Open Problems

Our results provide a first study of the side-channel security vs. performance tradeoff that prime masking provides in low-noise contexts. We compare this tradeoff for a complete `AES-prime` algorithm using our novel and optimized squaring gadgets, with the one obtained for the standard AES implemented with state-of-the-art gadgets offering similar security guarantees. In particular, for $d = 3$ shares we achieve security gains for prime masking corresponding to an increase of the attack complexity by a factor of up to 30 (w.r.t. dynamic power attacks) and 75 (w.r.t. static power attacks), while spending a factor between 1 and 2 in terms of area and between 1 and 7 in terms of critical path delay (5-7 for ASIC, 1-3 for FPGA). In our concrete example, the number of cycles per S-box execution is reduced, too. The positive trends we put forward are especially promising since they exhibit benefits that grow with the target security level and the `AES-prime` is likely not the best cipher for prime-field masking. For example, while squaring operations have linear performance overheads when protected with a Boolean masking scheme, they are non-linear and therefore have quadratic performance overheads in the prime-field setting. Given the stronger security that our analyses put forward, it suggests the design of new cipher structures that can take better advantage of prime-field masking as an interesting scope for further investigations. Design techniques developed for advanced applications (multiparty computation, fully homomorphic encryption, zero knowledge proofs) could be useful for this purpose, despite such ciphers generally target larger primes than our embedded applications can tolerate [GRR+16, AGR+16, GKR+21]. An even more promising approach would be to leverage the non-linearity of the square operation and its efficient masked version (compared to a multiplication) to design ciphers using such squares as only source of non-linearity – therefore turning the disadvantage of non-linear squaring operations into an asset, since our optimized squarings have lower computational complexity than secure multiplications. The latter also raises the question of which (possibly leakage-resistant) mode of operation to use in order to best take advantage of prime ciphers. For example, the need of an inverse or the possibility to leverage a leveled approach where an unprotected and a masked implementation are used in combination both have an impact of the best design approaches [BBC+20]. We hope our study gives motivation to investigate these questions and believe it shows the strong potential of prime-field masking for improved side-channel security at limited implementation cost.

We provide the source codes of our prime-field masked hardware implementations in the following repository: https://github.com/uclcrypto/prime_field_masking_hardware

Acknowledgments

François-Xavier Standaert is senior research associate of the Belgian Fund for Scientific Research (F.R.S.-FNRS). Gaëtan Cassiers was a research fellow of the F.R.S.-FNRS for a part of this work. This work has been funded by the European Union through the ERC Consolidator Grant SWORD (project number 724725) and by the Walloon Region through the Win2Wal project PIRATE (convention number 1910082).

References

[AGR+16] Martin R. Albrecht, Lorenzo Grassi, Christian Rechberger, Arnab Roy, and Tyge Tiessen. Mimc: Efficient encryption and cryptographic hashing with minimal multiplicative complexity. In Jung Hee Cheon and Tsuyoshi Takagi, editors, *Advances in Cryptology - ASIACRYPT 2016 - 22nd International*

Conference on the Theory and Application of Cryptology and Information Security, Hanoi, Vietnam, December 4-8, 2016, Proceedings, Part I, volume 10031 of *Lecture Notes in Computer Science*, pages 191–219, 2016.

[AGST09] Massimo Alioto, Luca Giancane, Giuseppe Scotti, and Alessandro Trifiletti. Leakage power analysis attacks: Theoretical analysis and impact of variations. In *16th IEEE International Conference on Electronics, Circuits, and Systems, ICECS 2009, Yasmine Hammamet, Tunisia, 13-19 December, 2009*, pages 85–88. IEEE, 2009.

[BBC+20] Davide Bellizia, Olivier Bronchain, Gaëtan Cassiers, Vincent Grosso, Chun Guo, Charles Momin, Olivier Pereira, Thomas Peters, and François-Xavier Standaert. Mode-level vs. implementation-level physical security in symmetric cryptography - A practical guide through the leakage-resistance jungle. In *CRYPTO (1)*, volume 12170 of *Lecture Notes in Computer Science*, pages 369–400. Springer, 2020.

