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）

電気通信大学大学院情報理工学研究科
崎山一男

2022年12月

エグゼクティブサマリー

本報告は，米国 NIST (National Institute of Standards and Technology)が推進し，標準化を進めているNIST軽量暗号 (LWC: Lightweight Cryptography)コンペティション [19]でファイナリストに選定された10方式(ASCON, Elephant, GIFT-COFB, Grain128-AEAD, ISAP, Photon-Beetle, Romulus, Sparkle, TinyJambu, Xoodyak) に対する実装性能評価に関する公開情報をまとめ考察を与えたものである．

ハードウェア実装性能の評価報告については，製品の異なるFPGA実装とライブラリの異なるASIC実装がある中で，最も多く使用されたプラットフォームのひとつは，Xilinx社のArtix-7である．図は，Xilinx Artix-7上に実装されたファイナリスト10方式のハードウェアの面積コストとスループット性能をプロットしたものである．細かい設定や測定の条件は考えずに報告された結果のみを用いているため，同じアルゴリズムでも同じ面積コストで異なるスループット性能をプロットしている場合もある．この図から，TinyJambuの回路面積が小さいこと，逆にSPARKLEが比較的面積コストが大きくなることが読み取れる．また，ASCON, Elephant, GIFT-COFB, Romulusのプロットは広く分散しており，軽量実装から高速実装まで，FPGA実装における設計の選択肢が多いアルゴリズムとなっていることがわかる．今後さらに研究が進

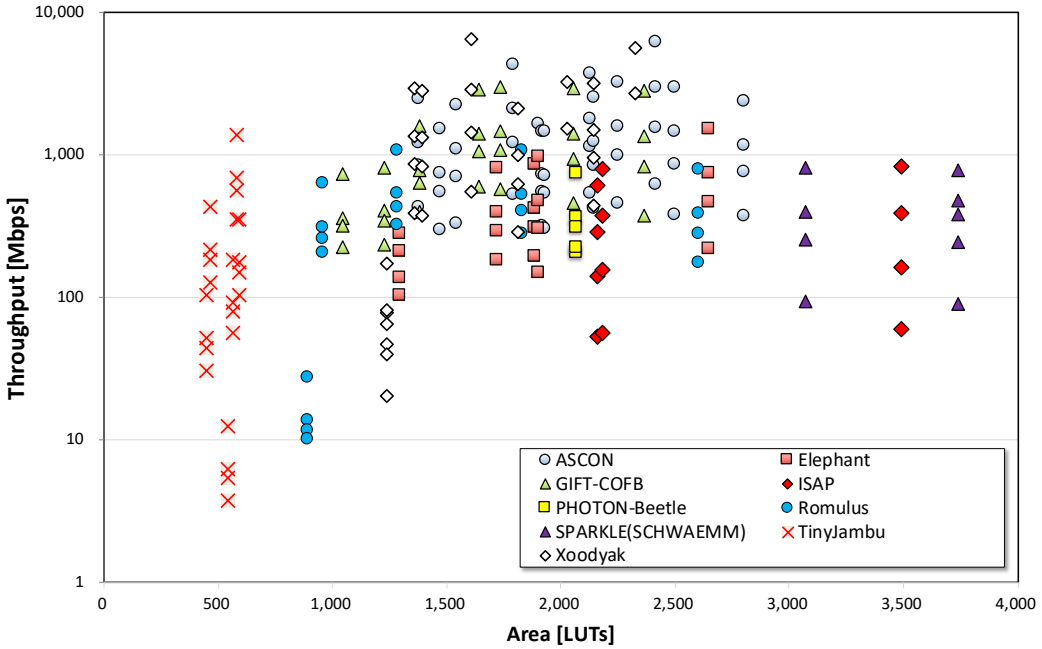

図　NIST 軽量暗号コンペティションファイナリストのXilinx Artix-7上の実装性能．Grain128-AEADは，今回の調査で結果が見つからなかったためプロットはない．

表　NIST軽量暗号コンペティションファイナリストのArm Cortex-M0上の実装性能 [12]

Candidate	Latency [msec]	Code Size [kByte]
ASCON (ascon128v12)	0.633	29.4
Elephant (elephant160v1)	1069	14.4
GIFT-COFB (giftcofb128v1)	6.25	15.1
Grain-128-AEAD (grain128aead)	82.54	15.8
ISAP (isapk128av20)	161.9	14.5
Photon-Beetle (photonbeetleaead128rate128v1)	42.39	15.4
Romulus (romulusn1v12)	17.16	17.0
Sparkle (schwaemm128128v1)	0.76	14.9
TinyJambu (tinyjambu128)	0.394	13.7
Xoodyak (xoodyakv1)	0.599	14.3

み，新たなデータが公開された場合には，他のアルゴリズムにおいても同様の柔軟性が見られる可能性はある．今回，Grain128-AEADについては，Xilinx Artix-7上の公開された実装結果は見つけることができなかった．

軽量暗号のアプリケーションを考えると，低リソースプラットフォーム上での実装性能評価の価値は高い．表は，Arm Cortex-M0上のソフトウェア実装の結果の一部をまとめたものである [12]．括弧内の名称は，各アルゴリズムのプライマリメンバー (primary member)[*1]のリファレンスソフトウェアである．表では，レイテンシとコードサイズをまとめている． RAMの使用量はコンパイル時の静的なメモリサイズのレポートから，どのアルゴリズムも約1kByte程度である．

レイテンシは，平文とADを0バイトから32バイトまで変化させた共通のテストベクトルを用いた際に，暗号化にかかった時間を平均したものである．表から，Elephantが最もレイテンシが高く，TinyJambu，Xoodyak，ASCON, Sparkleが低レイテンシであることがわかる．コードサイズについては，ASCONが最も大きく，他の候補に大きな差はみられなかった．ただし，この結果はほんの一面であり，プラットフォームに適した最適化実装によりより良い結果が得られるものと考える．

