軽量暗号 Ascon の実装性能に関する調査及び評価

電気通信大学 大学院情報理工学研究科

崎山 一男

2023 年 9 月

エグゼクティブサマリー

米国 National Institute of Standards and Technology (NIST) は，軽量暗号 (LWC: Lightweight Cryptography) コンペティション [34] で Ascon を選定した．本報告は，公開されている Ascon-128 の認証暗号モードにおける暗号化及び復号処理を行う実装研究を中心に論文を調査し，物理攻撃への耐性を持つハードウェア及びソフトウェア実装の性能評価結果をまとめ，考察を与えたものである．理論的安全性については，藤堂の文献 [48] を参照されたい．

Ascon は，認証暗号モードとハッシュモードに対応する軽量な暗号アルゴリズムであり，実装コストと処理パフォーマンスのトレードオフの点で高い柔軟性がある．つまり，ハードウェア実装でもソフトウェア実装でも，暗号機能を効率的に実現することができるため，様々なユースケースでの利用が期待される．高い柔軟性の理由は，以下のとおりである．

- 同じラウンド処理の繰り返しで実現できる
- ラウンド処理が並列実装にも対応できる
- 同じ 5 ビットの S-box が繰り返し使われている

5 ビットのコンパクトな S-box を同時に処理することで，実装コストの増加に見合う処理パフォーマンスを得ることができる．一方で，異なる時間に S-box を再利用して処理することで，処理パフォーマンスを犠牲にして実装コストを下げることができる．つまり，ハードウェア実装においては，インタフェースや求める性能に合わせてアーキテクチャを柔軟に変更することができ，ソフトウェア実装においては CPU のワードサイズに合わせたプログラミングが可能となる．また，複数のラウンド処理をまとめて計算することで，処理パフォーマンスのさらなる向上が実現できる．

Ascon が特に優れている理由のひとつは，ほとんど全ての処理を同じラウンド処理の繰り返して実現できる点にある．暗号化処理と復号処理の違いは，ラウンド処理における入出力データのインタフェース部分のみである．この極めて規則的な処理構造のおかげで，各モードの切り替えに対するオーバーヘッドは極めて小さくてすむ．AES 暗号にも，似たような実装上の性質はあるものの，暗号処理自体のデータパスが同じである Ascon は，実装における柔軟性がさらに高いと言える．

コンペティションで，AES 暗号が選定されたのは，1997 年 9 月であり，サイドチャネル攻撃の危険性を Kocher が最初に指摘したのが 1995 年 12 月である．そのため，AES 暗号に対してアルゴリズムレベルでの物理攻撃対策が十分に考慮できる状況ではなかった [24]．乱数によるマスキングや WDDL といった，サイドチャネル攻撃対策の研究が盛んになったのは 2000 年前後である [9, 43] ．つまり，Kocher によるサイドチャネル攻撃の論文や AES 暗号の選定により物理攻撃対策への研究者の意識が高まり，暗号アルゴリズムを新規に設計する場合においては，物理攻撃対策を含めた実装性は考慮すべきひとつの要素となった．Ascon はその最初の暗号アルゴ

リズムと言える．

その後，Nikova らによって Threshold Implementation (TI) が提案されたのが，2006 年である [32, 33]．現在 TI は，ハードウェア実装とソフトウエア実装の両方で多くの研究報告がなされており，現在も改良が進んでいる．Domain Oriented Masking (DOM) [20, 19] といった，TI よりもさらに効率的な実装を目指した対策技術が提案されるなど，暗号研究者内での理解は急速に進んだ．Ascon が最初に提案されたのは，2014 年の認証暗号のコンペティション CAESAR competition [4] である．Ascon を最初に提案したころは，最新の物理攻撃対策技術の成熟期にあった．実際に，Ascon-128 が物理攻撃対策との親和性が高い実装構造となっていることは興味深い事実である．

物理攻撃耐性を評価する手法にも大きな変化があった．サイドチャネル攻撃の発見直後は，鍵復元攻撃の成否や，少ない波形数での攻撃成功を目指すケーススタディが比較的多かった．非プロファイリング型の攻撃では選択関数やリーケージモデルの研究が，プロファイリング型の攻撃の場合ではテンプレートの作成方法に関する研究が研究の中心であった．これらは，攻撃者の能力に関するものである．適切な攻撃者が実装されていない場合には鍵が復元できないため，脆弱性は見つけることができない．さらに，実験における計算量の限界により攻撃を実装できない場合にも，脆弱性はないものとされてきた．つまり，攻撃が失敗したときには，暗号実装の安全性を判断することはできない．

現在では，Test Vector Leakage Assessment (TVLA) [18] による統計的に安全性を評価する手法が主流となっている．TVLA は，鍵が実際に導出できるかどうかを試すのではなく，サイドチャネルリークによる攻撃の可能性を判断するものである．未知の攻撃手法を含め，厳密に安全性評価が行えるようになった．現在の暗号アルゴリズムの実装研究では，TI といった乱数を用いたマスキング対策で物理攻撃を実装し，その安全性評価には TVLA を用いることが主流となっている．マスキング対策における実装上の問題は，冗長化した回路やプログラムのサイズによる実装コストと，マスキングに必要となる乱数コストである．実装コストを抑えるための研究成果は多く存在しているが，暗号アルゴリズム毎に最適な冗長化や乱数コストを狙う研究と，汎用的なコスト削減に向けた設計手法の確立を目指すものとに分かれる．

本報告では，最初に Ascon に適用する物理攻撃対策技術とその安全性評価手法に関する調査を行う．次に，実際のハードウェア実装及びソフトウェア実装における物理攻撃対策に関するケーススタディを 8 件を取り上げ，それらの内容をまとめた上で考察を与える．Ascon-128 の実装性能は非常に高く，特に物理攻撃対策については暗号研究者がこれまでに培った最新の技術を搭載しやすい構造である．一方で，今後 Ascon が，IoT デバイスとして様々なプラットフォームに実装されることを想定すると，対策を含めた暗号アルゴリズム実装について，その生産性の向上が重要となる．したがって，マスキング設計ツールやその安全性検証ツールを Ascon に適用した論文の調査を含め，今後の暗号実装研究の新たな方向性についても言及する．

目次

1 はじめに 1

2 本報告書の概要 2

2.1 調査対象 2

2.2 Ascon の実装性能評価 3

3 Ascon のセキュア実装 4

3.1 Ascon のアルゴリズム 4

3.2 Ascon の軽量 Permutation 7

3.3 Ascon に対するサイドチャネル攻撃対策 10

3.3.1 TI: Threshold Implementation 10

3.3.2 DOM: Domain Oriented Masking 11

3.4 Ascon に対するサイドチャネルからの漏洩評価 13

3.4.1 CPA: Correlation Power Analysis 13

3.4.2 TA: Template Attack 13

3.4.3 TVLA: Test Vector Leakage Assessment (Welch's t-test) 14

4 Ascon の物理安全性と実装性に関するケーススタディ 15

4.1 Niels と Daemen による報告（2017.05） [42] 15

4.1.1 著者, 所属機関 15

4.1.2 概要 15

4.1.3 CPA における選択関数 15

4.1.4 攻撃の結果 16

4.1.5 まとめ 16

4.2 Groß の 学位論文（2018.06） [19] 17

4.2.1 著者, 所属機関 17

4.2.2 概要 17

4.2.3 実装結果 17

4.2.4 まとめ 18

4.3 Batina らによる報告（2022.08） [5] 19

4.3.1 著者, 所属機関 19

4.3.2 概要 19

4.3.3 攻撃対象及び評価環境 19

4.3.4 評価結果 19

4.3.5 まとめ 20
4.4 Mohajerani らによる報告（2023.06） [30] 21
4.4.1 著者, 所属機関 21
4.4.2 概要 21
4.4.3 評価対象，手法，及び結果 22
4.4.4 攻撃対象のクロックとサンプリングクロックとの同期について 24
4.4.5 対策による面積コストと処理パフォーマンスへの影響 24
4.4.6 まとめ 24
4.5 Kandi らによる報告（2023.06） [28] 25
4.5.1 著者, 所属機関 25
4.5.2 概要 25
4.5.3 実装性能評価の結果 25
4.5.4 S-Box に用いられた 3 シェア TI 26
4.5.5 ３重化による故障利用攻撃対策 28
4.5.6 まとめ 29
4.6 Gigerl らによる報告（2023.06） [17] 30
4.6.1 著者, 所属機関 30
4.6.2 概要 30
4.6.3 Coco 30
4.6.4 攻撃対象及び評価環境 31
4.6.5 評価結果 31
4.6.6 まとめ 31
4.7 Liu と Schaumont による報告（2023.06） [28] 33
4.7.1 著者, 所属機関 33
4.7.2 概要 33
4.7.3 攻撃対象及び評価環境 33
4.7.4 評価の結果 34
4.7.5 実測による妥当性の評価 34
4.7.6 まとめ 35
4.8 You らによる報告（2023.09） [45] 36
4.8.1 著者, 所属機関 36
4.8.2 概要 36
4.8.3 攻撃対象及び評価環境 36
4.8.4 まとめ 37

図目次

1 認証暗号モードにおける Ascon の暗号化プロセス 5
2 認証暗号モードにおける Ascon の復号プロセス 6
3 Ascon の S-box 8

表目次

3.1 Ascon の S-box 8

4.1 Groß による対策付き Ascon AEAD 処理の ASIC 実装の報告（UMC-90nm Low-K） 17

4.2 対策済み Ascon の FPGA 実装に対する安全性評価結果 23

4.3 対策済み Ascon のソフトウェア実装に対する安全性評価結果 23

4.4 Kandi らによる Ascon AEAD 処理の ASIC 実装（STM 130nm） 25

4.5 Ascon AEAD 処理の FPGA 実装（Kintex-7） 26

4.6 Ascon AEAD の ASIC 実装（STM 130nm） 29

4.7 Daeman らによる Ascon のソフトウェア実装の結果 [Cycles/Byte] 31

1 はじめに

モノのインターネット（IoT: Internet of Things）が社会実装され，実世界のデータをサイバー空間に取り込む IoT デバイスが新たな攻撃対象となっている．IoT デバイスは，センサ，通信モジュール，データ処理を行う CPU や専用ハードウェアで構成されている．IoT デバイスの多くは小型であり，バッテリー消費を抑えた省エネルギー実装やアンテナからの電力伝送で動作できるほどの低電力実装が求められる．軽量暗号アルゴリズムは，そのような低リソースの環境下においても安全に機能しなければならない．

米国 NIST (National Institute of Standards and Technology) は，社会のニーズを鑑みて NIST 軽量暗号 (LWC: Lightweight Cryptography) コンペティション [34] を実施し，2022 年 2 月に Ascon を選定した．Ascon はデータの秘匿性と認証性を担保する認証付き暗号（認証暗号）とハッシュ関数の機能をサポートすることができる軽量な暗号アルゴリズムである．

本報告では，今後世界中で広く使われていくであろう Ascon-128 の物理攻撃耐性を含めた実装性能に関する調査を行うものである．つまり，サイドチャネル攻撃や故障利用解析攻撃への対策技術の実装に関して，単なる実装コストや処理コストの議論ではなく，物理安全性に必要となるコストを調査する．社会基盤である IoT システムの安全性は，IoT デバイスの物理攻撃耐性で決まる．この報告では，物理攻撃研究が極めて活発な状況にある欧米中の研究機関，企業，及び大学から発表された Ascon の物理攻撃耐性に関する論文の中から，特に重要と思われるものを調査の対象とする．

2 本報告書の概要

2.1 調査対象

本報告において, 調査する対象の暗号アルゴリズムは, NIST 軽量暗号として選定された Ascon である．認証暗号モード時の暗号化及び復号処理の Ascon-128 に関する ASIC 実装，FPGA 実装及びソフトウェア実装に関して，物理安全性を中心とする性能評価を扱う文献を調査する．以下が主な情報源である．

- NIST 主催のワークショップ Lightweight Cryptography Workshop
  `https://csrc.nist.gov/Projects/lightweight-cryptography/workshops`
- IACR Transactions on Cryptographic Hardware and Embedded Systems (TCHES)
  `https://tches.iacr.org`
- IACR Cryptology ePrint Archive
  `https://eprint.iacr.org`

他にも，Springer 社の LNCS (Lecture Notes in Computer Science) [46] や米国電気電子学会 IEEE の会議プロシーディングスと学術論文誌 [50] を調査する．結果として，本報告には，NIST は，2023 年 6 月 21, 22 日に開催された第 6 回軽量暗号ワークショップ（バーチャル）で発表された研究成果が多く含まれている．その理由は，これまでの NIST 軽量暗号コンペティションの選考中においては，理論的安全性，実装コスト，及び処理パフォーマンスによる議論が中心であった一方で，第 6 回軽量暗号ワークショップでは，これまでの軽量暗号の標準化に向けた技術的課題の議論，つまり社会実装に向けた Ascon の物理攻撃耐性に関する発表が多く見られたためである．

本報告では CRYPTREC 外部評価報告書 [41] での実装性能調査と同様，Ascon の実装形態を以下のカテゴリに分けて調査を行う．

- ハードウェアアクセラレータ
  - コプロセッサ（FPGA 実装）
  - コプロセッサ（ASIC 実装）
- 命令拡張
- ソフトウェア実装

Ascon の命令拡張実装は，CHES 2023 に採択された論文 [8] で実装性能の報告がなされているが，物理攻撃対策がなされていないため，本報告の対象から外す．

2.2 Ascon の実装性能評価

Ascon-128 の物理攻撃対策のある実装（以下，セキュア実装と呼ぶ）における実装コスト，処理パフォーマンス，及び物理攻撃耐性について，公開されている論文の実験結果をまとめ，考察を与える．従来の暗号アルゴリズムの物理攻撃対策では，通常の設計フローで対応できない部分については，その都度，必要となる対策回路の設計や検証を人手で行うことが多かった．しかし，乱数を用いたマスキングによる物理攻撃対策は複雑であり，正しく対策技術を実装するためには設計と検証の自動化の必要がある．設計及び検証の自動化により，多少の実装コストや処理パフォーマンスのオーバヘッドが生じる可能性はあるが，人手による安全性上のバグを防ぐためにはこういったツールの活用は不可欠と言える．したがって，調査した論文に設計手法やツールが書かれているものについては，性能評価における重要な要素として付記する．