[BBD+16] Gilles Barthe, Sonia Belaïd, François Dupressoir, Pierre-Alain Fouque, Benjamin Grégoire, Pierre-Yves Strub, and Rébecca Zucchini. Strong non-interference and type-directed higher-order masking. In Edgar R. Weippl, Stefan Katzenbeisser, Christopher Kruegel, Andrew C. Myers, and Shai Halevi, editors, *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security, Vienna, Austria, October 24-28, 2016*, pages 116–129. ACM, 2016.

[BBOS20] Davide Bellizia, Simone Bongiovanni, Mauro Olivieri, and Giuseppe Scotti. SC-DDPL: A novel standard-cell based approach for counteracting power analysis attacks in the presence of unbalanced routing. *IEEE Trans. Circuits Syst. I Regul. Pap.*, 67-I(7):2317–2330, 2020.

[BCPZ16] Alberto Battistello, Jean-Sébastien Coron, Emmanuel Prouff, and Rina Zeitoun. Horizontal side-channel attacks and countermeasures on the ISW masking scheme. 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 23–39. Springer, 2016.

[BCS+17] Davide Bellizia, Danilo Cellucci, Valerio Di Stefano, Giuseppe Scotti, and Alessandro Trifiletti. Novel measurements setup for attacks exploiting static power using DC pico-ammeter. In *2017 European Conference on Circuit Theory and Design, ECCTD 2017, Catania, Italy, September 4-6, 2017*, pages 1–4. IEEE, 2017.

[BDST17] Davide Bellizia, Milena Djukanovic, Giuseppe Scotti, and Alessandro Trifiletti. Template attacks exploiting static power and application to CMOS lightweight crypto-hardware. *Int. J. Circuit Theory Appl.*, 45(2):229–241, 2017.

[BFG14] Sonia Belaïd, Pierre-Alain Fouque, and Benoît Gérard. Side-channel analysis of multiplications in GF(2128) - application to AES-GCM. 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 306–325. Springer, 2014.

[BMD+20] Begül Bilgin, Lauren De Meyer, Sébastien Duval, Itamar Levi, and François-Xavier Standaert. Low AND depth and efficient inverses: a guide on s-boxes for low-latency masking. *IACR Trans. Symmetric Cryptol.*, 2020(1):144–184, 2020.

[BP12] Joan Boyar and René Peralta. A small depth-16 circuit for the AES s-box. In *SEC*, volume 376 of *IFIP Advances in Information and Communication Technology*, pages 287–298. Springer, 2012.

[BS21] Olivier Bronchain and François-Xavier Standaert. Breaking masked implementations with many shares on 32-bit software platforms or when the security order does not matter. *IACR Trans. Cryptogr. Hardw. Embed. Syst.*, 2021(3):202–234, 2021.

[BSS21] Davide Bellizia, Riccardo Della Sala, and Giuseppe Scotti. SC-DDPL as a countermeasure against static power side-channel attacks. *Cryptogr.*, 5(3):16, 2021.

[BST18] Davide Bellizia, Giuseppe Scotti, and Alessandro Trifiletti. TEL logic style as a countermeasure against side-channel attacks: Secure cells library in 65nm CMOS and experimental results. *IEEE Trans. Circuits Syst. I Regul. Pap.*, 65-I(11):3874–3884, 2018.

[CGLS21] Gaëtan Cassiers, Benjamin Grégoire, Itamar Levi, and François-Xavier Standaert. Hardware private circuits: From trivial composition to full verification. *IEEE Trans. Computers*, 70(10):1677–1690, 2021.

[CHK+21] Jihoon Cho, Jincheol Ha, Seongkwang Kim, ByeongHak Lee, Joohee Lee, Jooyoung Lee, Dukjae Moon, and Hyojin Yoon. Transciphering framework for approximate homomorphic encryption. In Mehdi Tibouchi and Huaxiong Wang, editors, *Advances in Cryptology - ASIACRYPT 2021 - 27th International Conference on the Theory and Application of Cryptology and Information Security, Singapore, December 6-10, 2021, Proceedings, Part III*, volume 13092 of *Lecture Notes in Computer Science*, pages 640–669. Springer, 2021.