このように，公開データを用いて比較の考察を与えることがあるが，厳密な公平性を調べているわけではない．また，実装性能の優劣は公開情報だけでは判断しづらい面があるため，以降はアルゴリズム毎に調査結果をまとめる．

[*1] 提出者が，NISTの要件（鍵128ビット以上，ノンス96ビット以上，タグ64ビット以上）に従ってアルゴリズムのパラメータを設定し，primary memberを一つ決める．

目次

表目次

1　はじめに

暗号アルゴリズムの選定においては，理論的安全性に加えて実装性能を評価することが求められている． NIST軽量暗号コンペティションにおいても，ハードウェアやソフトウェアによる実装性能評価が求めれており，提案者だけでなく多くの研究者が実装評価結果を公表している．本報告は，学術論文や国際会議プロシーディングスなどを中心に，NIST軽量暗号コンペティションのファイナリスト10方式(ASCON, Elephant, GIFT-COFB, Grain128-AEAD, ISAP, Photon-Beetle, Romulus, Sparkle, TinyJambu, Xoodyak)の実装性能評価の結果を調査する．

NIST軽量暗号コンペティションでは，認証付き暗号（AE: Authenticated Encryption with Associated Data)の暗号化と復号及びオプションとしてハッシュ関数の機能を考慮している．本報告の実装性能評価においても，これらの性能を調査する．実装性能の指標として，コスト，処理性能及び物理攻撃安全性であるが，アルゴリズム間でそれらの性能を比較することは極めて困難である．コストと処理性能に関しては，インタフェース回路や最適化手法（高速化を狙った実装，面積を小さくする実装，処理性能と面積コストのバランスを考慮した実装など）により結果が大きく異なってくる．そういった意味では，暗号アルゴリズムのコア部分に絞って実装性能を評価するのではなく，実装するプラットフォームや使用環境との相性で評価するべきものと考える．さらに，物理攻撃安全性においては，暗号アルゴリズムにおける計算処理の違いも影響するが，対策技術の特徴が結果に強く出ることが多いため，本報告では調査対象外とした．

本報告の主な調査対象は，暗号分野と暗号実装に関する学術論文や国際会議プロシーディングスなどである． CAESARプロジェクトでは，GMU (George Mason University)によるATHENAベンチマークシステム [1]でのハードウェア実装性能評価が参考となったが，NIST軽量暗号コンペティションではランキングのわかるデータは当該Webサイトで公開されていない（2022年12月21日時点）．このWebサイトでは，ソースコードや技術報告書（テクニカルレポート）といった検証が必要と思われるデータが中心であることから，今回はWebサイトのデータは直接参照せず，１次情報として発表された論文や国際会議の実装性能評価結果を調査に用いることにした．また，eBACS (ECRYPT Benchmarking of Cryptographic Systems) [5]で公開されている結果は，ソフトウェア実装性能の測定データとして十分よくまとまっているため，本報告では直接データは使用していない．

異なるアルゴリズムの実装性能調査する上で．暗号アルゴリズムの安全性に関するパラメータ，実装プラットフォーム，外部とのデータインタフェースや処理すべきデータといった実装性能に影響する条件を明記することにした．また，軽量実装向きか高速実装向きかといったアルゴリズムの自体の特徴や，得意とする最適化の方針によって実装性能が大きく影響をうけるため，本報告では，公開された情報から厳密な意味で比較を行うことはしない．実装性能評価は，それぞれの認証暗号アルゴリズムのほんの一面にすぎず，認証暗号の優劣を決定するものではないが，実使用で参考となるデータであると思われる．

2　調査対象と性能評価環境

2.1　概要

本報告の調査の対象は，NIST軽量暗号ファイナリストのASIC実装，FPGA実装及びソフトウェア実装の性能評価を研究成果としてまとめた公開出版物とする．国際暗号学会IACRのePrintアーカイブ [14]や国際会議プロシーディングス [15]，Springer社のLNCS (Lecture Note in Computer Science)や学術論文誌 [22]及び米国電気電子学会IEEEの会議プロシーディングスや学術論文誌 [16]を中心とした．

本稿では実装性能を調査した結果を，ファイナリストの暗号アルゴリズム毎に以下の３つのカテゴリに分けて報告をする．

- 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

ここで，ハードウェアアクセラレータは，いわゆる専用回路実装と呼ばれるものであり，処理に必要なデータを供給すれば結果を出力するものである．報告では，ハードウェアアクセラレータをFPGA実装とASIC実装に分けて調査する．軽量暗号の処理に限らないが，専用回路の処理においてインタフェース部がボトルネックとなる場合もあるため，入出力データのインタフェース部を含めて実装する必要がある． NIST軽量暗号ファイナリストに実装性能を可能なかぎり公平に評価する目的で，CAESARコンペティションで用いられたCAESAR HW API [13]とNIST軽量暗号コンペティションで提案されたLWC HW API [13]が使用されることが多いが，独自のインタフェースで評価した研究結果もみられる．

命令拡張では，認証暗号の処理性能を向上させるために命令を追加したCPUのハードウェア実装とそれに伴うソフトウェア性能の評価を行う．認証暗号の処理を汎用のCPUのソフトウェアで処理する場合，複数の命令を組み合わせたプログラムコードを作成するが，新たに命令を追加することでプログラムが単純化され，より少ないサイクル数で実行が可能となる．暗号処理に向けた命令拡張はすでにIntel社のAES-NI (Advanced Encryption Standard New Instructions)で採用されている．