3 Ascon のセキュア実装

3.1 Ascon のアルゴリズム

Ascon の設計者（提出者）は，IAIK, Graz University of Technology に所属している次の 4 名の研究者である．

- Christoph Dobraunig
- Maria Eichlseder
- Florian Mendel
- Martin Schläffer

本方式に関する概要と仕様は以下の URL から参照できる．

- Web サイト：`https://ascon.iaik.tugraz.at`
- 仕様:`https://csrc.nist.gov/CSRC/media/Projects/lightweight-cryptography/documents/finalist-round/updated-spec-doc/ascon-spec-final.pdf`

Ascon の認証暗号モードにおける処理は，大きく次の 3 つのステップから成る．以下で述べるワードは，64 ビットである．また，入力されるデータ長により，ステップ (2) の処理時間は異なる．ただし，Initialization と Finalization の処理時間は，入力されるデータに関係なく一定である．

ステップ (1): **Initialization**: ステートを鍵 K, ノンス N, 及び初期値 IV を用いて初期化する．

ステップ (2): **Iteration（データ処理部）**: Associated Data (AD) を分割し，A_i を入力とする処理を行う．その後，暗号化においては平文を決められたサイズに分割した P_i の処理を行い暗号文 C を出力する．復号においては分割された暗号分 C_i を入力とする処理を行い，平文 P を出力する．

ステップ (3): **Finalization**: 再び鍵を入力とする処理を行い，タグ T を出力する．

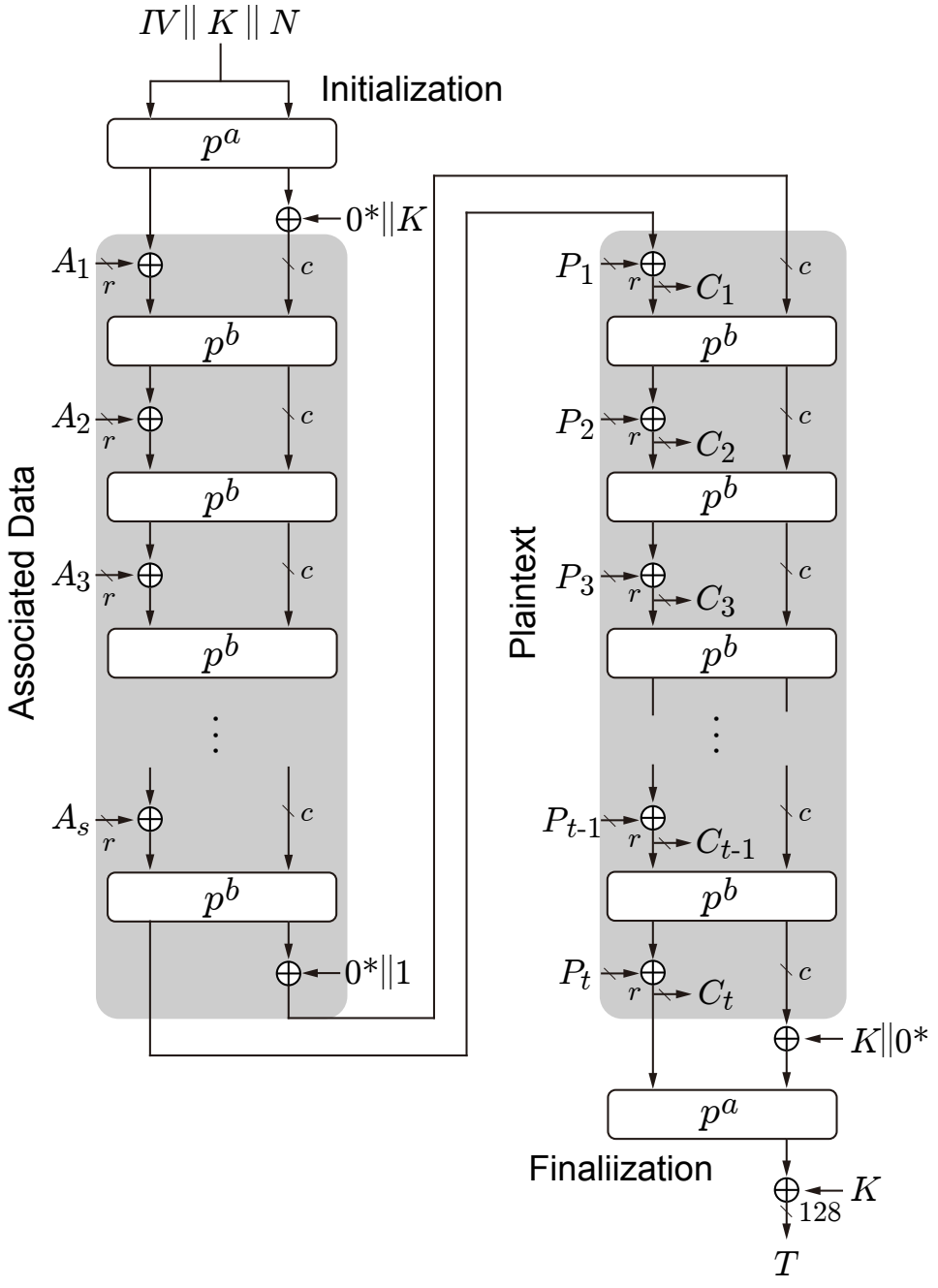

図 1: 認証暗号モードにおける Ascon の暗号化プロセス

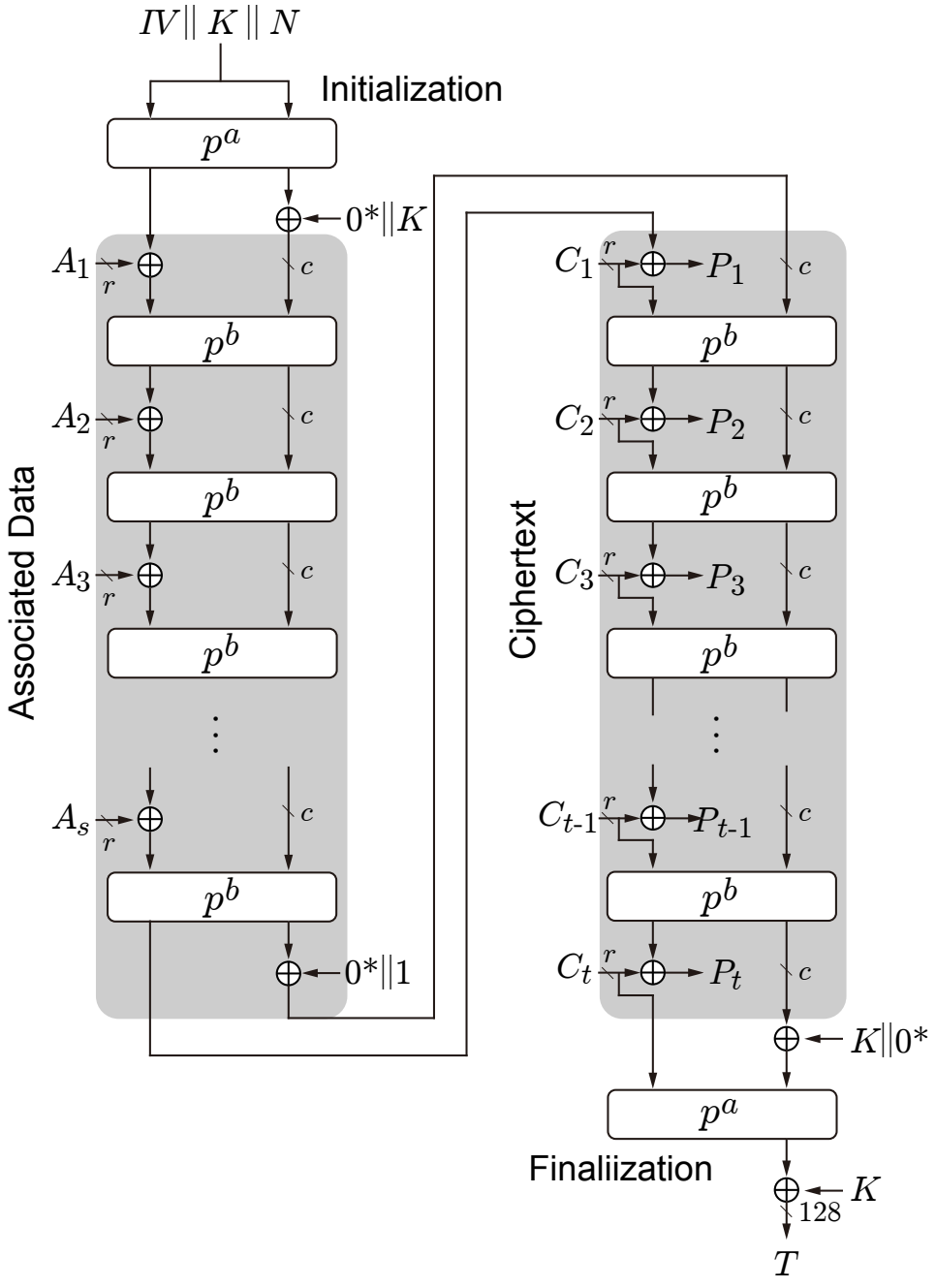

図 2: 認証暗号モードにおける Ascon の復号プロセス

3.2 Ascon の軽量 Permutation

Ascon-128 は，SPN（Substitution Permutation Network）型の Permutation p をラウンド関数として繰り返し使用している．先に，ASCON の認証暗号モードの処理の概要を述べたが，Initialization と Finalization の処理では，p を $a = 12$ 回繰り返す p^a の処理が行われ，AD 及び暗号化／復号処理においては p を $b = 8$ 回繰り返す p^b の処理が行われる．認証暗号モード時における Ascon の暗号化と復号の処理に対するブロック図を，それぞれ図 1 と図 2 に示す．

Ascon の Permutation p では， 5 個のワード（64 ビット長）の x_0, x_1, x_2, x_3, x_4 で構成される 320 ビットのステートに対して以下の処理を行う．

ステップ (1): **ラウンド定数加算** p_C: x_2 に対してラウンドごとに異なる 1 バイトの定数を XOR する．

ステップ (2): **非線形層** p_S: 5 個のワード x_0, x_1, x_2, x_3, x_4 に対してビットスライスを適用し，5 ビットの $x_{0,i}||x_{1,i}||x_{2,i}||x_{3,i}||x_{4,i}$ を入力とする Ascon S-box を 64 回適用する $(0 \leq i < 64)$.

ステップ (3): **線形拡散層** p_L : 5 個のワード x_0, x_1, x_2, x_3, x_4 ごとに，右ローテションシフトと XOR 処理からなる線形処理でステートを攪拌する．

つまり，$p = p_L \circ p_S \circ p_C$ である．

設計者らは，非線形層 p_S のリードマラー標準形（ANF: Algebraic Normal Form）を示している [13]. 以下の通り 2 次である．

$$
\begin{aligned}
y_{0,i} &= x_{4,i}x_{1,i} \oplus x_{3,i} \oplus x_{2,i}x_{1,i} \oplus x_{2,i} \oplus x_{1,i}x_{0,i} \oplus x_{1,i} \oplus x_{0,i},\\
y_{1,i} &= x_{4,i} \oplus x_{3,i}x_{2,i} \oplus x_{3,i}x_{1,i} \oplus x_{3,i} \oplus x_{2,i}x_{1,i} \oplus x_{2,i} \oplus x_{1,i} \oplus x_{0,i},\\
y_{2,i} &= x_{4,i}x_{3,i} \oplus x_{4,i} \oplus x_{2,i} \oplus x_{1,i} \oplus 1,\\
y_{3,i} &= x_{4,i}x_{0,i} \oplus x_{4,i} \oplus x_{3,i}x_{0,i} \oplus x_{3,i} \oplus x_{2,i} \oplus x_{1,i} \oplus x_{0,i},\\
y_{4,i} &= x_{4,i}x_{1,i} \oplus x_{4,i} \oplus x_{3,i} \oplus x_{1,i}x_{0,i} \oplus x_{1,i}.
\end{aligned}
$$

Ascon-128 の S-box は 5 ビットであり，AES S-box のような簡単な代数学的表現はない．テーブル参照でハードウェア実装する場合には，表 3.1 に示す真理値表を用いて実装する．

単純に 64 個の S-box を並列に実装する場合には，5 ビットの真理値表に対応するメモリが 64 個分の容量，つまり $10,240$ ビットのメモリ領域が必要となる．

Ascon S-box のハードウェア実装では，図 3 に示すような組み合わせ回路での実装が効率的である [13]．特に，Ascon では同じ処理が繰り返し実行されるため，ループアーキテクチャにおける組み合わせ回路に対して，いわゆるループアンローリングを適用することで，処理パフォーマンスの向上が得られる場合がある．つまり，高スループットを得るために本来 1 サイクルで実

表 3.1: Ascon の S-box

x	00	01	02	03	04	05	06	07	08	09	0a	0b	0c	0d	0e	0f
$S(x)$	04	0b	1f	14	1a	15	09	02	1b	05	08	12	1d	03	06	1c
x	10	11	02	03	04	05	06	07	08	09	0a	0b	0c	0d	0e	0f
$S(x)$	1e	13	07	0e	00	0d	11	18	10	0c	01	19	16	0a	0f	17

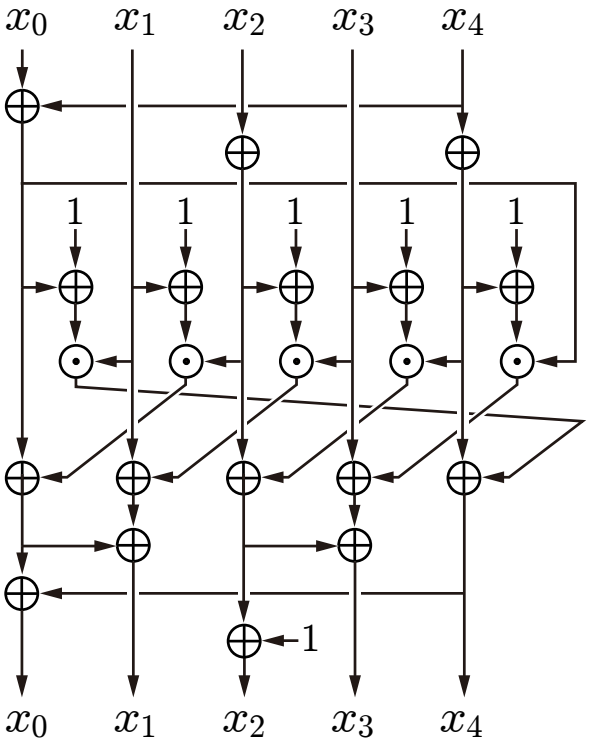

図 3: Ascon の S-box

行する組み合わせ回路のラウンド処理を，アンローリングにより複数ラウンドまとめて最適化することで，クリティカルパス遅延の短縮と同時にレイテンシ（クロックサイクル）の削減が狙える．このように，実装するシステムの様々な仕様に合わせて，面積コストと処理パフォーマンスのトレードオフを柔軟に変更することができる．これは，Ascon-128 の Permutation p のビルディングブロックである 5 ビット S-box の演算処理単位の小ささや，並列処理を可能とするデータ構造から説明できる．