[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.

[CS20] Gaëtan Cassiers and François-Xavier Standaert. Trivially and efficiently composing masked gadgets with probe isolating non-interference. *IEEE Trans. Inf. Forensics Secur.*, 15:2542–2555, 2020.

[CS21] Gaëtan Cassiers and François-Xavier Standaert. Provably secure hardware masking in the transition- and glitch-robust probing model: Better safe than sorry. *IACR Trans. Cryptogr. Hardw. Embed. Syst.*, 2021(2):136–158, 2021.

[DDF14] Alexandre Duc, Stefan Dziembowski, and Sebastian Faust. Unifying leakage models: From probing attacks to noisy leakage. In Phong Q. Nguyen and Elisabeth Oswald, editors, *Advances in Cryptology - EUROCRYPT 2014 - 33rd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Copenhagen, Denmark, May 11-15, 2014. Proceedings*, volume 8441 of *Lecture Notes in Computer Science*, pages 423–440. Springer, 2014.

[DEG+18] Christoph Dobraunig, Maria Eichlseder, Lorenzo Grassi, Virginie Lallemand, Gregor Leander, Eik List, Florian Mendel, and Christian Rechberger. Rasta: A cipher with low anddepth and few ands per bit. In Hovav Shacham and Alexandra Boldyreva, editors, *Advances in Cryptology - CRYPTO 2018 - 38th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 19-23, 2018, Proceedings, Part I*, volume 10991 of *Lecture Notes in Computer Science*, pages 662–692. Springer, 2018.

[DFS15] Alexandre Duc, Sebastian Faust, and François-Xavier Standaert. Making masking security proofs concrete - or how to evaluate the security of any leaking device. In Elisabeth Oswald and Marc Fischlin, editors, *Advances in Cryptology - EUROCRYPT 2015 - 34th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Sofia, Bulgaria, April 26-30, 2015, Proceedings, Part I*, volume 9056 of *Lecture Notes in Computer Science*, pages 401–429. Springer, 2015.

[DFS16] Stefan Dziembowski, Sebastian Faust, and Maciej Skórski. Optimal amplification of noisy leakages. In Eyal Kushilevitz and Tal Malkin, editors, *Theory of Cryptography - 13th International Conference, TCC 2016-A, Tel Aviv, Israel, January 10-13, 2016, Proceedings, Part II*, volume 9563 of *Lecture Notes in Computer Science*, pages 291–318. Springer, 2016.

[DGGK21] Christoph Dobraunig, Lorenzo Grassi, Anna Guinet, and Daniël Kuijsters. Ciminion: Symmetric encryption based on toffoli-gates over large finite fields. In Anne Canteaut and François-Xavier Standaert, editors, *Advances in Cryptology - EUROCRYPT 2021 - 40th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Zagreb, Croatia, October 17-21, 2021, Proceedings, Part II*, volume 12697 of *Lecture Notes in Computer Science*, pages 3–34. Springer, 2021.

[DMMS21] Sébastien Duval, Pierrick Méaux, Charles Momin, and François-Xavier Standaert. Exploring crypto-physical dark matter and learning with physical rounding towards secure and efficient fresh re-keying. *IACR Trans. Cryptogr. Hardw. Embed. Syst.*, 2021(1):373–401, 2021.

[FGP+18] Sebastian Faust, Vincent Grosso, Santos Merino Del Pozo, Clara Paglialonga, and François-Xavier Standaert. Composable masking schemes in the presence of physical defaults & the robust probing model. *IACR Trans. Cryptogr. Hardw. Embed. Syst.*, 2018(3):89–120, 2018.

[GKR+21] Lorenzo Grassi, Dmitry Khovratovich, Christian Rechberger, Arnab Roy, and Markus Schofnegger. Poseidon: A new hash function for zero-knowledge proof systems. In Michael Bailey and Rachel Greenstadt, editors, *30th USENIX Security Symposium, USENIX Security 2021, August 11-13, 2021*, pages 519–535. USENIX Association, 2021.