ソフトウェア実装には，さまざまなプラットフォームが考えられる．前述のとおり，eBACSのWebサイト [5]から，主にハイエンドCPU上の実装性能評価を得ることができる．そこで，本報告ではeBACSにはなく，省電力CPUであるArm Cortex-M0での実装性能評価を調査の中心とする．

2.2 ハードウェアアクセラレータ

ハードウェアアクセラレータの調査において，高速実装や軽量実装といったアーキテクチャレベルでの最適化は，RTL (Register Transfer Level)において行われる．さらに，合成時のオプションとしてツールで最適化のオプションを選択することもできる． Githubなどで，RTLコードだけでなく，合成時のスクリプトを公開し，だれでも再現できるようにしている情報もいまでは珍しくない． FPGA実装では，公開している情報をダウンロードすれば，著者らが主張と同じ実装性能結果を得ることができる．一方で，ASIC実装の場合はライブラリの種類が極めて多いことや，合成後の配置配線が実装性能をおおきく左右することなどから，合成条件や配置配線の結果を公開することは難しい．

軽量暗号ハードウェアを評価するインタフェース回路としては，主に 2 方式が提案されている． CAESARプロジェクトで提案されたCAESAR HW API [13]とNIST軽量暗号コンペティションで提案されたLWC HW API [17]である．いずれのインタフェースも暗号処理性能を評価する上で大きな違いはないが，入力されるデータを適切に処理するためには一般的にある程度のバッファメモリを用意する必要がある．その分，回路面積が増えることになるが，安定したデータ転送をハードウェア上で実現することが優先される．他にも，アクセラレータがバスに対して優先的にデータが送れるかといった，バスアービトレーションも考慮する必要がある．こういった詳細な検討は，実際のハードウェアのアーキテクチャを厳密に仕様策定する必要がある．文献[24]のように，CAESAR HW APIの問題点を指摘し改良した報告もある．カスタムのインタフェースを用いた結果や，インタフェースを明確に説明していない論文もある．

2.2.1 コプロセッサ（FPGA実装）

NIST軽量暗号ファイナリストのFPGA実装については，CAESARコンペティション時の評価を含めて，これまでにいくつかの報告がある [10, 26, 20, 24]． CAESARコンペティション時の実装評価結果は，厳密にはNIST軽量暗号の仕様とは異なる場合があるが，実装性能に関する貴重な情報であるため本報告では掲載する．

FPGA上の回路規模（面積コスト）の単位は，統一されていない． Xilinx社のFPGAの場合は， LUT (Look-Up Table)数で評価することが一般的である． Spartan-6, Artix-7及びZynq-7000は，いずれも6入力のLUTであるとしている．一方，Altera社では，Cyclone-VのALM (Adaptive Logic Module)やCyclone 10のLE (Logic Element)と呼ばれる複数のLUTを含むモジュールをビルディングブロックとしている．いずれも 2 入力NANDゲートを 1 単位とするGE (Gate Equivalent)に換算することは可能である．

Mohajeraniらの244ページに及び報告 [18]は，複数の異なるFPGAに対して，入力データの違いによる認証暗号とハッシュ関数の処理性能を網羅的に比較している．インタフェースにはLWC HW APIが用いられている．他にもFPGA実装に関する報告があるが，Mohajeraniらの研究ほ

ど全てのファイナリストの実装を深く研究した論文はない．本報告では，Mohajeraniらの膨大な実装性能評価の結果の中から，特に重要と思われる実装コストとスループット性能についてファイナリスト毎に調査を行った．なお，注目度が高いためか，ASCONをFPGAに実装した研究成果数はファイナリストの中で最も多くなっている．

2.2.2 コプロセッサ（ASIC実装）

ASIC実装に関する研究では，ゲート数やスループット性能に加えて，消費電力やエネルギー（電力量）効率の評価がより重要となる．リソースの限られたIoTデバイスなどを想定しているためである．バッテリーを持たないデバイスの場合には，消費電力が他の指標よりも重要となり，バッテリー駆動のデバイスの場合にはデバイスの寿命に直結するエネルギー効率を議論するべきである．エネルギーは電力を時間積分したものであるため，デバイスの使用率や消費電力の管理手法によりエネルギー効率は大きく変わる．しかし，このような実使用下のフィールドテストによる評価は難しく，アプリケーションにも大きく依存するため，多くの文献では，暗号処理中に特化して電力とエネルギーを評価している．また，評価は認証暗号コアとインタフェースのみの評価となっている．理想的には，通信機能などを含め，認証暗号が組み込まれるデバイス全体の電力やエネルギー効率評価を行うべきであろう．

ASIC実装では，使用するライブラリの特徴やレイアウトなどにより実装性能が大きく左右する．今回調査した限りでは，レイアウト後のシミュレーション結果も見られたが，NISTファイナリストの実装性能をシリコンチップ上で実測した研究はなかった．これは，ASIC実装の場合は評価基板上での動作確認を行うための工程が多いためである．なお，FPGA実装と同様，実装で使用されたインタフェース部の回路はNIST HW APIを含めいくつかあったが，1 kGE程度の回路規模であった．

高スループットを得るために，本来1サイクルで実行する関数（ラウンド関数など）をまとめて実行するアンローリング型のアーキテクチャを設計し評価している研究もあった．暗号アルゴリズムによっては，アンローリングに適していないものもあるが，例えばASCONに対してはアンローリングは柔軟に適用することができる．一般的に，アンロールするラウンド数が多くなるほど組み合わせ回路の面積が大きくなるため，全体の回路面積は大きくなる．また，組み合わせ回路のクリティカルパス遅延時間は長くなるため，最大動作周波数は低くなる．しかし，組み合わせ回路の規模は大きくなるものの，順序回路の規模は変わらない．また，組み合わせ回路を最適化することで，単純にアンロールした関数の遅延時間の足し算にはならず，最大動作周波数はそれほど低下しない．そのため，スループットの向上を狙うことができる．なお，アンローリングによって消費電力は増加するが，所望の処理を短い時間で実行することができるため，エネルギー効率を高められる場合がある．