一方，ソフトウェア実装では，ハードウェア実装とは異なり，プラットフォームに適した実装が求められる．特に，メモリ容量が少なく CPU の処理性能が低い IoT デバイスへの実装において十分なサイドチャネル対策を実現するためには，乱数生成に必要なコストと安全性のトレー

ドオフを慎重に模索する必要がある．CPU のワードサイズに合わせて x_0, x_1, x_2, x_3, x_4 に対する非線形演算を，いかに安全に効率よく実装するかが重要となる．

3.3 Ascon に対するサイドチャネル攻撃対策

3.3.1 TI: Threshold Implementation

TI は秘密分散法に基づく乱数マスキングである．2006 年に Nikova らによって提案された [32, 33]．TI では、計算対象の値 x はシェアと呼ばれる複数の値で表現される．ここで，$\mathrm{GF}(2^m)$ 上のある関数 $z = \mathrm{N}(x, y)$ を考える．$\mathrm{GF}(2^m)$ 上の非線形変換 $z = \mathrm{N}(x, y)$ に対しても，シェアの考え方を適用できる．

例えば，3つの関数 $\{\mathrm{f}_1, \mathrm{f}_2, \mathrm{f}_3\}$ が，以下に示す Correctness（正確性），Non-Completeness（不完全性），及び Uniformity（均一性）の性質を有する場合，$\mathrm{N}(x, y)$ はシェアに分けて，2次プロービングモデル[*1]に対して耐性のある計算処理が実現できる．2つ以下のシェアからは元の値を復元することはできないからである．

Correctness

例の場合，関数 $\{\mathrm{f}_1, \mathrm{f}_2, \mathrm{f}_3\}$ は，以下の関係が満たされる場合に Correctness を満たす．

$$
\begin{aligned}
z &= z_1 \oplus z_2 \oplus z_3 \\
&= \mathrm{f}_1(x_2, x_3, y_2, y_3) \oplus \mathrm{f}_2(x_3, x_1, y_3, y_1) \oplus \mathrm{f}_3(x_1, x_2, y_1, y_2) \\
&= \mathrm{N}(x, y).
\end{aligned}
$$

Non-completeness

それぞれの関数が x, y の少なくとも1つのシェア値に依存しないように，例えば次のように計算する．

$$
\begin{aligned}
z_1 &= \mathrm{f}_1(x_2, x_3, y_2, y_3), \\
z_2 &= \mathrm{f}_2(x_3, x_1, y_3, y_1), \\
z_3 &= \mathrm{f}_3(x_1, x_2, y_1, y_2).
\end{aligned}
$$

こうすることで，2つ以下のシェアに分けた関数の処理からは，元の値 x に関する情報を知ることはできない．

Uniformity

例えば x のシェアの発生確率が等しくない場合，攻撃者がその偏りを利用することで，全てのシェアが揃わなくても元の x を復元させることができる．したがって，全ての x のシェアにお

[*1] プロービングモデルは，暗号処理を行うハードウェアやソフトウェアに対して，攻撃者が本来観測することができない内部信号を1本あるいは複数のプローブ（針）を用いて観測可能とする攻撃者モデル [21]．d 次プロービングモデルの場合には，攻撃者は異なる d 本のプローブを用いて d 個の中間値を観測できると仮定する．ただし，同じ回路を使い回すシェア型のハードウェアアーキテクチャの場合，同じプローブで異なる時間の複数の中間値を取得することも想定できる [47]．

いて，そのとりうる値の発生確率は等しくなければならない．

例として $m = 1$ の場合，つまり GF(2) の乗算では，

$$\Pr[x_1, x_2, x_3] = 1/8,$$

を満たさなければならない．

なお，次数 t の関数が，d 次プロービングモデルに対してサイドチャネル攻撃耐性を持つためのシェア数は，最小で $td + 1$ である．

3.3.2 DOM: Domain Oriented Masking

TI の他にも d 次プロービングモデルに耐性を持つマスキング手法がある．シェア数をある規則にしたがって削減する手法として，DOM がよく知られている [19]．TI が，関数レベルで 3 つのプロパティ (Correctness, Non-completeness, Uniformity) の性質を考慮するマスキング方式であるのに対して，DOM では，非線形演算により増加するシェア数を抑制するために，ドメインと呼ばれる概念を導入し，ドメインごとにシェアを再構成する．

例えば DOM では，変数 x のシェア x_0, x_1 を、それぞれドメイン 0 と 1 に関連付ける，そして， d 次プロービングモデルへの耐性を実現するために，変数ごとに $d + 1$ 個のシェアを使用する．つまり，ドメインの数は，$d + 1$ 個である．ここで，1 次プロービングモデルに対して耐性のある DOM GF(2^m) 乗算器を考える．この乗算器を 1 次セキュアな DOM 乗算器と呼ぶ．入力値 x, y は，それぞれシェアで表現され，TI と同様，以下の計算を処理する．

$$\begin{aligned} xy &= (x_0 \oplus x_1)(y_0 \oplus y_1) \\ &= x_0y_0 \oplus x_0y_1 \oplus x_1y_0 \oplus x_1y_1. \end{aligned}$$

ドメイン 0 において，入力 x_0 と y_0 を入力とする AND 演算，つまり x_0y_0 の処理は安全である．なぜなら，どの中間値をプロービング（サイドチャネル情報）により読み出したとしても，x, y が復元できないからである．同様に，x_1y_1 の処理もドメイン 1 で安全に処理される．さらに，x_0y_1 と x_1y_0 の計算においても，x, y から独立しているものであるため，それぞれの処理だけでは x, y に関するサイドチャネルからのリーケージは観測できない．しかし，x_0y_1 をドメイン 0 に取り込み，$x_0y_0 \oplus x_0y_1$ の計算をした場合には問題が生じる可能性がある．異なるドメインのシェア y_0, y_1 が，直接的ではないが XOR 計算で発生するためである．これにより y が即座に復元できるわけではないが，サイドチャネルリークの危険性があると考えるべきである．そこで，x_0y_1 及び x_1y_0 は，いずれもクロスドメインで計算しなければならない処理とみなし，特定のドメイン 0 や 1 における計算とは切り離して考える．ここまでが，DOM における Calculation ステップである．

次に，クロスドメインの計算結果を特定のドメインに取り込むために，Resharing（再シェア）

とよばれるステップを実行する．具体的には，x_0y_1 と x_1y_0 の計算の後に，フレッシュな乱数でマスキングを行う．この Resharing においては，同じ乱数 r は使っても良いとしている．つまり，$x_0y_1 \oplus r$ と $x_1y_0 \oplus r$ のように処理できる．また，クロスドメインに関する一連の処理に起因して生じるグリッチの伝搬については，Rehareing の結果を FF (Flip-Flop) に格納することで情報漏洩を抑止する．パイプライン処理では，ドメイン 0 や 1 における計算のタイミングを揃える必要があるため，x_0y_0 と y_0y_1 の計算結果に対しても，FF (Flip-Flop) に格納する．

最後に，Integration ステップでは，以下のように特定ドメインとクロスドメインの 2 項の統合，つまり XOR 演算を行う．

$$q_0 = (x_0y_0) \oplus (x_0y_1 \oplus r),$$
$$q_1 = (x_1y_0 \oplus r) \oplus (x_1y_1).$$

より高次のセキュア DOM GF(2^m) 乗算器 も同様に設計することができる．また，上述の 3 ステップは，S-box などの非線形演算にも適用できる．

このように DOM は，ドメイン単位の管理によって，シェア数を適切に管理することができ，Resharing における乱数を工夫することで，対策実装コストの低減が期待できる．TI による回路サイズの削減には，数学的な処理の変換が必要となることが多い．一方，DOM による設計手法は，任意の回路に対して単純なステップの処理を繰り返せばよいため，設計の自動化がしやすいマスキング手法であると言える．つまり，DOM によるマスキング実装の設計生産性は高い．なお，DOM により生成した回路のコストは，最適化されていない TI 実装よりも低く，最適化された TI に匹敵するという結果も得られている．TI との実際の実装における安全性については，Ascon だけでなく，様々な暗号アルゴリズムに対して今後比較する必要がある．

3.4 Ascon に対するサイドチャネルからの漏洩評価

3.4.1 CPA: Correlation Power Analysis

電力のサイドチャネル情報を効率よく解析する方法として，最もよく知られているのが 相関電力解析 （CPA: Correlation Power Analysis）である [6]．電磁波サイドチャネルに対しては，CEMA（Correlation ElectroMagnetic Analysis）と呼ばれる．Ascon の実装に対しても，CPA による情報漏洩評価は重要である．

DPA（Differential Power Analysis) [25] では，特定の 1 ビットに対する電力モデルが採用される．一方で，CPA では複数ビットの電力消費をモデル化するため，測定ノイズや処理アルゴリズムに起因するノイズの影響を軽減することが期待できる．DPA では，鍵などの秘密情報の予測にもとづいて電力波形データを 2 つのグループに分け，2 つのデータの平均の差を調べるが，CPA は波形データをより多くのグループに分け，電力モデルとの相関を調べる．Ascon に限らず，多くの暗号アルゴリズムでは，予想した鍵によってレジスタに格納される複数ビットの中間値を選択関数により導出できる場合には，CPA が最適である．

本報告のケーススタディ（4 章）でも，情報漏洩評価として，CPA を用いた解析が採用されている．Ascon に対する CPA 評価は，AEAD 暗号化あるいは復号処理において，攻撃者が秘密鍵の復元ができるかどうかで判断する．選択関数（Selection Function）は，Initialization あるいは Finalization 処理から選ばれることが多い．これは，アルゴリズムの処理に鍵が直接関与しており，鍵予測によって中間値の導出が可能なためである．この選択関数の選び方については，文献 [42] を参照されたい．

3.4.2 TA: Template Attack

DPA とは対照的に，事前のプロファイリングが必要なテンプレート攻撃（TA: Template Attack）も Ascon の攻撃耐性評価では重要である．本報告でも，Ascon のソフトウェア実装への TA の報告をまとめている（4.8 章 [45].）

TA の前提として，暗号アルゴリズムを処理するデバイスが，攻撃者の完全な制御下になければならない．なぜならば，攻撃者が自由に平文や鍵情報をデバイスに設定し，デバイスから漏洩したサイドチャネル情報の確率分布から，デバイスの物理特性をプロファイリングするためである．つまり，TA は簡略化した電力モデルの代わりに，実際のデバイスの複雑な物理的な振る舞いに関するモデルを利用する．攻撃者が無制限にデバイスを実行することができれば，測定ノイズを十分削減することができるため，最も強力な攻撃手法となりうる．なお，プロファイリング型の攻撃が発展したものとして，教師あり機械学習を用いた攻撃やディープラーニング（DL: Deep Learning）などが登場している．Ascon がファイナリストとして選定される前の 2020 年の文献であるが，対策なしの Ascon に対して 24K 個の波形トレース[*2]で DL 攻撃に成功したと

[*2] オシロスコープ等で取得した物理情報の時系列変化の軌跡を波形トレースあるいは単にトレースと呼ぶ．波形ト

している [37].

3.4.3 TVLA: Test Vector Leakage Assessment (Welch's t-test)

ウエルチの t 検定は，さまざまな分野において，広く仮説検定に使用される統計手法である．サイドチャネルリークの評価における t 検定は，TVLA と名付けられている．その目的は，鍵復元や秘密情報の取得ではなく，暗号処理デバイスの内部データとサイドチャネル情報の依存性を評価し，潜在的な脆弱性を特定することにある．攻撃者の計算能力や攻撃手法に関係なく，暗号実装の安全性に関する汎用的な評価指標が提供できるツールとして，広く用いられるようになった．

具体的には，ある 2 つの基準に従って暗号アルゴリズムを実行し，その際に測定したサイドチャネル情報の波形データを，それぞれ集合 A と B に分ける．これらのデータセットの各サンプル点に対して，以下の式で t 値を算出する．

$$\mathrm{t} = \frac{\mu_A - \mu_B}{\sqrt{\frac{\sigma_A^2}{n_A} + \frac{\sigma_B^2}{n_B}}}. \tag{1}$$

ここで，μ、σ^2、n は，サンプル点における波形データ値の平均，分散，及び標本数であり，集合 A と B に対してそれぞれ求める．

サイドチャネルリークの評価で一般的に使われている基準は，鍵を固定にし，波形データの集合の 1 つを固定平文とし，もう 1 つをランダム平文とするものである [18]．$|\mathrm{t}| < 4.5\sigma$ を満たしていれば，そのサンプル点ではサイドチャネルリークがないものと判断する．

平易に表現すると，固定した平文の値に依存したサイドチャネールリークがあるかを調べるものである．もし，平文をランダムに入力した場合のサイドチャネル情報と比較して，なんらかの差異が見られるようであれば，攻撃者はその情報を使って内部の秘密情報を取得できる可能性があると判断する．したがって，t 検定において漏洩の可能性が示されたとしても，具体的な攻撃が実装できるかは不明であるが，未知の攻撃を含めて安全性評価をより厳格に行うことができると言える．

レースの単位として用いることもある．

4 Ascon の物理安全性と実装性に関するケーススタディ

4.1 Niels と Daemen による報告（2017.05） [42]

4.1.1 著者, 所属機関

- Niels Samwel, DiS Group, Radboud University
- Joan Daemen, DiS Group, Radboud University and ST Microelectronics

4.1.2 概要

この論文は，Ascon が，軽量暗号コンペティションの前に実施された CAESAR コンペティション [4] の候補であった時のものである．実際に，Ascon のハードウェア実装に対して，サイドチャネル攻撃 (CPA) を行った最初の論文として重要である．論文では，２つの CAESAR 候補である Keyak と Ascon の FPGA 実装におけるサイドチャネル攻撃が説明されているが，本報告書では，Ascon に絞り論文の要点をまとめる．Ascon は，繰り返し構造を持つスポンジ型の暗号アルゴリズムであり，非線形処理である S-box の出力に対して効率の良い選択関数を提案している．提案した選択関数により，FPGA 上に実装した対策のない Ascon-128 に対して， 50K トレースを用いた CPA 攻撃に成功している．さらに，３シェアの TI 実装 をした Ascon に対しては，シミュレーションベースで CPA 攻撃に成功したとしている．