[GM18] Hannes Groß and Stefan Mangard. A unified masking approach. *J. Cryptogr. Eng.*, 8(2):109–124, 2018.

[GMK16] Hannes Groß, Stefan Mangard, and Thomas Korak. Domain-oriented masking: Compact masked hardware implementations with arbitrary protection order. In Begül Bilgin, Svetla Nikova, and Vincent Rijmen, editors, *Proceedings of the ACM Workshop on Theory of Implementation Security, TIS@CCS 2016 Vienna, Austria, October, 2016*, page 3. ACM, 2016.

[GMK17] Hannes Groß, Stefan Mangard, and Thomas Korak. An efficient side-channel protected AES implementation with arbitrary protection order. In Helena Handschuh, editor, *Topics in Cryptology - CT-RSA 2017 - The Cryptographers' Track at the RSA Conference 2017, San Francisco, CA, USA, February 14-17, 2017, Proceedings*, volume 10159 of *Lecture Notes in Computer Science*, pages 95–112. Springer, 2017.

[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.

[GPS14] Vincent Grosso, Emmanuel Prouff, and François-Xavier Standaert. Efficient masked s-boxes processing - A step forward -. In David Pointcheval and Damien Vergnaud, editors, *Progress in Cryptology - AFRICACRYPT 2014 - 7th International Conference on Cryptology in Africa, Marrakesh, Morocco, May 28-30, 2014. Proceedings*, volume 8469 of *Lecture Notes in Computer Science*, pages 251–266. Springer, 2014.

[GR17] Dahmun Goudarzi and Matthieu Rivain. How fast can higher-order masking be in software? In Jean-Sébastien Coron and Jesper Buus Nielsen, editors, *Advances in Cryptology - EUROCRYPT 2017 - 36th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Paris, France, April 30 - May 4, 2017, Proceedings, Part I*, volume 10210 of *Lecture Notes in Computer Science*, pages 567–597, 2017.

[GRR+16] Lorenzo Grassi, Christian Rechberger, Dragos Rotaru, Peter Scholl, and Nigel P. Smart. Mpc-friendly symmetric key primitives. In Edgar R. Weippl, Stefan Katzenbeisser, Christopher Kruegel, Andrew C. Myers, and Shai Halevi, editors, *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security, Vienna, Austria, October 24-28, 2016*, pages 430–443. ACM, 2016.

[GSST07] Jacopo Giorgetti, Giuseppe Scotti, Andrea Simonetti, and Alessandro Trifiletti. Analysis of data dependence of leakage current in CMOS cryptographic hardware. In Hai Zhou, Enrico Macii, Zhiyuan Yan, and Yehia Massoud, editors, *Proceedings of the 17th ACM Great Lakes Symposium on VLSI 2007, Stresa, Lago Maggiore, Italy, March 11-13, 2007*, pages 78–83. ACM, 2007.

[ISW03] Yuval Ishai, Amit Sahai, and David A. Wagner. Private circuits: Securing hardware against probing attacks. In Dan Boneh, editor, *Advances in Cryptology - CRYPTO 2003, 23rd Annual International Cryptology Conference, Santa Barbara, California, USA, August 17-21, 2003, Proceedings*, volume 2729 of *Lecture Notes in Computer Science*, pages 463–481. Springer, 2003.

[JS17] Anthony Journault and François-Xavier Standaert. Very high order masking: Efficient implementation and security evaluation. In Wieland Fischer and Naofumi Homma, editors, *Cryptographic Hardware and Embedded Systems - CHES 2017 - 19th International Conference, Taipei, Taiwan, September 25-28, 2017, Proceedings*, volume 10529 of *Lecture Notes in Computer Science*, pages 623–643. Springer, 2017.

[KMM19] Naghmeh Karimi, Thorben Moos, and Amir Moradi. Exploring the effect of device aging on static power analysis attacks. *IACR Trans. Cryptogr. Hardw. Embed. Syst.*, 2019(3):233–256, 2019.

[KMMS22] David Knichel, Amir Moradi, Nicolai Müller, and Pascal Sasdrich. Automated generation of masked hardware. *IACR Trans. Cryptogr. Hardw. Embed. Syst.*, 2022(1):589–629, 2022.