ASCON-64-bitは，64ビットの算術論理ユニット(ALU)にもとづきデーターパスを設計したものである．ソフトウェア実装のように，ラウンド処理を複数のサイクルに分けて実行するため，処理性能は低下するが，面積は小さくなる． ASCON-x-low-areaは面積を小さくすることを狙

った実装である， ASCON-64-bitよりもさらに小さいデーターパスを使い，3.75KGEのコンパクト実装を実現している． ASCON-64-bitとASCON-x-low-areaは，そちらも消費電力を低く抑えることができるが，ASCON-fastと比べてエネルギー消費は大幅に増加することがわかる．

文献 [9]でElsadekらは，ファイナリストのASIC実装におけるスループット性能とエネルギー消費について報告している．インタフェース回路は考慮されておらず，コアのみの評価となっている．レイアウトは実施していない．

エネルギー効率については，著者らは以下の式を用いている．

$$\text{Energy Efficiency [bit/J]} = \frac{\text{Throughput [bit/sec]}}{\text{Power [W]}}$$

この式からわかるとおり，1ジュールのエネルギーで処理できるビット数は，スループット性能を高めるか消費電力を低くすれば良いことがわかる．スループット性能を高めるためには電力が必要であることから，最適なエネルギー効率は電力とスループットのトレードオフで決まることがわかる．設計段階では，先に述べたアンローリングなどのアーキテクチャレベルでのトレードオフの模索が効果的である．また，クロック周波数や供給電力を変更することでもエネルギー効率の最適化ははかれる．

2.3 命令拡張

命令拡張は，ソフトウェア実装の柔軟性を維持しつつ，性能を格段に向上することができる魅力的な手法である． SteineggerとPrimasは，RISC-V (RI5CY Core)の命令拡張としてASCON-*p*命令の実装を報告している [23]． Chengらの研究では，ファイナリスト10候補に対してRISC-Vの命令拡張の結果を示している [7].

命令拡張により，ハードウェアアクセラレータほどの処理性能を得ることは難しいが，プログラムコードの修正だけで高速化できることや，CPUの周辺回路やインタフェースがそのまま使える．こういった長所はあるが，認証暗号機能を使わない場合や，特定のアルゴリズムに特化した命令の場合，無駄な回路となってしまうことがある． Intel社のCPUに搭載されたAES命令AES-NI (Advanced Encryption Standard New Instructions) のように，広いアプリケーションで長期間の使用が見込める認証暗号アルゴリズムが登場した場合，命令拡張は極めて有望なソリューションといえる．

2.4 ソフトウェア実装

本報告では，ソフトウェア実装についてそれほど多くの調査結果は掲載していない． eBACS (ECRYPT Benchmarking of Cryptographic Systems)で公開されている結果が，十分よくまとまっているためためである．ただし，Intel XeonやArm Cortex-M7といった処理性能の高いCPUでの結果が中心であり，IoTデバイス向けの低消費電力のCPUなどの結果はない．そこで本稿では，低消費CPUであるArm Cortex-M0上で，ファイナリストの処理性能を行なった文

献 [12]を調査することにする．

一般的に，ソフトウェアの処理性能は，1バイトの処理に必要なサイクル数 (cycles/byte)で評価かレイテンシで行う．認証暗号やハッシュの処理では，初期化における処理などとのオーバーヘッド時間が必要となる．したがって，処理するデータ長が短い場合には，処理性能は低くなる傾向になる．ファイナリストの中には，比較的この種のオーバーヘッドが重たいものも存在する．そのため，データ量の少ない処理を行うことが主であるようなアプリケーションを評価をする際には，1バイトの処理に必要なサイクル数よりも，処理に必要なレイテンシを評価する方が適していると場合もある．

一方で，十分にな量のデータを処理する場合には，オーバーヘッドは無視できるため，認証暗号アルゴリズムのベストな処理性能を得ることができる．この場合には，スループット性能が重視されるべきであるため，1バイトの処理に必要なサイクル数が適している．

ソフトウェアプログラムの性能を評価する際には，こういった入力データの量の違いによる結果を調査する必要がある．また，AEADにおいては，平文(PT: Plaintext), 暗号文(CT: Ciphertext)やAD(Associated Data)のサイズをそれぞれ分けて実装性能を評価するものが多い．本報告では，できるかぎりデータの種類と入力データ量に関する情報を付記している．

3 ASCON

3.1 概要

ASCONの設計者（提出者）は以下のとおりである．

- 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`

3.2 ハードウェアアクセラレータ

3.2.1 コプロセッサ（FPGA実装）

ここでは，CAESARコンペティションでの評価結果を含めて，ASCONのFPGA実装の実装性能について述べる．ファイナリストの中で研究報告の数は最も多い．表3.1は，文献 [10, 26, 20, 24]での面積コストとスループット性能に関する研究報告の結果をまとめたものである．FPGAの種類とインタフェースを併記している．

表3.1の中でもっともコンパクトな実装は，文献 [26]の結果で報告のある684 LUTsである．もっとも高速な実装は，文献 [10]で報告されており，約2.9 Gbpsの認証暗号の処理性能

表3.1: ASCONのFPGA実装の結果 [10, 26, 20, 24]