4.1.3 CPA における選択関数

著者らは，ハードウェア実装におけるステート遷移 $S0_i(M, K^*), S1_i(M, K^*)$ に関する 64 ビットのレジスタレジスタ x'_0, x'_1 に着目し，以下に示す選択関数を提案している．

$$S0_i(M, K^*) = k_0^*(m'_i + 1) + m_i + k_1^*(m'_{i+45} + 1) + m_{i+45} + k_2^*(m'_{i+36}) + m_{i+36},$$

$$S1_i(M, K^*) = m_i(k_0^* + 1) + m'_i + m_{i+3}(k_1^* + 1) + m'_{i+3} + m_{i+25}(k_2^* + 1) + m'_{i+25}.$$

ここで，M は 128 ビットのノンスである．m, m' は，64 ビットのレジスタ，k_i^* は攻撃者が予測する鍵ビットである．一つ目の選択関数により，３ビットの鍵 k_0^*, k_1^*, k_2^* を予測することで，１ビットのステート値が導出できることが分かる．この値に対して，ハミング距離（HD: Hamming Distance）モデルを適用し，64 ビットの x'_0 レジスタからのリークとの相関を調べることで，鍵が復元できるとしている．しかし，一つ目の選択関数だけでは，攻撃成功により導出できる鍵は 高々 64 ビットであるため，二つ目の選択関数を用いて x'_1 のレジスタを攻撃し，全ての鍵が導出可能であるとしている．

4.1.4 攻撃の結果

Samwel と Daemen らは，SAKURA-G [49] に実装した Ascon に対して， 50K トレースで全ての鍵ビットの導出に成功したとしている．これにより，選択関数による電力モデルが効果的であることが分かるった．また，３シェア TI の Ascon に対しては．シミュレーションにより同様の CPA 攻撃を行い，900K トレースで全ての鍵を復元することに成功したとしている．TI 実装の詳細が不明ではあるが，この結果については漏洩の原因は明らかにはされていない．

4.1.5 まとめ

Ascon のハードウェア実装における Initialization へのノンスを用いた CPA 攻撃論文である．著者らは，Ascon のアルゴリズムとハードウェアレジスタのデータ遷移を良く考察した選択関数を導出している．このことは，Ascon のアルゴリズム処理における中間値データの格納の仕方は，バイト単位で処理を行う AES 暗号 [12] などとは異なり，アーキテクチャに強く依存することを意味している．つまり，AES 暗号では S-box の入力値のデータ遷移をモデルとすれば，アーキテクチャをあまり意識することなく攻撃ができたが，Ascon の場合にはそういった汎用的なモデルの構築に難しさがあるとも言える．アーキテクチャに適した選択関数を作成する必要があると思われる．なお，セキュア実装において，著者らはシミュレーションベースの CPA 攻撃に成功している．また，SAKURA-G 上に搭載された FPGA Spartan-6 を用いて実験が行われているが，実装コストや処理パフォーマンスについては説明がない．

本論文では考察されていないが，同様の手法で，Finalization に対しても CPA 攻撃は可能であると思われる．ただし，Initialization への攻撃ほどシンプルではなく，上述のとおりアーキテクチャにも依存するものと思われる．今後の課題として，Ascon に対する，より汎用的なサイドチャネル攻撃モデルの構築があげられる．さらに，Finalization への故障注入が効果的であると思われるため，その攻撃効率や対策技術のコストについても研究が必要であろう．

4.2 Groß の 学位論文（2018.06） [19]

4.2.1 著者, 所属機関

- Hannes Groß, IAIK, Graz University of Technology

4.2.2 概要

本学位論文は，Groß が著者として関与した複数の国際会議論文やジャーナル論文の内容が含まれており，DOM を体系的に理解できるものである．Groß は，TI よりも実装効率の良い暗号アルゴリズムのマスキング対策手法の確立を目指し， 3.3.2 章で紹介した Domain-Oriented Masking (DOM) を提案している．DOM には，Unified Masking (UMA) and Low-Latency Masking (LOLA) と呼ばれる 2 つのバリアントがある．UMA では，暗号アルゴリズムのデータパスにレジスタを追加することで，安全性上クリティカルとされるデータを適切に制御し，DOM の乱数コストを削減している．レジスタを追加することから，1 ラウンドの処理に必要となるサイクル数は増加するため，レイテンシは増加しスループットは低下する．しかし，必要となるフレッシュな乱数は少なくて済む．UMA とは対照的に，LOLA ではレジスタによるステージ数を減らし，処理パフォーマンスの向上を狙うものである．代わりに，非線形処理におけるシェア数が増加するため，より多くのデータの冗長性や追加の回路が必要になり，乱数コストも増加すると報告している．

4.2.3 実装結果

DOM による Ascon のセキュア実装結果を表 4.1 にまとめる．UMC-90nm Low-K の CMOS ライブラリで合成した結果である．UMA については，レイテンシとスループットをを犠牲にして，乱数コストが抑えられることが分かる．ただし，現実的な実装として捉えられる 1 次プロービングモデルに耐性のある 1 次セキュア UMA は，1 次セキュア DOM と比べて，乱数コストと回路の面積コストはほぼ同じである．シェア数が少ない場合には，UMA の実装コスト低下は限定的であると言える．5 次セキュア UMA では，5 次セキュア DOM と比べて，必要となる

表 4.1: Groß による対策付き Ascon AEAD 処理の ASIC 実装の報告（UMC-90nm Low-K）

デザイン	Area [KGE]	Cycle/Round	Throughput [Gbps]	Randomness [bit/cycles]
1 次セキュア DOM	28.89	3	2.25	320
1 次セキュア UMA	27.18	3	2.25	320
1 次セキュア LOLA	42.75	1	2.77	2,048
5 次セキュア DOM	161.87	3	1.86	4,800
5 次セキュア UMA	220.01	7	0.85	3,520
5 次セキュア LOLA	339.82	1	2.99	18,432

フレッシュ乱数を少なくすることに成功しているが，レジスタの追加により実装コスト自体は増加している．

一方，LOLA の実装については，１サイクルで１ラウンドの処理が行えるため，低レイテンシが実現できていることが分かる．面積コストは，DOM や UMA と比べて大きくなり，必要となる乱数が多い５次セキュア LOLA では，１サイクルあたり約 18K ビットと非常に多くのフレッシュな乱数を必要としている．

4.2.4 まとめ

通常の TI よりも少ないシェア数が実現できる DOM は，設計手法としても興味深い．DOM とそのバリアントにより，実装コスト，処理パフォーマンス，及び必要となる乱数におけるトレードオフは大幅に増えている．さらに設計者の選択肢が増えたことに加えて，設計手法自体が規則的であり汎用的なマスキングツールにできることは，生産性の向上につながると思われる．TI の実装においては，暗号アルゴリズムの数学的特徴をうまく利用して，シェア数を少なくする工夫が考えられている．Ascon のマスキング実装との比較については，今後の研究で模索されるべきであり，特に人手による最適化とツールによる最適化との比較を，生産性の観点から見ていく必要があると考える．

4.3 Batina らによる報告（2022.08） [5]

4.3.1 著者, 所属機関

- Lejla Batina, CESCA Radboud University
- Ileana Buhan, CESCA Radboud University
- Łukasz Chmielewski, CESCA Radboud University
- Ellen Gunnarsdóttir, CESCA Radboud University
- Vahid Jahandideh, CESCA Radboud University
- Tom Stock, CESCA Radboud University
- Léo Weissbart, CESCA Radboud University

4.3.2 概要

本論文は，NIST LWC のファイナリストのいくつかに対して，サイドチャネル解析の初期段階の結果をまとめたものである．2022 年 8 月 19 日に公開された Radboud 大の CESCA Lab の研究成果である．評価の対象とする暗号アルゴリズムは，Ascon, Xoodyak, 及び ISAP である．Ascon については，サイドチャネル対策のない実装とマスキング対策によるセキュア実装に対して TVLA と CPA を用いて物理攻撃に対する安全性を評価している．実装形態は，Arm-v6 上のソフトウェア実装であり，サイドチャネル情報として電力波形を用いている．以下，Ascon の安全性評価に絞って報告書の内容を紹介し考察を与える．

4.3.3 攻撃対象及び評価環境

評価に用いた Ascon は，Primary Recommendation である Ascon-128 である．ソフトウェアは，Ascon の開発チームが公開している C コードを用いている．電力測定には，Riscure 社の Piñata development board [39] を用いている．当該ボードには，32 ビットの Arm マイクロコントローラをベースとする SoC STM32F407IGT6 が搭載されている．動作周波数は 168 MHz である．電力波形は，Riscure 社のカレントプローブ（型番不明）と Picoscope 社のオシロスコープ（model 3206D） [40] を用いて取得している．著者らが行った CPA 攻撃では，4.1 章で紹介した Niels と Daemen により提案された選択関数を採用している．

4.3.4 評価結果

電力波形 50K トレースを用いて，対策なしの Ascon 全体の処理に対して TVLA を行った結果，Initialization で t 値をがしきい値を大きく超えていることが示されている．また，Initialization に特化して，100K トレースでの TVLA を実施し，CPA 攻撃に最適なサンプル時間を特定し，500K トーレスを用いた CPA 攻撃により，Niels と Daemen の攻撃手法に従い，正解鍵の復元に成功したことが示されている．２つある選択関数では，攻撃の成功率に差がある

という結果が得られている．

著者らは，マスキング対策のある Ascon に対しても同様の TVLA 及び CPA 攻撃を行っている．Ascon 設計者らによる公式コードの中でも，Arm-v6 向けに作成されたものを用いている．このソフトウェア実装は，乱数をほとんど使わない２～４シェアで対策されたセキュア実装である．この実装に対して，15M トレースを用いて Initialization の最初に処理される Permutation に対して，CPA 攻撃の攻撃箇所（サンプル時間）の特定を行った．ノンスはランダムに変化させ，その他のデータは全て固定としている．15M 波形トーレスを用いて CPA 攻撃を行った結果，暗号処理の２個の中間値を利用する２次 CPA でも攻撃は成功しなかった．攻撃が成功しなかったのは，波形数が少なかったことが原因であるとしている．

4.3.5 まとめ

Ascon-128 のソフトウェア実装について，電力サイドチャネル攻撃の結果を示す論文である．対策なしの実装では容易に鍵復元ができたが，セキュア実装に対しては２次の CPA 攻撃でも鍵を復元することができなかった．Niels と Daemen らのシミュレーションによる CPA の結果とは異なるものである．著者らは，トレース数の問題を指摘している．これは，TVLA の結果から妥当な指摘であるが，選択関数に改善の余地があるようにも思われる．なぜならば，Ascon の場合，ハードウェア実装に用いた選択関数をそのままソフトウェア実装に適用しても，効果的な解析ができるかどうかは明らかではないからである．たしかに，対策なしの Ascon では攻撃に成功しているため，ソフトウェア実装に対しても当該選択関数が一定の精度をもっていると言えるが，乱数を用いたセキュア実装の場合には，マスキングによってアルゴリズムの処理手順が大きく異なることを考慮しなければいけない．最適な選択関数として，アルゴリズムでのデータフローだけでなく，マスキング手法や CPU のアーキテクチャ，及び攻撃者能力の前提を考慮したものを用いるべきである．さらには，プロファイリング型の攻撃に対する脆弱性を調べる必要があると考える．

4.4 Mohajerani らによる報告（2023.06） [30]

4.4.1 著者, 所属機関

- Kamyar Mohajerani: CERG, George Mason University
- Luke Beckwith, CERG: George Mason University, PQSecure Technologies
- Abubakr Abdulgadir: PQSecure Technologies
- Eduardo Ferrufino: CERG, George Mason University
- Jens-Peter Kaps: CERG, George Mason University
- Kris Gaj: CERG, George Mason University

4.4.2 概要

本論文は，Lightweight Cryptography Workshop 2023 (Day 1) で公表されたものである．

NIST の軽量暗号 (LWC) 標準化プロセスにおいて，選定における主要な指標のひとつとされているサイドチャネル攻撃耐性について，複数の研究チームが調査やベンチマークを行った結果を報告している．サイドチャネル攻撃耐性の評価には，多くのマンパワーが必要となる．なぜなら，通常の実装よりも複雑な対策技術をアルゴリズムにあわせて実装し，膨大なサイドチャネル情報を長時間かけて収集し，解析をする必要があるためである．LWC コンペティションでは，多くの暗号アルゴリズムが候補として提出された．ラウンド 1 とラウンド 2 で，それぞれ 56 候補 と 32 候補まで絞られたものの，サイドチャネル耐性評価を公平に行うことは，時間的にも労力的負担が大きいため難しい．最終ラウンドでようやく，候補が 10 になったため，サイドチャネル攻撃耐性を評価する機運が高まったように思う．本論文では，そういったタイミングで，暗号アルゴリズム候補の物理耐性評価を行う機関を集め，物理攻撃耐性の一般的な評価フレームワークを提案している．参加した大学，研究機関，及び企業は次の七つの機関である．

1) IAIK, Graz University of Technology, Austria
2) CCSL, Shanghai Jiao Tong University, China
3) HSCP Lab, Tsinghua University, Beijing, China
4) Secure-IC, France
5) CERG, George Mason University, USA
6) Ruhr-Universitat Bochum, Germany
7) CESCA Lab, Radboud University, the Netherlands

対策技術の安全性の評価を行うとともに，対策技術の追加によって，実装コストと処理パフォーマンスに与える影響を実証実験している．サイドチャネル攻撃に対して耐性を持つセキュア実装に関して，安全性を含めた実装性能が報告されている．本論文では，対象を Ascon に絞り，当該論文の内容をまとめ考察を与える．