[KSM22] David Knichel, Pascal Sasdrich, and Amir Moradi. Generic hardware private circuits towards automated generation of composable secure gadgets. *IACR Trans. Cryptogr. Hardw. Embed. Syst.*, 2022(1):323–344, 2022.

[MCS22] Charles Momin, Gaëtan Cassiers, and François-Xavier Standaert. Handcrafting: Improving automated masking in hardware with manual optimizations. In Josep Balasch and Colin O'Flynn, editors, *Constructive Side-Channel Analysis and Secure Design - 13th International Workshop, COSADE 2022, Leuven, Belgium, April 11-12, 2022, Proceedings*, volume 13211 of *Lecture Notes in Computer Science*, pages 257–275. Springer, 2022.

[MKSM22] Nicolai Müller, David Knichel, Pascal Sasdrich, and Amir Moradi. Transitional leakage in theory and practice unveiling security flaws in masked circuits. *IACR Trans. Cryptogr. Hardw. Embed. Syst.*, 2022(2):266–288, 2022.

[MM21] Thorben Moos and Amir Moradi. Countermeasures against static power attacks - comparing exhaustive logic balancing and other protection schemes in 28 nm CMOS -. *IACR Trans. Cryptogr. Hardw. Embed. Syst.*, 2021(3):780–805, 2021.

[MMMS22] Loïc Masure, Pierrick Méaux, Thorben Moos, and François-Xavier Standaert. Effective and efficient masking with low noise using small-mersenne-prime ciphers. *IACR Cryptol. ePrint Arch.*, page 863, 2022.

[MMR17] Thorben Moos, Amir Moradi, and Bastian Richter. Static power side-channel analysis of a threshold implementation prototype chip. In David Atienza and Giorgio Di Natale, editors, *Design, Automation & Test in Europe Conference & Exhibition, DATE 2017, Lausanne, Switzerland, March 27-31, 2017*, pages 1324–1329. IEEE, 2017.

[MMR20] Thorben Moos, Amir Moradi, and Bastian Richter. Static power side-channel analysis - an investigation of measurement factors. *IEEE Trans. Very Large Scale Integr. Syst.*, 28(2):376–389, 2020.

[MMSS19] Thorben Moos, Amir Moradi, Tobias Schneider, and François-Xavier Standaert. Glitch-resistant masking revisited or why proofs in the robust probing model are needed. *IACR Trans. Cryptogr. Hardw. Embed. Syst.*, 2019(2):256–292, 2019.

[Moo19] Thorben Moos. Static power SCA of sub-100 nm CMOS asics and the insecurity of masking schemes in low-noise environments. *IACR Trans. Cryptogr. Hardw. Embed. Syst.*, 2019(3):202–232, 2019.

[Moo20] Thorben Moos. Unrolled cryptography on silicon A physical security analysis. *IACR Trans. Cryptogr. Hardw. Embed. Syst.*, 2020(4):416–442, 2020.

[Mor14] Amir Moradi. Side-channel leakage through static power - should we care about in practice? In Lejla Batina and Matthew Robshaw, editors, *Cryptographic Hardware and Embedded Systems - CHES 2014 - 16th International Workshop, Busan, South Korea, September 23-26, 2014. Proceedings*, volume 8731 of *Lecture Notes in Computer Science*, pages 562–579. Springer, 2014.

[MSGR10] Marcel Medwed, François-Xavier Standaert, Johann Großschädl, and Francesco Regazzoni. Fresh re-keying: Security against side-channel and fault attacks for low-cost devices. In Daniel J. Bernstein and Tanja Lange, editors, *Progress in Cryptology - AFRICACRYPT 2010, Third International Conference on Cryptology in Africa, Stellenbosch, South Africa, May 3-6, 2010. Proceedings*, volume 6055 of *Lecture Notes in Computer Science*, pages 279–296. Springer, 2010.

[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.

[NRS11] Svetla Nikova, Vincent Rijmen, and Martin Schläffer. Secure hardware implementation of nonlinear functions in the presence of glitches. *J. Cryptol.*, 24(2):292–321, 2011.