Design	Platform	I/F	Area	Throughput
ASCON-128 [10]	Spartan-6	CAESAR	1,402 LUTs	1,906.3 Mbps
ASCON-128a [10]		HW API	1,712 LUTs	2,884.3 Mbps
ASCON-128 [26]	Spartan-6	CAESAR	684 LUTs	60.10 Mbps
ASCON-128a [26]		HW API	684 LUTs	119.16 Mbps
ASCON-128 [20]	Artix-7	LWC HW API	1,898 LUTs	1683.2 Mbps
	Spartan-6		1,913 LUTs	1116.4 Mbps
	Cyclone-V		1,051 ALMs	1295.6 Mbps
ASCON-128 HASH [20]	Artix-7	LWC HW API	2,181 LUTs	1032.5 Mbps
	Spartan-6		2,188 LUTs	678.1 Mbps
	Cyclone-V		1,064 ALMs	898.0 Mbps
ASCON-128 [24]	Zynq-7000-6	CAESAR-based	6,325 LUTs	-

を1,712LUTsで達成できるとしている．また，ハッシュ関数として機能させた場合，Artix-7上で1.0 Gbpsの処理を2,181 LUTsで達成できるとしている [20].

インタフェースには，CAESARプロジェクトで提案されたCAESAR HW API [13]とNIST軽量暗号コンペティションで提案されたLWC HW API [13]が使われている．文献 [24]では，CAESAR HW APIを改良したインタフェースを用いている．先述のとおり，インタフェースの仕様は，ハードウェアモジュールの面積コストと処理性能に大きな影響を及ぼすため，表の数値だけでASCONの性格な性能を評価することは難しい．しかし，少なくとも，軽量実装が可能なことや高い処理性能を実現できることは表3.1の結果から読み取ることができる．実装性能のトレードオフを模索できる柔軟性はASCONの特徴といえる．

他のFPGA実装の結果としては，Mohajeraniらの報告 [18]が秀逸である．複数の異なる種類のFPGAに対して，入力データの違いによる認証暗号とハッシュ関数の処理性能を網羅的に比較している．研究結果は膨大であるため，面積コストとスループット性能に絞り結果の一部を表3.2にまとめる．

表3.2: ASCONのFPGA実装の結果(つづき) [18]

Design	Platform	Data	Area	Throughput
Ascon-GMU-v1	Artix-7	AD+PT(Long)	2,410 LUTs	6,297.6 Mbps
		Hash(Long)	-	-
	Cyclone 10	AD+PT(Long)	4,552 LEs	3,031.0 Mbps
		Hash(Long)	2,415 LEs	864.4 Mbps
	ECP5	AD+PT(Long)	5,909 LUTs	2,158.1 Mbps
		Hash(Long)	-	-
	Artix-7	AD+PT(1536 Bytes)	2,410 LUTs	3,022.8 Mbps
		AD+PT(64 Bytes)		1,574.4 Mbps
		AD+PT(16 Bytes)		629.8 Mbps
		Hash	-	-
	Cyclone 10	AD+PT(1536 Bytes)	4,552 LEs	1,454.9 Mbps
		AD+PT(64 Bytes)		757.8 Mbps
		AD+PT(16 Bytes)		303.1 Mbps
		Hash	-	-
	ECP5	AD+PT(1536 Bytes)	5,909 LUTs	1,035.9 Mbps
		AD+PT(64 Bytes)		539.5 Mbps
		AD+PT(16 Bytes)		215.8 Mbps
		Hash	-	-
Ascon-GMU-v2	Artix-7	AD+PT(Long)	1,790 LUTs	4,366.2 Mbps
		Hash(Long)	-	-
	Cyclone 10	AD+PT(Long)	3,113 LEs	2,284.7 Mbps
		Hash(Long)	3,215 LEs	1,232.5 Mbps
	ECP5	AD+PT(Long)	4,641 LUTs	1,666.3 Mbps

Continued on next page

表 3.2 – continued from previous page

Design	Platform	Data	Area	Throughput
		Hash(Long)	-	-
	Artix-7	AD+PT(1536 Bytes)	1,790 LUTs	2,115.8 Mbps
		AD+PT(64 Bytes)		1,237.7 Mbps
		AD+PT(16 Bytes)		538.3 Mbps
		Hash	-	-
	Cyclone 10	AD+PT(1536 Bytes)	3,113 LEs	1,107.1 Mbps
		AD+PT(64 Bytes)		647.6 Mbps
		AD+PT(16 Bytes)		281.7 Mbps
		Hash	-	-
	ECP5	AD+PT(1536 Bytes)	4,641 LUTs	807.5 Mbps
		AD+PT(64 Bytes)		472.3 Mbps
		AD+PT(16 Bytes)		205.4 Mbps
		Hash	-	-
Ascon-GMU2-v1h	Artix-7	AD+PT(Long)	1,375 LUTs	2,523.4 Mbps
		Hash(Long)		1,358.8 Mbps
	Cyclone 10	AD+PT(Long)	2,415 LEs	1,605.4 Mbps
		Hash(Long)	4,161 LEs	1,173.5 Mbps
	ECP5	AD+PT(Long)	2,928 LUTs	1,006.3 Mbps
		Hash(Long)		541.8 Mbps
	Artix-7	AD+PT(1536 Bytes)	1,375 LUTs	1,236.9 Mbps
		AD+PT(64 Bytes)		851.3 Mbps
		AD+PT(16 Bytes)		430.8 Mbps
		Hash(1536 Bytes)		1,321.7 Mbps
		Hash(64 Bytes)		812.1 Mbps
		Hash(16 Bytes)		368.0 Mbps
	Cyclone 10	AD+PT(1536 Bytes)	2,415 LEs	786.9 Mbps
		AD+PT(64 Bytes)		541.6 Mbps
		AD+PT(16 Bytes)		274.1 Mbps
		Hash(1536 Bytes)		840.9 Mbps
		Hash(64 Bytes)		516.7 Mbps
		Hash(16 Bytes)		234.1 Mbps
	ECP5	AD+PT(1536 Bytes)	2,928 LUTs	493.2 Mbps
		AD+PT(64 Bytes)		339.5 Mbps
		AD+PT(16 Bytes)		171.8 Mbps
		Hash(1536 Bytes)		527.1 Mbps
		Hash(64 Bytes)		323.9 Mbps
		Hash(16 Bytes)		146.7 Mbps
Ascon-GMU2-v2h	Artix-7	AD+PT(Long)	2,126 LUTs	3,744.0 Mbps
		Hash(Long)		2,139.4 Mbps
	Cyclone 10	AD+PT(Long)	3,215 LEs	2,157.0 Mbps