4.4.3 評価対象，手法，及び結果

Ascon-128 のハードウェアでのセキュア実装の安全性評価には，FPGA が用いられている．Ascon-128_Bochum_d1 は，Ascon-128_Graz-x1 をベースにマスキング対策が施されたものが使われている．対策のない HDL コードから，乱数マスキング対策のある HDL コードを半自動生成する AGEMA [23] と呼ばれるツールを利用している．このコードは、Ruhr-Universitat Bochum によって生成されたとしている．AGEMA は，合成後のネットリストに対して，サイドチャネル情報攻撃に対して保護する必要があるワイヤとゲートを特定し，それらに対して必要な乱数マスキングを施す．また，AGEMA は Probe Isolating Non-Interference (PINI) [7] とコンポーザビリティの概念に基づいている．しかしながら，AGEMA は，制御ロジックに対する制御ができないため，全てのコード変換を自動化することはできない．そのため，一部のコードに対しては手動でマスキング対策を施す必要がある．これが半自動生成ツールと呼ばれる理由である．Ascon-128_Graz_d1 に対しては，Domain Oriented Masking (DOM) [19] が採用されている．Ascon-128 ハードウェアに対するマスキングは全て 1 個の中間値に着目する 1 次攻撃に対して安全とされるものである．

Ascon の FPGA 上へのセキュア実装と Arm-Cortex-M4 上へのソフトウェア実装に対して，参加した研究機関が行なった安全性評価の結果を，表 4.1 と 表 4.2 にまとめる．ハードウェア評価では，電力と電磁波のサイドチャネル情報に対して，TVLA，χ^2-test, 及び CPA 攻撃で情報漏洩の可能性の有無を確認している．波形数はおよそ 100 万から 1,000 万程度である．CREG による評価だけが TVLA のしきい値である 4.5 を超えたと報告しているが，他の機関からは特段の情報漏洩の可能性はないと報告されている．CERG ラボによる Ascon-128_Bochum_d1 のテストでは，数個 (3〜10) のサンプルで しきい値 4.5 を超えたとしている．これらのテストにおいては，攻撃対象のクロックと同期したサンプリングクロックを使用していたことが原因としている．ただし，1M を超えるトレースが考慮されるまではしきい値を超えていないとしている．

一方，ソフトウェア実装は，すべて Arm Cortex-M4 [1] 上に実装されたものである．Ascon-128_Graz_d1 と Ascon-128_Graz_d2 が用いられている．いずれの安全性評価においても，EM サイドチャネルから取得した波形データ使っている．なお，オシロスコープのサンプリングクロックは，攻撃対象の CPU の動作クロックと同期していない．

評価の結果， CPA 攻撃による鍵復元は成功していない．CESCA グループ による 2 次 CPA 攻撃では，15M トレースを使用しても Ascon-128_Graz_d1 の鍵に関する情報を一切明らかにすることができないとしている．参考までに，対策なしの Ascon 実装に対する CPA 攻撃は，500K トレースで鍵復元に成功している．このことから，用いたコードのマスキング対策は正しく機能していることが分かる．

表 4.2: 対策済み Ascon の FPGA 実装に対する安全性評価結果

ソースコード	評価機関	評価プラットフォーム	オシロスコープ	サイドチャネル	評価手法	波形数（M トレース）	評価結果
Ascon-128_Bochum_d1	CERG	CW305 (Artix-7)	FOBOS3 ADC	電力	TVLA	10	リーク有 (1.5M トレース)
Ascon-128_Bochum_d1	IAIK	CW305 (Artix-7)	PicoScope 6404C	電力	TVLA	10	リーク無
Ascon-128_Bochum_d1	CCSL	SAKURA-X (Kintex-7)	LeCroy 610Zi	電磁波	TVLA	1	リーク無
Ascon-128_Bochum_d1	CCSL	SAKURA-X (Kintex-7)	LeCroy 610Zi	電磁波	χ^2-test	1	リーク無
Ascon-128_Bochum_d1	CCSL	SAKURA-X (Kintex-7)	LeCroy 610Zi	電磁波	CPA	11	リーク無
Ascon-128_Graz_d1	HSCP	SAKURA-G (Spartan-6)	WaveRunner 8404M	電力	TVLA	10	リーク無

表 4.3: 対策済み Ascon のソフトウェア実装に対する安全性評価結果

ソースコード	評価機関	評価プラットフォーム	オシロスコープ	サイドチャネル	評価手法	波形数（M トレース）	評価結果
Ascon-128_Graz_d1	CESCA	STM32F407 (Arm Cortex-M4)	Pico 3206D	電磁波	2次 CPA	15	リーク無
Ascon-128_Graz_d2	CCSL	STM32F303 (Arm Cortex-M4)	Pico 3203D	電磁波	TVLA	0.06	リーク無
Ascon-128_Graz_d2	CCSL	STM32F303 (Arm Cortex-M4)	Pico 3203D	電磁波	χ^2-test	0.06	リーク無
Ascon-128_Graz_d2	CCSL	STM32F303 (Arm Cortex-M4)	Pico 3203D	電磁波	CPA	0.06	リーク無

4.4.4 攻撃対象のクロックとサンプリングクロックとの同期について

オシロスコープのサンプリングと攻撃対象のデバイスのクロックの同期性は，鍵復元攻撃に必要となる波形数に影響を及ぼすことが報告されている [36]．本論文の実験においても，攻撃対象のクロックに同期したサンプリングクロックを使用した場合に、非同期クロックを使用する場合と比べ，大幅に少ない波形数で情報漏洩が検出できるとしている．原因として，データパスの乱数マスキング時に制御ロジックの一部のクロックサイクルで漏洩につながる問題があり，サイドチャネルリークが発生するとしている．オシロスコープのサンプリングを攻撃対象のクロックと同期させることで，少ない波形数でも TVLA の t 値が高くなることが実験的に示されている．サンプリングレートは 50MS/s と高くないにも関わらず，このような結果が得られているのは特筆に値する．単にサイドチャネル情報を取得する波形数を増やすのではなく，測定系での同期に注意を払うことが安全性評価として厳格にできる場合があることを示唆している．しかも，攻撃者が攻撃対象のクロックを観測することは可能であるため，現実的かつ厳密な評価を行う上で，サンプリングクロックの設定は詳細な議論が必要と考える．

4.4.5 対策による面積コストと処理パフォーマンスへの影響

本論文の図から，Ascon-128_Bochum_d1 FPGA 実装では，対策により面積コストがおよそ 3 倍程度増加し，スループットが約 1/3 倍に低下していることが読み取れる．一方，Ascon-128_Graz_d1 では，対策による面積コストの増加は 2 倍弱と Ascon-128_Bochum_d1 に比べて少なく，スループットも Ascon-128_Bochum_d1 ほど低下していないことが分かる．これは，DOM を人手により実装したことで，効率の良い対策技術が実現できているためと思われる．なお，ソフトウェア実装に関する実装結果は資料には掲載されていない．

4.4.6 まとめ

Ascon については，コンペティションの最終候補を選定する段階から，物理攻撃に対する実装上の脆弱性を知ることができている．これは，早期から Ascon に高い関心が高まり，軽量暗号アルゴリズムの選定の段階から，物理攻撃耐性を含めた実装性能評価で，世界中の研究者やエンジニアの協力があったためである．この安全性評価の取り組みに，日本から参加がなかったのは残念である．暗号技術に関する実装のノウハウは，企業で蓄積されるがあまり公にされることはない．このような物理攻撃に対する安全性の評価に関する取り組みでは，思いもよらぬ攻撃に対抗するためにも，グローバルな視野を持って学際的研究を進めていくことが重要と考える．

4.5 Kandi らによる報告（2023.06） [28]

4.5.1 著者, 所属機関

- Aneesh Kandi, Indian Institute of Technology Madras
- Anubhab Baksi, Nanyang Technological University
- Tomas Gerlich, Brno University of Technology
- Sylvain Guilley, Télécom Paris, Secure-IC
- Peizhou Gan, Nanyang Technological University
- Jakub Breier, Silicon Austria Labs
- Anupam Chattopadhyay, Nanyang Technological University
- Ritu Ranjan Shrivastwa, Télécom Paris, Secure-IC
- Zdenek Martinasek, Brno University of Technology
- Shivam Bhasin, Nanyang Technological University

4.5.2 概要

本論文は，Lightweight Cryptography Workshop 2023 (Day 1) で公表されたものである．著者らは，Ascon のハードウェア実装に関して，Ascon AEAD の暗号化とタグ生成，復号とタグ検証，及び Ascon のハッシュ関数としての処理性能について報告している．サイドチャネル攻撃への対策が施されていない実装に加えて，TI によるサイドチャネル攻撃対策と計算の 3 重化による故障利用攻撃対策が紹介されている．サイドチャネル攻撃対策と故障利用攻撃対策は，互いに直交した概念に基づき実装されているため，それぞれの対策が相互に影響しないとしている．つまり，要求仕様に応じて，どちらかの対策を実装することも，両方の対策を実装することも可能としている．

4.5.3 実装性能評価の結果

STM 130nm ライブラリによる ASIC 実装では，TI で保護された Ascon の 5 ビット S-box のゲートサイズは 56 gate で，線形層は 320 gate としている．このことから，1 サイクルで 1 ラウンドを処理する回路では，組み合わせのゲートサイズは少なくとも 12.6 Kgate 程度のコス

表 4.4: Kandi らによる Ascon AEAD 処理の ASIC 実装（STM 130nm）

コア	面積コスト [μm^2]	クリティカルパス遅延時間 [psec]
暗号化とタグ生成	73,803 (1.00)	8,595 (1.00)
復号とタグ検証	71,873 (0.97)	8,586 (1.00)
暗号化とタグ生成（3 シェア TI）	273,857 (3.71)	10,001 (1.16)
復号とタグ検証（3 シェア TI）	274,688 (3.72)	9,981 (1.16)

表 4.5: Ascon AEAD 処理の FPGA 実装（Kintex-7）

コア	面積コスト [LUT]	クロック周期* [psec]
暗号化とタグ生成	944 (1.00)	5,525 (1.00)
復号とタグ検証	1,05 8 (0.97)	5,525 (1.00)
暗号化とタグ生成（3 シェア TI）	3,977 (4.21)	6,024 (1.09)
復号とタグ検証（3 シェア TI）	3,795 (4.02)	6,010 (1.09)

∗ 表 4.4 及び 表 4.6 との比較のため，最大動作周波数からクロック周期を算出した.

トが必要なことが分かる．実際の実装では，ステートを保持する F/F，インターフェイス回路，乱数生成器が必要となる．

著者らは，まず Ascon AEAD の暗号化と復号処理の機能に関して，ASIC と FPGA（Kintex-7）の実装結果をまとめている（表 4.4 と 表 4.5）．ASIC 実装では，暗号化処理と復号処理での違いはほとんど見られないという結果を報告している．これは，暗号化処理と復号処理で Ascon のデーターパスの違いにほとんどないためである．3 シェアの TI による対策実装では，回路面積は 4 倍弱になっている．TI のシェア数は Ascon S-box の代数次数 2 に 1 を加えた 3 以上でなければならず，今回は 3 シェアを採用したとしている．

Kintex-7 は，28 nm テクノロジを採用した FPGA である．最先端テクノロジではないが，低電力でコストパフォーマンスの良い FPGA として広く利用されている．表 4.5 から分かるように，FPGA 実装においても，暗号化処理と復号処理での違いはほとんど見られない．3 シェア TI のセキュア実装では，暗号化処理と復号処理のいずれの回路サイズも 4 倍以上の LUT を必要としている．一方，クロック周期については，10% 程度の増加と少ない．残念ながら，レイテンシやスループットの処理パフォーマンスに対する考察は報告されていない．

4.5.4 S-Box に用いられた 3 シェア TI

本論文で採用された Ascon ハードウェアのアーキテクチャは，1 サイクルで Ascon の Permutation p を処理するものである．前述のとおり，非線形処理の S-box に対して，3 シェア TI が適用されている．その詳細を以下に示す．5 ビットの Ascon S-box の入力値 x_i $(0 \leq i \leq 4)$ を，x_{i0}, x_{i1}, x_{i2} の 3 つの値に分ける．$x_i = x_{i0} \oplus x_{i1} \oplus x_{i2}$ である．3 つのシェアに分けて以下の処理を行うことで，出力 y_i が得られる．ここで，$y_i = y_{i0} \oplus y_{i1} \oplus y_{i2}$ である．

【シェア０】

$$
\begin{aligned}
y_{00} &= x_{00} \oplus x_{01}x_{11} \oplus x_{01}x_{12} \oplus x_{01} \oplus x_{11}x_{21} \oplus x_{11}x_{41} \oplus x_{11}x_{02} \oplus x_{11}x_{22} \oplus x_{11}x_{42} \\
&\quad \oplus x_{11} \oplus x_{21}x_{12} \oplus x_{21} \oplus x_{31} \oplus x_{41}x_{12} \oplus x_{02}x_{12} \oplus x_{12}x_{22} \oplus x_{12}x_{42} \oplus x_{12} \oplus x_{22} \oplus x_{32}, \\
y_{20} &= x_{20} \oplus x_{11} \oplus x_{21} \oplus x_{31}x_{41} \oplus x_{31}x_{42} \oplus x_{41}x_{32} \oplus x_{41} \oplus x_{12} \oplus x_{32}x_{42} \oplus x_{42} \oplus 1, \\
y_{10} &= x_{10} \oplus x_{01} \oplus x_{11}x_{21} \oplus x_{11}x_{31} \oplus x_{11}x_{22} \oplus x_{11}x_{32} \oplus x_{11} \oplus x_{21}x_{31} \oplus x_{21}x_{12} \oplus x_{21}x_{32} \\
&\quad \oplus x_{21} \oplus x_{31}x_{12} \oplus x_{31}x_{22} \oplus x_{31} \oplus x_{41} \oplus x_{02} \oplus x_{12}x_{22} \oplus x_{12}x_{32} \oplus x_{22}x_{32} \oplus x_{22} \oplus x_{32} \oplus x_{42}, \\
y_{30} &= x_{30} \oplus x_{01}x_{31} \oplus x_{01}x_{41} \oplus x_{01}x_{32} \oplus x_{01}x_{42} \oplus x_{01} \oplus x_{11} \oplus x_{21} \oplus x_{31}x_{02} \oplus x_{31} \oplus x_{41}x_{02} \\
&\quad \oplus x_{41} \oplus x_{02}x_{32} \oplus x_{02}x_{42} \oplus x_{02} \oplus x_{12} \oplus x_{22} \oplus x_{42}, \\
y_{40} &= x_{40} \oplus x_{01}x_{11} \oplus x_{01}x_{12} \oplus x_{11}x_{41} \oplus x_{11}x_{02} \oplus x_{11}x_{42} \oplus x_{11} \oplus x_{31} \oplus x_{41}x_{12} \oplus x_{41} \\
&\quad \oplus x_{02}x_{12} \oplus x_{12}x_{42} \oplus x_{12} \oplus x_{32}.
\end{aligned}
$$