[PR13] Emmanuel Prouff and Matthieu Rivain. Masking against side-channel attacks: A formal security proof. In Thomas Johansson and Phong Q. Nguyen, editors, *Advances in Cryptology - EUROCRYPT 2013, 32nd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Athens, Greece, May 26-30, 2013. Proceedings*, volume 7881 of *Lecture Notes in Computer Science*, pages 142–159. Springer, 2013.

[PSKM15] Santos Merino Del Pozo, François-Xavier Standaert, Dina Kamel, and Amir Moradi. Side-channel attacks from static power: when should we care? In Wolfgang Nebel and David Atienza, editors, *Proceedings of the 2015 Design, Automation & Test in Europe Conference & Exhibition, DATE 2015, Grenoble, France, March 9-13, 2015*, pages 145–150. ACM, 2015.

[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.

[SA08] François-Xavier Standaert and Cédric Archambeau. Using subspace-based template attacks to compare and combine power and electromagnetic information leakages. In *CHES*, volume 5154 of *Lecture Notes in Computer Science*, pages 411–425. Springer, 2008.

[sak] Side-channel AttacK User Reference Architecture. http://satoh.cs.uec.ac.jp/SAKURA/index.html.

[SM16] Tobias Schneider and Amir Moradi. Leakage assessment methodology - extended version. *J. Cryptogr. Eng.*, 6(2):85–99, 2016.

[Sta18] François-Xavier Standaert. How (not) to use welch's t-test in side-channel security evaluations. In Begül Bilgin and Jean-Bernard Fischer, editors, *Smart Card Research and Advanced Applications, 17th International Conference, CARDIS 2018, Montpellier, France, November 12-14, 2018, Revised Selected Papers*, volume 11389 of *Lecture Notes in Computer Science*, pages 65–79. Springer, 2018.

[SVO+10] François-Xavier Standaert, Nicolas Veyrat-Charvillon, Elisabeth Oswald, Benedikt Gierlichs, Marcel Medwed, Markus Kasper, and Stefan Mangard. The world is not enough: Another look on second-order DPA. In Masayuki Abe, editor, *Advances in Cryptology - ASIACRYPT 2010 - 16th International Conference on the Theory and Application of Cryptology and Information Security, Singapore, December 5-9, 2010. Proceedings*, volume 6477 of *Lecture Notes in Computer Science*, pages 112–129. Springer, 2010.

[VGS14] Nicolas Veyrat-Charvillon, Benoît Gérard, and François-Xavier Standaert. Soft analytical side-channel attacks. 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 I*, volume 8873 of *Lecture Notes in Computer Science*, pages 282–296. Springer, 2014.

[XH19] Jiming Xu and Howard M. Heys. Kernel-based template attacks of cryptographic circuits using static power. *Integr.*, 66:67–79, 2019.

A Cost and Performance on Spartan-6 FPGA

In this section, we report the cost and performance numbers of AES and `AES-prime` implementations, similar to Section 4, on a Xilinx Spartan-6 FPGA (XC6SLX75-2CSG484C) which is the exact device mounted on the Sakura-G board used for the experimental case study in Section 5. The synthesis and implementation are performed using the admittedly quite old device-compatible Xilinx design tool, namely ISE Design Suite 14.7. For the masked implementations, the global `-keep_hierarchy` attribute is set to `yes`. The resulting resource utilization and critical path delays are given in Table 11. The results for unprotected full round-based implementations are listed in Table 12. However, due to the limited amount of resources on the device (in particular DSP slices), masked round-based implementations with 20 parallel masked S-boxes are excluded here.

Table 11: Cost and performance figures for glitch-robust PINI AES and `AES-prime` S-boxes synthesized using Xilinx ISE for a Spartan-6 FPGA.