Continued on next page

表 3.2 – continued from previous page

Design	Platform	Data	Area	Throughput
		Hash(Long)	-	-
	ECP5	AD+PT(Long)	3,764 LUTs	1,427.5 Mbps
		Hash(Long)		815.7 Mbps
	Artix-7	AD+PT(1536 Bytes)	2,126 LUTs	1,825.6 Mbps
		AD+PT(64 Bytes)		1,163.2 Mbps
		AD+PT(16 Bytes)		544.6 Mbps
		Hash(1536 Bytes)		2,077.6 Mbps
		Hash(64 Bytes)		1,248.0 Mbps
		Hash(16 Bytes)		554.7 Mbps
	Cyclone 10	AD+PT(1536 Bytes)	3,215 LEs	1,051.8 Mbps
		AD+PT(64 Bytes)		670.1 Mbps
		AD+PT(16 Bytes)		313.7 Mbps
		Hash(1536 Bytes)		1,196.9 Mbps
		Hash(64 Bytes)		719.0 Mbps
		Hash(16 Bytes)		319.5 Mbps
	ECP5	AD+PT(1536 Bytes)	3,764 LUTs	696.1 Mbps
		AD+PT(64 Bytes)		443.5 Mbps
		AD+PT(16 Bytes)		207.6 Mbps
		Hash(1536 Bytes)		792.1 Mbps
		Hash(64 Bytes)		475.8 Mbps
		Hash(16 Bytes)		211.5 Mbps
Ascon-GMU2-v3h	Artix-7	AD+PT(Long)	2,493 LUTs	3,029.3 Mbps
		Hash(Long)		1,817.6 Mbps
	Cyclone 10	AD+PT(Long)	4,161 LEs	1,955.8 Mbps
		Hash(Long)	-	-
	ECP5	AD+PT(Long)	4,925 LUTs	1,305.6 Mbps
		Hash(Long)		783.4 Mbps
	Artix-7	AD+PT(1536 Bytes)	2,493 LUTs	1,470.0 Mbps
		AD+PT(64 Bytes)		876.0 Mbps
		AD+PT(16 Bytes)		386.7 Mbps
		Hash(1536 Bytes)		1,762.5 Mbps
		Hash(64 Bytes)		1,038.6 Mbps
		Hash(16 Bytes)		454.4 Mbps
	Cyclone 10	AD+PT(1536 Bytes)	4,161 LEs	949.1 Mbps
		AD+PT(64 Bytes)		565.5 Mbps
		AD+PT(16 Bytes)		249.7 Mbps
		Hash(1536 Bytes)		1,137.9 Mbps
		Hash(64 Bytes)		670.6 Mbps
		Hash(16 Bytes)		293.4 Mbps
	ECP5	AD+PT(1536 Bytes)	4,925 LUTs	633.6 Mbps

Continued on next page

表 3.2 – continued from previous page

Design	Platform	Data	Area	Throughput
		AD+PT(64 Bytes)		377.5 Mbps
		AD+PT(16 Bytes)		166.7 Mbps
		Hash(1536 Bytes)		759.6 Mbps
		Hash(64 Bytes)		447.6 Mbps
		Hash(16 Bytes)		195.8 Mbps
Ascon-Graz-v1	Artix-7	AD+PT(Long)	1,465 LUTs	1,528.0 Mbps
		Hash(Long)		873.1 Mbps
	Cyclone 10	AD+PT(Long)	2,517 LEs	1,131.0 Mbps
		Hash(Long)		646.3 Mbps
	ECP5	AD+PT(Long)	2,544 LUTs	474.2 Mbps
		Hash(Long)		271.0 Mbps
	Artix-7	AD+PT(1536 Bytes)	1,465 LUTs	752.0 Mbps
		AD+PT(64 Bytes)		552.5 Mbps
		AD+PT(16 Bytes)		301.8 Mbps
		Hash(1536 Bytes)		850.1 Mbps
		Hash(64 Bytes)		528.6 Mbps
		Hash(16 Bytes)		242.1 Mbps
	Cyclone 10	AD+PT(1536 Bytes)	2,517 LEs	556.6 Mbps
		AD+PT(64 Bytes)		409.0 Mbps
		AD+PT(16 Bytes)		223.4 Mbps
		Hash(1536 Bytes)		629.2 Mbps
		Hash(64 Bytes)		391.3 Mbps
		Hash(16 Bytes)		179.2 Mbps
	ECP5	AD+PT(1536 Bytes)	2,544 LUTs	233.4 Mbps
		AD+PT(64 Bytes)		171.5 Mbps
		AD+PT(16 Bytes)		93.7 Mbps
		Hash(1536 Bytes)		263.8 Mbps
		Hash(64 Bytes)		164.1 Mbps
		Hash(16 Bytes)		75.1 Mbps
Ascon-Graz-v2	Artix-7	AD+PT(Long)	1,541 LUTs	2,272.0 Mbps
		Hash(Long)		973.7 Mbps
	Cyclone 10	AD+PT(Long)	2,634 LEs	1,529.1 Mbps
		Hash(Long)		655.3 Mbps
	ECP5	AD+PT(Long)	2,603 LUTs	683.1 Mbps
		Hash(Long)		292.8 Mbps
	Artix-7	AD+PT(1536 Bytes)	1,541 LUTs	1,108.6 Mbps
		AD+PT(64 Bytes)		712.8 Mbps
		AD+PT(16 Bytes)		336.6 Mbps
		Hash(1536 Bytes)		948.0 Mbps
		Hash(64 Bytes)		589.5 Mbps