【シェア１】y_{01} の式にある $y_{20}x_{11}$ は $x_{20}x_{11}$ の誤りと思われる.

$$
\begin{aligned}
y_{01} &= x_{00}x_{10} \oplus x_{00}x_{11} \oplus x_{00}x_{12} \oplus x_{10}x_{20} \oplus x_{10}x_{40} \oplus x_{10}x_{01} \oplus x_{10}x_{21} \oplus x_{10}x_{41} \oplus x_{10}x_{02} \\
&\quad \oplus x_{10}x_{22} \oplus x_{10}x_{42} \oplus x_{10} \oplus x_{20}x_{11} \oplus x_{20}x_{12} \oplus x_{20} \oplus x_{30} \oplus x_{40}x_{11} \oplus x_{40}x_{12}, \\
y_{11} &= x_{00} \oplus x_{10}x_{20} \oplus x_{10}x_{30} \oplus x_{10}x_{21} \oplus x_{10}x_{31} \oplus x_{10}x_{22} \oplus x_{10}x_{32} \oplus x_{20}x_{30} \oplus x_{20}x_{11} \oplus x_{20}x_{31} \\
&\quad \oplus x_{20}x_{12} \oplus x_{20}x_{32} \oplus x_{20} \oplus x_{30}x_{11} \oplus x_{30}x_{21} \oplus x_{30}x_{12} \oplus x_{30}x_{22} \oplus x_{30} \oplus x_{40}, \\
y_{21} &= x_{10} \oplus x_{30}x_{40} \oplus x_{30}x_{41} \oplus x_{30}x_{42} \oplus x_{40}x_{31} \oplus x_{40}x_{32} \oplus x_{40}, \\
y_{31} &= x_{00}x_{30} \oplus x_{00}x_{40} \oplus x_{00}x_{31} \oplus x_{00}x_{41} \oplus x_{00}x_{32} \oplus x_{00}x_{42} \oplus x_{00} \oplus x_{10} \oplus x_{20} \oplus x_{30}x_{01} \\
&\quad \oplus x_{30}x_{02} \oplus x_{40}x_{01} \oplus x_{40}x_{02} \oplus x_{40}, \\
y_{41} &= x_{00}x_{10} \oplus x_{00}x_{11} \oplus x_{00}x_{12} \oplus x_{10}x_{40} \oplus x_{10}x_{01} \oplus x_{10}x_{41} \oplus x_{10}x_{02} \oplus x_{10}x_{42} \oplus x_{10} \oplus x_{30} \\
&\quad \oplus x_{40}x_{11} \oplus x_{40}x_{12}.
\end{aligned}
$$

【シェア２】

$$
\begin{aligned}
y_{02} &= x_{02}, \\
y_{12} &= x_{12}, \\
y_{22} &= x_{22}, \\
y_{32} &= x_{32}, \\
y_{42} &= x_{42}.
\end{aligned}
$$

本論文では，３シェア実装の別の例を紹介している．

【シェア０】

$$
\begin{aligned}
y_{00} &= x_{01}x_{30} \oplus x_{01}x_{31} \oplus x_{01}x_{32} \oplus x_{01} \oplus x_{02}x_{30} \oplus x_{02}x_{31} \oplus x_{02}x_{32} \oplus x_{02} \oplus x_{11} \oplus x_{12} \\
&\quad \oplus x_{30}x_{40} \oplus x_{30}x_{41} \oplus x_{30}x_{42} \oplus x_{30} \oplus x_{31}x_{40} \oplus x_{31}x_{41} \oplus x_{31}x_{42} \oplus x_{31} \oplus x_{32}x_{40} \oplus x_{32}x_{41} \\
&\quad \oplus x_{32}x_{42} \oplus x_{32}, \\
y_{10} &= x_{01}x_{40} \oplus x_{01}x_{41} \oplus x_{01}x_{42} \oplus x_{01} \oplus x_{02}x_{40} \oplus x_{02}x_{41} \oplus x_{02}x_{42} \oplus x_{02} \oplus x_{11}x_{40} \oplus x_{11}x_{41} \\
&\quad \oplus x_{11}x_{42} \oplus x_{11} \oplus x_{12}x_{40} \oplus x_{12}x_{41} \oplus x_{12}x_{42} \oplus x_{12} \oplus x_{21} \oplus x_{22} \oplus x_{30} \oplus x_{31} \oplus x_{32} \oplus x_{40} \\
&\quad \oplus x_{41} \oplus x_{42}, \\
y_{20} &= x_{01}x_{11} \oplus x_{01}x_{12} \oplus x_{01} \oplus x_{02}x_{11} \oplus x_{02}x_{12} \oplus x_{02} \oplus x_{21} \oplus x_{22} \oplus x_{30} \oplus x_{31} \oplus x_{32} \oplus 1, \\
y_{30} &= x_{01} \oplus x_{02} \oplus x_{11}x_{21} \oplus x_{11}x_{22} \oplus x_{11}x_{30} \oplus x_{11}x_{31} \oplus x_{11}x_{32} \oplus x_{11} \oplus x_{12}x_{21} \oplus x_{12}x_{22} \\
&\quad \oplus x_{12}x_{30} \oplus x_{12}x_{31} \oplus x_{12}x_{32} \oplus x_{12} \oplus x_{21}x_{30} \oplus x_{21}x_{31} \oplus x_{21}x_{32} \oplus x_{21} \oplus x_{22}x_{30} \oplus x_{22}x_{31} \\
&\quad \oplus x_{22}x_{32} \oplus x_{22} \oplus x_{30} \oplus x_{31} \oplus x_{32} \oplus x_{40} \oplus x_{41} \oplus x_{42}, \\
y_{40} &= x_{01}x_{30} \oplus x_{01}x_{31} \oplus x_{01}x_{32} \oplus x_{02}x_{30} \oplus x_{02}x_{31} \oplus x_{02}x_{32} \oplus x_{11} \oplus x_{12} \oplus x_{21}x_{30} \oplus x_{21}x_{31} \\
&\quad \oplus x_{21}x_{32} \oplus x_{21} \oplus x_{22}x_{30} \oplus x_{22}x_{31} \oplus x_{22}x_{32} \oplus x_{22} \oplus x_{30}x_{40} \oplus x_{30}x_{41} \oplus x_{30}x_{42} \oplus x_{30} \\
&\quad \oplus x_{31}x_{40} \oplus x_{31}x_{41} \oplus x_{31}x_{42} \oplus x_{31} \oplus x_{32}x_{40} \oplus x_{32}x_{41} \oplus x_{32}x_{42} \oplus x_{32} \oplus x_{40} \oplus x_{41} \oplus x_{42}.
\end{aligned}
$$

【シェア１】

$$
\begin{aligned}
y_{01} &= x_{00}x_{30} \oplus x_{00}x_{31} \oplus x_{00}x_{32} \oplus x_{00}, \\
y_{11} &= x_{00}x_{40} \oplus x_{00}x_{41} \oplus x_{00}x_{42} \oplus x_{00} \oplus x_{10}x_{40} \oplus x_{10}x_{41} \oplus x_{10}x_{42} \oplus x_{10}, \\
y_{21} &= x_{00}x_{10} \oplus x_{00}x_{12} \oplus x_{00} \oplus x_{02}x_{10} \oplus x_{20}, \\
y_{31} &= x_{00} \oplus x_{10}x_{20} \oplus x_{10}x_{22} \oplus x_{10}x_{30} \oplus x_{10}x_{31} \oplus x_{10}x_{32} \oplus x_{10} \oplus x_{12}x_{20} \oplus x_{20}x_{30} \oplus x_{20}x_{31} \\
&\quad \oplus x_{20}x_{32} \oplus x_{20}, \\
y_{41} &= x_{00}x_{30} \oplus x_{00}x_{31} \oplus x_{00}x_{32} \oplus x_{10} \oplus x_{20}x_{30} \oplus x_{20}x_{31} \oplus x_{20}x_{32}.
\end{aligned}
$$

【シェア２】

$$
\begin{aligned}
y_{02} &= x_{10}, \\
y_{12} &= x_{20}, \\
y_{22} &= x_{00}x_{11} \oplus x_{01}x_{10}, \\
y_{32} &= x_{10}x_{21} \oplus x_{11}x_{20}, \\
y_{42} &= x_{20}.
\end{aligned}
$$

4.5.5　３重化による故障利用攻撃対策

暗号モジュールに対して，これまでに数多くの故障攻撃が提案されている．差分故障解析（DFA: Differential Fault Analysis）[3] は最も知られている攻撃の一つであるが，暗号処理の２

表 4.6: Ascon AEAD の ASIC 実装（STM 130nm）

SCA 対策	FA 対策	面積コスト [μm^2]	クリティカルパス遅延時間 [psec]
-	-	98,524 (1.00)	8,520 (1.00)
-	3重化	258,224 (2.62)	8,518 (1.00)
3シェア TI	-	364,320 (3.70)	9,830 (1.15)
3シェア TI	3重化	948,544 (9.63)	9,832 (1.15)

重化，つまり同じ処理を2回行い，その結果を比較することで誤り暗号文を出力しないといった対策が存在する．Fault Sensitivity Analysis (FSA) [27] や Statistical Ineffective Fault Attack (SIFA) [10] といったさらに高度な攻撃では，単純な2重化の対策では不十分である．最新の共通鍵暗号への故障攻撃については，SoK 論文 [2] が詳しい．

著者らは，同じ処理を3回行い，結果の多数を出力とする対策を提案している．もし，3つの結果が全て異なる場合には，乱数を出力するとしている．故障利用攻撃に対する安全性評価の報告がないため詳細は不明であるが，3重化は原理的には DFA 対策としては十分であると考える．一方で，FSA 攻撃や SIFA 攻撃に対しては，一定の効果はあると思われるが，より強力な攻撃者に対しては厳密な安全性評価が必要と考える．

Ascon のコア部分の回路規模は，単純に3倍となる．より正確には，多数決により結果を決定する処理が必要となるため，3倍よりも大きくなる．表 4.6 から分かるように，実際にはインタフェイスなどの回路面積は増えないため，全体としては3倍弱となっている．また，空間的な3重化を行っているため，クリティカルパス遅延時間への影響はない．

4.5.6 まとめ

Ascon のハードウェア実装に関して，サイドチャネル対策と故障利用解析について実装結果を示している．3シェア TI については，乱数コストを下げる工夫が可能と思われる．また，故障利用解析対策として，3重化は一定の効果があるものの，協力な攻撃者を想定した評価についてはさらなる実験が必要と考える．元のサイズの約 10 倍となるコスト増については，現実的ではなく一層のコスト削減が必要である．故障利用解析においては，Daemen らのトフォリゲートを用いた効率的な対策 [11] が提案されている．また，暗号アルゴリズムだけの対策を進めるのではなく，レーザー検知などのセンサーの利用を考慮するべきである．例えば，文献 [29] では，AES 暗号ハードウェアのレーザーを利用した故障利用攻撃対策を 面積コスト 28% 増で実装できるとしている．処理パフォーマンスの低下はあるものの，センサーの感度を高くすることで，Ineffective Fault を用いた攻撃は全て無効とすることができる．アルゴリズムレベルでの対策とセンサーレベルでの対策を融合する研究が必要と考える．

4.6 Gigerl らによる報告（2023.06） [17]

4.6.1 著者, 所属機関

- Barbara Gigerl, Graz University of Technology
- Florian Mendel, Infineon Technologies
- Martin Schläffer, Infineon Technologies
- Robert Primas, Graz University of Technology

4.6.2 概要

この資料は，Lightweight Cryptography Workshop 2023 (Day 1) で公表されたものである．

この論文では，サイドチャネル攻撃耐性を持つ Ascon の効率的なソフトウェア実装が提案されている．２個の中間値を利用する２次の電力解析攻撃耐性を目指すものである．既存技術での Keccak S-box の１次マスキングを拡張し，コストの高いオンラインでの乱数性を必要としない効率的な２次マスキングが実装されている．また，処理パフォーマンスをさらに向上させるべく，マスクされたソフトウェアにおけるシェアを減らす実装のコツについて議論されている．ソフトウェアは，ARM Cortex-M4 をコアとする STM32F303 マイクロプロセッサに実装され，TVLA によって対策技術の安全性が評価されている．さらに，RISC-V Ibex コアのネットリストに対して，ゲートレベルの安全性検証ツール Coco (Co-Design and Co-Verification of Masked Software Implementations [16]) を使用して，ソフトウェア実装の安全性証明を与えている．マスキング対策の施された Ascon-128 認証暗号化ソフトウェアの ARM 及び RISC-V におけるベンチマークの結果，２シェアと３シェアで，それぞれ約 300， 550 Cycles/Byte のスループットが実現されたとしている．Initialization と Finalization のみに施す対策である Leveled Implementation のテクニックを利用すると，マスクされた実装のスループットは約 90 Cycles/Byte まで増加するとしている．

4.6.3 Coco

Coco は，CPU のネットリスト上に実装されたマスク対策の施されたソフトウェアの協調設計，及び検証ツールである [16]．Coco は，ゲートレベルのネットリストで記述された CPU 上で，マスクされたアセンブリを実行した際の安全性を検証することができ，d プローブを空間的及び時間的に実現することができる．つまり，同じクロックサイクル内の異なる位置で d 回のプロビングによる測定を実行したり，異なるクロックサイクルで同じ位置にプローブ測定を d 回実行したり，あるいは，その両方を組み合わせられる攻撃者を想定することができる．攻撃者が元の情報を復元できない場合，マスクされたソフトウェア実装はこの d プローブモデルにおいて d 次安全であるとみなしてる．

4.6.4 攻撃対象及び評価環境

攻撃対象は，Daeman らによる 2 シェアの 5 ビットの S-box [11] と Shahmirzadi と Moradi による 3 シェアの 5 ビットの S-box [38] である．さらに，Share-rotation と呼ばれる技術を追加している．ターゲットデバイスは，STM32F303 マイクロプロセッサと IBEX である．

STM32F303 は，32 ビットの ARM Cortex-M4 をコアとして搭載しており，ChipWhisperer UFO ボード [31] とツールチェーンの組み合わせで性能を評価を行っている．実験では，鍵，ノンス，平文を乱数マスキングしてから，ターゲットデバイスに送信している．また，本論文では，オンラインの乱数生成器を必要ではないため，ターゲットデバイス上の単純なソフトウェアで生成した乱数を使っている．TVLA の評価では，10M トレースの電力波形を用いて実施している．