S-box	Shares d	FFs	LUTs	Slic.	DSPs	Crit. Path [ns]	Reg. Stag.
AES	1	0	43	15	0	8.18	0
	2	587	591	998	0	8.15	6
	3	1275	1457	2300	0	9.43	6
	4	2226	2310	3827	0	9.76	6
`AES-prime`	1	0	103	32	1	22.68	0
	2	140	394	180	8	22.48	3
	3	375	804	397	21	26.62	4
	4	1048	1683	907	34	19.19	5
`AES-prime`*	1	0	175	51	0	23.49	0
	2	140	903	282	0	21.56	3
	3	375	2151	676	0	24.67	4
	4	1048	3880	1333	0	17.52	5

*Use of DSP slices prohibited

Table 12: Cost and performance figures for unprotected round-based implementations of AES and `AES-prime` synthesized using Xilinx ISE for a Spartan-6 FPGA.

Cipher	FFs	LUTs	Slices	DSPs	Crit. Path [ns]	Lat. [cycles]
AES	269	1524	448	0	12.96	10
`AES-prime`	248	3533	1151	20	32.51	14
`AES-prime`*	236	5068	1347	0	38.95	14

*Use of DSP slices prohibited

B Cost and Performance Comparison between Optimized and Trivial Composition

B.1 ASIC

Table 13: Cost and performance figures for glitch-robust PINI `AES-prime` S-boxes for $\neq$ composition strategies synthesized for minimum area using a 65 nm ASIC library.

S-box Comp.	Shares d	Area [GE]	Crit. Path [ns]	Reg. Stages	Randomn. [bits]
Optimized	2	4682.75	9.32	3	35
	3	11467.50	10.54	4	91
	4	22335.00	7.66	5	210
Trivial (HPC1)	2	4807.25	9.32	3	42
	3	11514.50	10.54	3	105
	4	23236.00	7.84	5	238

Table 14: Cost and performance figures for glitch-robust PINI `AES-prime` S-boxes for $\neq$ composition strategies synthesized for max. frequency using a 65 nm ASIC library.

S-box Comp.	Shares d	Area [GE]	Crit. Path [ns]	Reg. Stages	Randomn. [bits]
Optimized	2	6449.50	1.93	3	35
	3	16493.75	2.21	4	91
	4	31992.75	1.50	5	210
Trivial (HPC1)	2	6977.25	1.86	3	42
	3	16533.25	2.19	3	105
	4	32640.00	1.51	5	238

B.2 FPGA

Table 15: Cost & performance figures for glitch-robust PINI `AES-prime` S-boxes for $\neq$ composition strategies synthesized for min. area using a Xiling UltraScale+ FPGA.

S-box Comp.	Shares d	FFs	LUTs	Slic.	DSPs	Crit. Path [ns]	Reg. Stag.
Optimized	2	133	379	87	10	9.15	3
	3	364	927	219	21	10.12	4
	4	1022	1885	444	36	9.71	5
Optimized*	2	133	963	184	0	10.00	3
	3	364	2271	432	0	11.92	4
	4	1022	4133	803	0	10.41	5
Trivial (HPC1)	2	133	409	104	10	11.85	3
	3	322	989	220	21	13.71	3
	4	1106	2005	481	36	13.32	5
Trivial (HPC1)*	2	133	993	179	0	11.68	3
	3	322	2333	435	0	14.20	3
	4	1106	4252	843	0	16.53	5

*Use of DSP slices prohibited

Table 16: Cost & performance figures for glitch-robust PINI `AES-prime` S-boxes for $\neq$ composition strategies synthesized for max. frequency using a Xiling UltraScale+ FPGA.

S-box Comp.	Shares d	FFs	LUTs	Slic.	DSPs	Crit. Path [ns]	Reg. Stag.
Optimized	2	133	495	123	10	5.95	3
	3	364	1207	246	21	6.69	4
	4	1022	2462	487	36	7.02	5
Optimized*	2	133	1095	202	0	5.55	3
	3	364	2589	464	0	6.26	4
	4	1022	4774	834	0	7.13	5
Trivial (HPC1)	2	133	534	120	10	9.89	3
	3	350	1281	289	21	11.34	3
	4	1106	2630	547	36	10.67	5
Trivial (HPC1)*	2	133	1132	204	0	9.41	3
	3	322	2667	458	0	11.17	3
	4	1106	4960	912	0	11.42	5

*Use of DSP slices prohibited