Continued on next page

表 3.2 – continued from previous page

Design	Platform	Data	Area	Throughput
		Hash(16 Bytes)		269.9 Mbps
	Cyclone 10	AD+PT(1536 Bytes)	2,634 LEs	746.1 Mbps
		AD+PT(64 Bytes)		479.7 Mbps
		AD+PT(16 Bytes)		226.5 Mbps
		Hash(1536 Bytes)		638.0 Mbps
		Hash(64 Bytes)		396.7 Mbps
		Hash(16 Bytes)		181.7 Mbps
	ECP5	AD+PT(1536 Bytes)	2,603 LUTs	333.3 Mbps
		AD+PT(64 Bytes)		214.3 Mbps
		AD+PT(16 Bytes)		101.2 Mbps
		Hash(1536 Bytes)		285.0 Mbps
		Hash(64 Bytes)		177.2 Mbps
		Hash(16 Bytes)		81.2 Mbps
Ascon-Graz-v3	Artix-7	AD+PT(Long)	2,142 LUTs	2,572.8 Mbps
		Hash(Long)		1,608.0 Mbps
	Cyclone 10	AD+PT(Long)	3,716 LEs	1,403.6 Mbps
		Hash(Long)		877.3 Mbps
	ECP5	AD+PT(Long)	3,305 LUTs	815.0 Mbps
		Hash(Long)		509.4 Mbps
	Artix-7	AD+PT(1536 Bytes)	2,142 LUTs	1,260.1 Mbps
		AD+PT(64 Bytes)		857.6 Mbps
		AD+PT(16 Bytes)		428.8 Mbps
		Hash(1536 Bytes)		1,564.2 Mbps
		Hash(64 Bytes)		961.8 Mbps
		Hash(16 Bytes)		436.1 Mbps
	Cyclone 10	AD+PT(1536 Bytes)	3,716 LEs	687.5 Mbps
		AD+PT(64 Bytes)		467.9 Mbps
		AD+PT(16 Bytes)		233.9 Mbps
		Hash(1536 Bytes)		853.4 Mbps
		Hash(64 Bytes)		524.7 Mbps
		Hash(16 Bytes)		237.9 Mbps
	ECP5	AD+PT(1536 Bytes)	3,305 LUTs	399.2 Mbps
		AD+PT(64 Bytes)		271.7 Mbps
		AD+PT(16 Bytes)		135.8 Mbps
		Hash(1536 Bytes)		495.5 Mbps
		Hash(64 Bytes)		304.7 Mbps
		Hash(16 Bytes)		138.1 Mbps
Ascon-Graz-v4	Artix-7	AD+PT(Long)	2,249 LUTs	3,296.0 Mbps
		Hash(Long)		1,648.0 Mbps
	Cyclone 10	AD+PT(Long)	3,730 LEs	1,738.4 Mbps

Continued on next page

表 3.2 – continued from previous page

Design	Platform	Data	Area	Throughput
		Hash(Long)		869.2 Mbps
	ECP5	AD+PT(Long)	3,379 LUTs	989.6 Mbps
		Hash(Long)		494.8 Mbps
	Artix-7	AD+PT(1536 Bytes)	2,249 LUTs	1,605.2 Mbps
		AD+PT(64 Bytes)		1,004.5 Mbps
		AD+PT(16 Bytes)		462.6 Mbps
		Hash(1536 Bytes)		1,603.1 Mbps
		Hash(64 Bytes)		985.7 Mbps
		Hash(16 Bytes)		446.9 Mbps
	Cyclone 10	AD+PT(1536 Bytes)	3,730 LEs	846.6 Mbps
		AD+PT(64 Bytes)		529.8 Mbps
		AD+PT(16 Bytes)		244.0 Mbps
		Hash(1536 Bytes)		845.5 Mbps
		Hash(64 Bytes)		519.9 Mbps
		Hash(16 Bytes)		235.7 Mbps
	ECP5	AD+PT(1536 Bytes)	3,379 LUTs	481.9 Mbps
		AD+PT(64 Bytes)		301.6 Mbps
		AD+PT(16 Bytes)		138.9 Mbps
		Hash(1536 Bytes)		481.3 Mbps
		Hash(64 Bytes)		296.0 Mbps
		Hash(16 Bytes)		134.2 Mbps
Ascon-Graz-v5	Artix-7	AD+PT(Long)	2,797 LUTs	2,400.0 Mbps
		Hash(Long)		1,600.0 Mbps
	Cyclone 10	AD+PT(Long)	4,905 LEs	1,281.9 Mbps
		Hash(Long)		854.6 Mbps
	ECP5	AD+PT(Long)	4,646 LUTs	889.8 Mbps
		Hash(Long)		593.2 Mbps
	Artix-7	AD+PT(1536 Bytes)	2,797 LUTs	1,173.3 Mbps
		AD+PT(64 Bytes)		775.8 Mbps
		AD+PT(16 Bytes)		376.5 Mbps
		Hash(1536 Bytes)		1,555.4 Mbps
		Hash(64 Bytes)		948.1 Mbps
		Hash(16 Bytes)		426.7 Mbps
	Cyclone 10	AD+PT(1536 Bytes)	4,905 LEs	626.7 Mbps
		AD+PT(64 Bytes)		414.4 Mbps
		AD+PT(16 Bytes)		201.1 Mbps
		Hash(1536 Bytes)		830.8 Mbps
		Hash(64 Bytes)		506.4 Mbps
		Hash(16 Bytes)		227.9 Mbps
	ECP5	AD+PT(1536 Bytes)	4,646 LUTs	435.0 Mbps