一方，IBEX では，RISC-V IBEX コアのネットリスト上で，Ascon Permutation 1 ラウンド分を実装し，サイクルベースのシミュレーションを行っている．形式検証は，先述の Coco を用いて実施している．

4.6.5 評価結果

処理性能については，表 4.7 に示す結果が報告されている．

STM32F303 に対する TVLA の結果，2 シェア実装では，t 検定の一部でリークが見られる．実際のマスキングされたソフトウェア実装では，マイクロアーキテクチャレベルでのリークが存在することは知られており，今回も同様の結果となっている．3 シェア実装では，Share-rotation の効果もあってか，10M トレースの TVLA では，顕著なリークは見られなかった．

4.6.6 まとめ

本論文では，マスキング対策された Ascon のソフトウェア実装に対する安全性評価と性能評価が報告されている．STM32F303 上の実装では，検証ツールと実装結果で安全性評価の結果が一致しない例を紹介している．マイクロアーキテクチャからのリーケージは，これまでの研究でも報告があり，今後さらなら研究が必要と考える．また，協調設計／検証ツールである Coco を用いた RISC-V IBEX に対する評価結果についても言及されている．一般的に，実装前の安全性検証においては，現実よりも厳しい基準を満たすものでなければ意味をなさない．本論文では，追加の対策である Share-rotation により，TVLA でのリークはなくなったとしているが，

表 4.7: Daeman らによる Ascon のソフトウェア実装の結果 [Cycles/Byte]

実装	STM32F303	IBEX
対策なし	59	-
Leveled Implementation	89	-
2 シェア	318	260
3 シェア	542	500

理想的には Share-rotation の効果がシミュレートできるほどの精度を持つ検証ツールが必要と考える．つまり，Share-rotation により， TVLA でリークがなくなる理由をマイクロアーキテクチャと照らして明確にし，検証ツールにフィードバックするなどの取り組みが期待される．

4.7 Liu と Schaumont による報告（2023.06） [28]

4.7.1 著者, 所属機関

- Zhenyuan Liu, Worcester Polytechnic Institute
- Patrick Schaumont, Worcester Polytechnic Institute

4.7.2 概要

本論文は，Lightweight Cryptography Workshop 2023 (Day 1) で公表されたものである．

Ascon のサイドチャネルからの情報漏洩，つまりサイドチャネルリークを分析し，情報漏洩の根本原因の特定をハードウェアのコンポーネント単位で試みるものである．RISC-V SoC として実装された 反復型の Ascon のアクセラレータ実装と，RISC-V (RV32IMC) 上のソフトウェア実装に対して，サイドチャネルリーケージを分析している．ゲートレベルの電力シミュレーションを使用して，電力波形でリーケージが発生する箇所の当をつけ，ハードウェア及びソフトウェア実装のどの処理部分で秘密鍵の漏洩が最も多いかを特定している．シミュレーションによる電力波形と，同じ設計の 180 nm ASIC 実装から測定した波形とを比較し，シミュレーションの精度の違いなどを議論している．

4.7.3 攻撃対象及び評価環境

本研究では，著者らは，Ascon を二つの異なる実装で安全性評価を行っている．ひとつは，ハードウェアコプロセッサによる実装であり，もう一つは，ソフトウェア実装である．いずれの実装でも，180 nm スタンダードセル の RISC-V SoC が利用されている．ゲートレベルのネットリストと実際のシリコンに対して，情報漏洩の根本原因の特定に関する実験を行い，安全性を評価している．

Ascon のコプロセッサには，Fivez が設計した反復型アーキテクチャを使用している [15] ．コプロセッサは，PicoRV32 RISC-V の SoC の一部として組み込まれ，平文や AD などのデータ，IV，ノンス，及び鍵は，全てメモリマップインターフェイス経由で制御される．ハードウェア実装における情報漏洩の根本原因の解析は，アクセラレータへの鍵の取り込み時と Initialization 処理である．4 MHz で動作させたハードウェアに対して，クロック周波数の 4 倍のサンプリング周波数で電力波形を取得している．波形数は 2,000 トレースであり，鍵に対して Random-vs-Fix の評価（固定鍵とランダム鍵に対する評価）を行っている．鍵以外のパラメータは固定としている．

一方，Ascon のソフトウェア実装は，Ascon 設計者による ASCON128v12 のリファレンス実装を `-O2` の最適化オプションでコンパイルしたものを使用している．ソフトウェア実装に対しては，4 MHz 動作の RISC-V (RC32IMC) に対して，クロック周波数と同じサンプリング周波数で電力波形をシミュレーションで取得している．波形数は 1,000 トレースとハードウェアの評

価と比べて少ないが，鍵に対する Random-vs-Fix の評価の条件などは同じである．鍵の取り込み部と Initialization 処理に加え，Finalization 処理も評価の対象としている．

4.7.4 評価の結果

コプロセッサに対する評価

著者らの実装では，鍵を 32 ビットずつコプロセッサに書き込んでいる．RAM に格納された鍵データを読み出し，32 ビットのバスでコプロセッサに送っている．全ての鍵ビットの書き込みには 214 クロックサイクルが必要としている．この鍵の書き込みに関する処理において，サイドチャネルリークのシミュレーションを実施したところ，鍵の漏洩の可能性があるとされるゲート数は，最大で約 750 gates/cycle であるとしている．チップ全体のセル数が，57,671 であることと比べて，ゲート数は十分少ないとしている．

Initialization においても同様の実験を行い，12 クロックサイクルで 2,000 gate 以上のサイドチャネルリークが疑われる箇所が見つかったとしている．しかし，これに関しても，チップ全体からすればごく僅かであるとしている．

ソフトウェアに対する評価

ソフトウェア実装では，Ascon Permuation は 900 サイクルで処理される．Initialization における最初のラウンド処理では，Permutation 処理に加えて，鍵とノンスの読み出し部分に 957 クロックサイクルを消費し，そのうち，239 サイクルでリークの疑いのあるゲートが見つかったとしている[*3]．Initialization の 12 ラウンド目における鍵との XOR 演算には，973 クロックサイクルを消費し，そのうち，30 サイクルでリークの疑いのあるゲートが見つかったとしている．Finalization の最初のラウンドにおける鍵との XOR 演算には，941 クロックサイクルを消費し，そのうち，48 サイクルでリークの疑いのあるゲートが見つかったとしている．Finalization における鍵との XOR 演算には，922 クロックサイクルを消費し，そのうち，26 サイクルでリークの疑いのあるゲートが見つかったとしている．いずれのリークについても，コプロセッサでの評価と同様，チップ全体のセル数と比べて漏洩の可能性のあるゲート数は少ないとしている．

4.7.5 実測による妥当性の評価

最後に著者らは，Ascon ソフトウェア実装に対して，情報漏洩の根本原因に関する分析のシミュレーション結果と，同じ設計に基づくチップ実装から取得したサイドチャネル情報の実装値と比較している．実測では，測定ノイズが問題となるため，使用する電力波形数を 50,000 トレースとしている．

実測値での t 検定の結果は，鍵の読み出し，Initialization 処理，及び Finalization における t 検定の結果と類似している．著者らは，今回行った根本原因の分析に関して，シミュレーションと実測で，意味のあるつながりが見られるとしている．

[*3] クロックサイクルによってリークの疑いのあるゲート数は異なる．

4.7.6 まとめ

この論文の結果から，チップ作製前のシミュレーションによる t 検定においても，リークのある可能性があるゲートが見つけられる可能性が示されたと考える．対策技術が考慮していないリークの存在を解明するツールが期待できるため，研究の方向性として興味深い．しかしながら，今回の実験では，全体の回路サイズに対して検知できた漏洩原因の対象となるゲート数がそもそも少なかったため，漏洩原因の特定には至っていない．シリコン作製前後でのリーケージの検知を比較することで，設計フローにおける早期のサイドチャネルリーク対策につながりうる研究といえる．

プレシリコンにおける安全性解析の有効性は，セキュリティ上のマージンを含むものでなければならないと考える．つまり，実測では検知できない脆弱性をシミュレーションは検知できなければならない．シリコン作製前にリーケージが検知できれば，生産性が向上する．そういった意味においても，本論文を含めて，物理攻撃の安全性を担保する設計及び検証のツール開発にはさらなるエフォートが必要と考える．

4.8 You らによる報告（2023.09） [45]

4.8.1 著者, 所属機関

- Shih-Chun You, University of Cambridge
- Markus G. Kuhn, University of Cambridge
- Sumanta Sarkar, University of Warwick
- Feng Hao, University of Warwick

4.8.2 概要

この資料は，Transaction of Cryptographic Hardware and Embedded Systems (TCHES) 2023 に採択されたものである [45]．本論文では，Ascon 実装のサイドチャネル攻撃のリスクを評価をするために，Weatherley の Ascon-128 の 32 ビット実装 [44] を STM32F303 (Arm Cortex-M4) 上に実装し，電力ベースのテンプレート攻撃について評価結果を紹介するものである．著者らは，フラグメントテンプレート攻撃とビリーフプロパゲーション（Beleif propagatation: 確率伝播法）法とキーエニュメレーション（Key Enumeration: 鍵列挙）技術を組み合わせて安全性評価を行なった．主な成果として，大きく 3 つの報告がなされている．

1) サイドチャネル対策のないコードに対して，コンパイル時に `-Os` で最適化した場合，単一の波形トレースで攻撃成功率が 100% になった．
2) コンパイラの最適化オプションを `-O3` とした場合，電力波形 3 トレースで成功率が約 95 % になった．
3) マスキング対策のあるコードに対して `-Os` で最適化した場合は，最大 2^{24} 個の鍵候補を列挙した後に，電力波形を 20 トレースを用いると攻撃成功率が 90% 以上となった．

1 次マスキングで保護された Ascon 実装であっても，テンプレート攻撃によって鍵漏洩の危険性のあることが示されている．さらには、プログラミングのスタイルの違いや，コンパイラの最適化の設定でさえも結果に大きな影響を与える可能性があることが示されている．

4.8.3 攻撃対象及び評価環境

著者らは，Weatherley が作成した Ascon-128 の C コード [44] を SCA プラットフォームである ChipWhisperer-Lite 上の 32 ビットのミックスドシグナルのマイクロコントローラである STM32F303 に実装した．CPU コアは，Arm Cortex-M4 である．電力波形は，ターゲットボードに 5 MHz のクロックを供給し，10 ビットのオシロスコープ PXIe-5160 を使用して，サンプリングレート 2.5 GHz としている．

著者らの実験によると，フラグメントテンプレート攻撃の結果，異なる鍵を使うことで，より攻撃に有利に働くテンプレートが作成できることを示した．著者らは，ビリーフプロパゲーショ

ンとキーエニュメレーション技術の効果を調べるために，ビリーフプロパゲーションを使用しない場合や，ルーピービリーフプロパゲーションといったアルゴリズムに変更をして，最適化オプション -Os を使って実験を行い，鍵列挙の探索の深さに対する成功率の関係を明らかにした．結果は、単一のトレースでループ状の信念伝播を使用すると、2^{32} 個の鍵列挙がほぼ 100% の成功率で達成でき，ツリー状の確率伝播法を適用すると，2^{20} まで削減できることを示している．

さらに，コンパイラオプションの最適化がテンプレート攻撃に与える影響を調べるために，-O3 で実験を行った．ここで，Weatherley のコードは，Ascon Permutation の部分はアセンブリ記述であるため，最適化オプションの影響は受けないものとしている．実験の結果，攻撃効率は明らかに悪くなることが示されている．3 個の波形トレースで成功率が約 95% であり，ビリーフプロパゲーションとキーエニュメレーション技術の両方が揃わなければ，-O3 での攻撃はほとんど実用的ではないと結論づけている．

最後に，1 次攻撃に耐性のあるブーリアンマスキング対策をした Weatherley の C コードを用いて同様の実験を行った．最適化オプションは，-Os である．実験の結果，他の 2 種類の実験と比べて多くのトレースが必要であるものの，20 トレースで 2^{24} 個のキーエニュメレーションの成功率が 90% となったとしている．

4.8.4 まとめ

本論文の研究により，Ascon に対しても効率的なプロファイリングに基づくサイドチャネル攻撃が効果的であることが明らかとなった．Ascon のセキュア実装に対する効率的な性能評価手のひとつとして重要と思われる．Ascon のソフトウェア実装に対する安全性評価として，この論文でも取り扱われたテンプレート攻撃は，厳格な評価を実施する上で不可欠なものである．プロファイリング技術の深化を考慮しつつ，TVLA やプロファイリングを用いない鍵復元攻撃との関係性についても考慮する必要がある．また，コンパイラオプションによるサイドチャネルリークへの影響については，プラットフォーム毎に，安全性向上にむけた研究の取り組みや既存製品の評価実験が必要である．

参考文献

[1] Arm Cortex-M4, https://www.arm.com/ja/products/silicon-ip-cpu/cortex-m/cortex-m4.

[2] Anubhab Baksi, Shivam Bhasin, Jakub Breier, Dirmanto Jap, and Dhiman Saha, "A Survey On Fault Attacks On Symmetric Key Cryptosystems," *ACM Computing Surveys*, Vol.55, No.4, pages 1-34, 2022.

[3] Eli. Biham and Adi Shamir, "Differential fault analysis of secret key cryptosystems," In Burton S. Kaliski Jr., editor, *Advances in Cryptology — CRYPTO*, volume 1294 of Lecture Notes in Computer Science, pages 513–525 Springer, 1997. Available at https://link.springer.com/content/pdf/10.1007/BFb0052259.pdf.

[4] CAESAR: Competition for Authenticated Encryption: Security, Applicability, and Robustness. https://competitions.cr.yp.to/caesar.html.

[5] Lejla Batina, Ileana Buhan, Łukasz Chmielewski, Ellen Gunnarsdóttir, Vahid Jahandideh, Tom Stock, and Léo Weissbart, "Side-Channel Evaluation Report on Implementations of Several NIST LWC Finalists," Nijmegen : Cryptographic Engineering & Side-Channel Analysis (CESCA) Lab, 2022. Available at https://repository.ubn.ru.nl/bitstream/handle/2066/253567/253567.pdf?sequence=1&isAllowed=y.

[6] Eric Brier, Christophe Clavier, and Francis Olivier, "Correlation Power Analysis with a Leakage Model," In Marc Joye and Jean-Jacques Quisquater, editors, *Cryptographic Hardware and Embedded Systems (CHES)*, volume 3156 of Lecture Notes in Computer Science, pages 16–29. Springer, 2004. Available at https://link.springer.com/content/pdf/10.1007/978-3-540-28632-5_2.pdf.

[7] Gaëtan Cassiers and François-Xavier Standaert, "Trivially and Efficiently Composing Masked Gadgets With Probe Isolating Non-Interference," *IEEE Transactions on Information Forensics and Security*, Vol.15, pages 2542-2555, IEEE, 2020.

[8] Hao Cheng, Johann Großschädl, Ben Marshall, Dan Page, and Thinh Pham, "RISC-V Instruction Set Extensions for Lightweight Symmetric Cryptography," *IACR Transactions on Cryptographic Hardware and Embedded Systems (TCHES)*, Vol.2023, No.1, pages 192-237, 2023. Available at https://tches.iacr.org/index.php/TCHES/article/view/9951/9454.

[9] Jean-Sébastien Coron and Louis Goubin, "On Boolean and Arithmetic Masking against Differential Power Analysis," In Çetin Kaya Koç and Christof Paar, editors, *Cryptographic Hardware and Embedded Systems (CHES)*, volume 1965 of Lecture Notes in Computer Science, pages 231-237, Springer, 2000. Available at https://link.

springer.com/content/pdf/10.1007/3-540-44499-8_18.pdf.

[10] Christoph Dobraunig, Maria Eichlseder, Thomas Korak, Stefan Mangard, Florian Mendel, and Robert Primas, "SIFA: Exploiting Ineffective Fault Inductions on Symmetric Cryptography," *IACR Transactions on Cryptographic Hardware and Embedded Systems (TCHES)*, Vol.2018, No.3, pages 547–572, 2018. Available at https://tches.iacr.org/index.php/TCHES/article/view/7286/6463.

[11] Joan Daemen, Christoph Dobraunig, Maria Eichlseder, Hannes Gross,Florian Mendel, and Robert Primas, "Protecting against Statistical Ineffective Fault Attacks," *IACR Transactions on Cryptographic Hardware and Embedded Systems (TCHES)*, Vol.2020, No.3, pages 508-543, 2020. Available at https://tches.iacr.org/index.php/TCHES/article/view/8599/8166.

[12] Joan Daemen, Vincent Rijmen, "The Design of Rijndael," Springer, 2002. Available at https://cs.ru.nl/~joan/papers/JDA_VRI_Rijndael_2002.pdf.

[13] Christoph Dobraunig, Maria Eichlseder, Florian Mendel, and Martin Schläffer, "Ascon v1.2," *Submission document to NIST*, 2021. Available at https://ascon.iaik.tugraz.at/files/asconv12-nist.pdf.

[14] Christoph Dobraunig, Maria Eichlseder, Florian Mendel, and Martin Schläffer, "Invited talk: The Ascon Family: Lightweight Authenticated Encryption, Hashing, and More," *Lightweight Cryptography Workshop 2023*. Available at https://csrc.nist.gov/csrc/media/Presentations/2023/the-ascon-family/images-media/june-21-mendel-the-ascon-family.pdf.

[15] Michael Fivez, "Energy efficient hardware implementations of CAESAR submissions," *Master's thesis, ESAT COSIC, KULeuven*, 2016. Available at https://www.esat.kuleuven.be/cosic/publications/thesis-279.pdf.

[16] Barbara Gigerl, Vedad Hadzic, Robert Primas, Stefan Mangard, and Roderick Bloem, "Coco: Co-Design and Co-Verification of Masked Software Implementations on CPUs," In Michael Bailey and Rachel Greenstadt, editors, *Proceedings of 30th USENIX Security Symposium, USENIX Security*, pages 1469-1468, ACM, 2021. Available at https://www.usenix.org/system/files/sec21fall-gigerl.pdf.

[17] Barbara Gigerl, Florian Mendel, Martin Schläffer, and Robert Primas, "Efficient Second-Order Masked Software Implementations of Ascon in Theory and Practice," *Lightweight Cryptography Workshop 2023*. Available at https://csrc.nist.gov/csrc/media/Events/2023/lightweight-cryptography-workshop-2023/documents/accepted-papers/04-efficient-second-order-masked-software.pdf.

[18] Gilbert Goodwill, Benjamin Jun, Josh Jaffe, and Pankaj Rohatgi, "A testing method-

ology for side-channel resistance validation," *NIST Non-invasive attack testing workshop*, pages 115-136, 2011. Available at https://csrc.nist.gov/csrc/media/events/non-invasive-attack-testing-workshop/documents/08_goodwill.pdf.

[19] Hannes Groß, "Domain-Oriented Masking—Generically Masked Hardware Implementations," *PhD Thesis, IAIK, Graz University of Technology*, 2018. Available at https://diglib.tugraz.at/download.php?id=5c80ea0c43a56&location=browse.

[20] Hannes Groß, Stefan Mangard, and Thomas Korak, "Domain-Oriented Masking: Compact Masked Hardware Implementations with Arbitrary Protection Order," *IACR Cryptology ePrint Archive, Paper 2023/484*, 2023. Available at http://eprint.iacr.org/2016/486.

[21] Yuval Ishai, Amit Sahai, and David Wagner , "Private Circuits: Securing Hardware against Probing Attacks," In Dan Boneh, editor, *Advances in Cryptology - CRYPTO*, volume 2729 of Lecture Notes in Computer Science, pages 463-481, Springer, 2003. Available at https://link.springer.com/content/pdf/10.1007/978-3-540-45146-4_27.pdf

[22] Aneesh Kandi, Anubhab Baksi, Tomas Gerlich, Sylvain Guilley, Peizhou Gan, Jakub Breier, Anupam Chattopadhyay, Ritu Ranjan Shrivastwa, Zdenek Martinasek, and Shivam Bhasin, "Hardware Implementation of ASCON," *Lightweight Cryptography Workshop 2023*. Available at https://csrc.nist.gov/csrc/media/Events/2023/lightweight-cryptography-workshop-2023/documents/accepted-papers/07-hardware-implementation-of-ascon.pdf.

[23] David Knichel, Pascal Sasdrich, and Amir Moradi, "Generic Hardware Private Circuits Towards Automated Generation of Composable Secure Gadgets," *IACR Transactions on Cryptographic Hardware and Embedded Systems (TCHES)*, Vol.2022, No.1, pages 323-344, 2022. Available at https://tches.iacr.org/index.php/TCHES/article/view/9299/8865.

[24] Paul C. Kocher, "Cryptanalysis of Diffie-Hellman, RSA, DSS, and Other Systems Using Timing Attacks," EXTENDED ABSTRACT, 1995. Available at https://citeseerx.ist.psu.edu/viewdoc/download;jsessionid=466E62B26E868456707AE59A26CA7FFE?doi=10.1.1.397.192&rep=rep1&type=pdf.

[25] Paul Kocher, Joshua Jaffe, and Benjamin Jun, "Differential Power Analysis," In Neal Koblitz, editor, *Advances in Cryptology - CRYPTO*, volume 1666 of Lecture Notes in Computer Science, pages 388–397, Springer, 1996. Available at https://link.springer.com/content/pdf/10.1007/3-540-48405-1_25.pdf.

[26] Sinian Luo, Weibin Wu, Yanbin Li, Ruyun Zhang, and Zhe Liu, "An efficient soft

analytical side-channel attack on Ascon," In Lei Wang, Michael Segal, Jenhui Chen, and Tie Qiu, editors, *Wireless Algorithms, Systems, and Applications*, pages 389–400. Springer, 2022.

[27] Yang Li, Kazuo Sakiyama, Shigeto Gomisawa, Toshinori Fukunaga, Junko Takahashi, and Kazuo Ohta, "Fault Sensitivity Analysis," In Stefan Mangard and François-Xavier Standaert, editors, *Cryptographic Hardware and Embedded Systems (CHES)*, volume 6225 of Lecture Notes in Computer Science, pages 320-334, Springer, 2010. Available at `https://link.springer.com/content/pdf/10.1007/978-3-642-15031-9_22.pdf`.

[28] Zhenyuan Liu and Patrick Schaumont, "Root-cause Analysis of the Side Channel Leakage from ASCON Implementations," *Lightweight Cryptography Workshop 2023*. Available at `https://csrc.nist.gov/csrc/media/Events/2023/lightweight-cryptography-workshop-2023/documents/accepted-papers/13-root-cause-analysis-of-side-channel-leakage.pdf`.

[29] Kohei Matsuda, Tatsuya Fujii, Natsu Shoji, Takeshi Sugawara, Kazuo Sakiyama, Yuichi Hayashi, Makoto Nagata, and Noriyuki Miura, "A 286 F^2/Cell Distributed Bulk-Current Sensor and Secure Flush Code Eraser Against Laser Fault Injection Attack on Cryptographic Processor," *Journal of Solid-State Circuits*, Vol.53, No.11, pages 3174-3182, 2018. Available at `https://da.lib.kobe-u.ac.jp/da/kernel/90005512/90005512.pdf`.

[30] Kamyar Mohajerani, Luke Beckwith, Abubakr Abdulgadir, Eduardo Ferrufino, Jens-Peter Kaps, and Kris Gaj, "SCA Evaluation and Benchmarking of Finalists in the NIST Lightweight Cryptography Standardization Process," *IACR Cryptology ePrint Archive, Paper 2023/484*, 2023. Available at `https://eprint.iacr.org/2023/484.pdf`.

[31] NewAE, CW308 UFO, `https://rtfm.newae.com/Targets/CW308%20UFO/`.

[32] Svetla Nikova, Christian Rechberger, and Vincent Rijmen, "Threshold Implementations Against Side-Channel Attacks and Glitches," In Peng Ning, Sihan Qing, and Ninghui Li, editors, *International Conference and Communications Security (ICICS)*, volume 4307 of Lecture Notes in Computer Science, pages 529-545, Springer, 2006. Available at `https://link.springer.com/content/pdf/10.1007/11935308_38.pdf`.

[33] Svetla Nikova, Vincent Rijmen, and Martin Schläffer, "Secure Hardware Implementation of Nonlinear Functions in the Presence of Glitches,", *Journal of Cryptology*, Vol.24, No.2, pages 292-321, 2011. Available at `https://link.springer.com/content/pdf/10.1007/s00145-010-9085-7.pdf`.

[34] NIST: National Institute of Standards and Technology. `https://www.nist.gov`.

[35] NIST, "NIST IR 8454 Status Report on the Final Round of the NIST Lightweight

Cryptography Standardization Process," Available at https://nvlpubs.nist.gov/nistpubs/ir/2023/NIST.IR.8454.pdf

[36] Colun O' Flynn, "A Framework for Embedded Hardware Security Analysis," PhD thesis, Dalhousie University, 2017. Available at https://dalspace.library.dal.ca/bitstream/handle/10222/73002/OFlynn-Colin-PhD-ECED-June-2017.pdf.

[37] Keyvan Ramezanpour, Abubakr Abdulgadir, William Diehl, Jens-Peter Kaps, and Paul Ampadu, "Active and Passive Side-Channel Key Recovery Attacks on Ascon," *Lightweight Cryptography Workshop 2020*. Available at https://csrc.nist.gov/CSRC/media/Events/lightweight-cryptography-workshop-2020/documents/papers/active-passive-recovery-attacks-ascon-lwc2020.pdf.

[38] Aein Rezaei Shahmirzadi and Amir Moradi, "Re-consolidating first-order masking schemes nullifying fresh randomness," *IACR Transactions on Cryptographic Hardware and Embedded Systems (TCHES)*, Vol.2021, No.1, pages 305–342, 2021. Available at https://tches.iacr.org/index.php/TCHES/article/view/8736/8336.

[39] Riscure, Piñata (Training Target), https://www.riscure.com/products/pinata-training-target/.

[40] Pico Technology, PicoScope, https://www.pico-t.co.jp/product/picoscope/.

[41] 崎山 一男, "軽量暗号の実装性能に関する調査及び評価（NIST 軽量暗号コンペティションファイナリスト）," CRYPTREC 外部評価報告書, CRYPTREC EX-3205-2022, 2022. Available at https://www.cryptrec.go.jp/exreport/cryptrec-ex-3205-2022.pdf.

[42] Niels Samwel and Joan Daemen, "DPA on hardware implementations of Ascon and Keyak," In *Proceedings of the Computing Frontiers Conference*, pages 415–424, ACM, 2017.

[43] Kris Tiri and Ingrid Verbauwhede, "A Logic Level Design Methodology for a Secure DPA Resistant ASIC or FPGA Implementation," In Proceedings of *Design, Automation and Test in Europe Conference and Exposition (DATE*, pages 246-251, IEEE, 2004. Available at https://www.esat.kuleuven.be/cosic/publications/article-697.pdf.

[44] Rhys Weatherley, "Finalists to the NIST lightweight cryptography competition," *GitHub*, 2021. Available at https://github.com/rweather/lwc-finalists/tree/5d2b22c9ff7744be429cabda0c078ea5b7b6f79e.

[45] Shih-Chun You, Markus G. Kuhn, Sumanta Sarkar, and Feng Hao, "Low Trace-Count Template Attacks on 32-bit Implementations of ASCON AEAD," *IACR Transactions on Cryptographic Hardware and Embedded Systems (TCHES)*, Vol.2023, No.4, pages 344–366, 2023. Available at https://tches.iacr.org/index.php/TCHES/article/

view/11169/10608.

[46] Springer, Lecture Notes in Computer Science (LNCS). https://www.springer.com/gp/computer-science/lncs.

[47] Takeshi Sugawara, Yang Li, and Kazuo Sakiyama, "Probing attack of share-serial threshold implementation of advanced encryption standard," *IET Electronics Letters*, Vol.55, No.9, pages 517-519, 2019. Available at https://ietresearch.onlinelibrary.wiley.com/doi/epdf/10.1049/el.2018.7518.

[48] 藤堂 洋介, "軽量暗号の安全性に関する調査及び評価 (Ascon, Grain-128AEAD, Tiny-Jambu)," CRYPTREC 外部評価報告書, CRYPTREC EX-3203-2022, 2022. Available at https://www.cryptrec.go.jp/exreport/cryptrec-ex-3203-2022.pdf.

[49] TROCHE Co.,Ltd., SAKURA-G, SAKURA-X. http://www.troche.com/sakura/order.html.

[50] IEEE, IEEE Xplore. https://ieeexplore.ieee.org/Xplore/home.jsp.