Continued on next page

表 3.2 – continued from previous page

Design	Platform	Data	Area	Throughput
		AD+PT(64 Bytes)		287.6 Mbps
		AD+PT(16 Bytes)		139.6 Mbps
		Hash(1536 Bytes)		576.7 Mbps
		Hash(64 Bytes)		351.5 Mbps
		Hash(16 Bytes)		158.2 Mbps
Ascon-Graz-v6	Artix-7	AD+PT	-	-
		Hash	-	-
	Cyclone 10	AD+PT	-	-
		Hash	-	-
	ECP5	AD+PT(Long)	5,346 LUTs	827.9 Mbps
		Hash(Long)		496.8 Mbps
		AD+PT(1536 Bytes)		402.4 Mbps
		AD+PT(64 Bytes)		245.3 Mbps
		AD+PT(16 Bytes)		110.4 Mbps
		Hash(1536 Bytes)		482.7 Mbps
		Hash(64 Bytes)		292.2 Mbps
		Hash(16 Bytes)		130.7 Mbps
Ascon-VT-v1	Artix-7	AD+PT(Long)	1,913 LUTs	1,491.2 Mbps
		Hash(Long)	-	-
	Cyclone 10	AD+PT(Long)	2,432 LEs	1,130.4 Mbps
		Hash(Long)	-	-
	ECP5	AD+PT(Long)	3,130 LUTs	543.4 Mbps
		Hash(Long)	-	-
	Artix-7	AD+PT(1536 Bytes)	1,913 LUTs	735.4 Mbps
		AD+PT(64 Bytes)		560.1 Mbps
		AD+PT(16 Bytes)		320.7 Mbps
	Cyclone 10	AD+PT(1536 Bytes)	2,432 LEs	557.5 Mbps
		AD+PT(64 Bytes)		424.6 Mbps
		AD+PT(16 Bytes)		243.1 Mbps
	ECP5	AD+PT(1536 Bytes)	3,130 LUTs	268.0 Mbps
		AD+PT(64 Bytes)		204.1 Mbps
		AD+PT(16 Bytes)		116.9 Mbps
Ascon-VT-v2	Artix-7	AD+PT(Long)	1,928 LUTs	1,475.4 Mbps
		Hash(Long)		934.4 Mbps
	Cyclone 10	AD+PT(Long)	2,695 LEs	1,158.7 Mbps
		Hash(Long)		733.9 Mbps
	ECP5	AD+PT(Long)	3,041 LUTs	508.1 Mbps
		Hash(Long)		321.8 Mbps
	Artix-7	AD+PT(1536 Bytes)	1,928 LUTs	726.9 Mbps
		AD+PT(64 Bytes)		544.3 Mbps

Continued on next page

表 3.2 – continued from previous page

Design	Platform	Data	Area	Throughput
		AD+PT(16 Bytes)		304.7 Mbps
		Hash(1536 Bytes)		910.4 Mbps
		Hash(64 Bytes)		572.1 Mbps
		Hash(16 Bytes)		264.5 Mbps
	Cyclone 10	AD+PT(1536 Bytes)	2,695 LEs	570.9 Mbps
		AD+PT(64 Bytes)		427.5 Mbps
		AD+PT(16 Bytes)		239.3 Mbps
		Hash(1536 Bytes)		715.0 Mbps
		Hash(64 Bytes)		449.3 Mbps
		Hash(16 Bytes)		207.7 Mbps
	ECP5	AD+PT(1536 Bytes)	3,041 LUTs	250.3 Mbps
		AD+PT(64 Bytes)		187.5 Mbps
		AD+PT(16 Bytes)		104.9 Mbps
		Hash(1536 Bytes)		313.5 Mbps
		Hash(64 Bytes)		197.0 Mbps
		Hash(16 Bytes)		91.1 Mbps
LWC HW APIを用いている．認証暗号の性能は暗号化時のものである．				

表3.2において，Designの項目はRTLコードの名前であり，設計者とバージョンの違いがわかるようになっている． Dataには入力データの種類があり，例えばAD+PT(Long)は，十分にデータ長の長いADとPTに対して，暗号化処理を行なう場合を意味する．括弧内がバイト数の場合は，入力長を意味する．同じFPGA実装でも，入力長の違いによりスループット性能が異なることがわかる．最も高速な結果では，2,410 LUTsで6 Gbpsを超える報告がある．これはArtix-7上で，入力がAD+PT(Long)の場合の結果である[*2]．

3.2.2 コプロセッサ（ASIC実装）

Großらは，90 nm UMC low-Kライブラリを用いてASCONの専用回路 (ASIC)としての性能を調べた [11]．その結果を表3.3にまとめる．

インタフェース部の回路（鍵レジスタと64ビットのバスインタフェース）はおよそ0.87から1.18 kGEのゲートサイズである． ASCON-fastは，高スループットの実装を狙った実装である． ASCON-fast 1 roundは，アンロールなしの実装を意味している．他にも，2,3,6ラウンド分を1サイクルで実行するASCON-fast 2 rounds, 3 rounds, 6 roundsの結果をまとめる．今回のGroßらの結果から，面積コストと処理性能のトレードオフがみられる．表3.3の結果では，アンロール数が多くなるほど，高い処理性能は高くなり，ASCON-fast 6 roundsで約13 GHzのス

[*2] 実際に高スループット性能を達成するためには，データの入出力がボトルネックにならないことが条件である

表3.3: GroßらによるASCONのASIC実装の結果 [11]

Design	I/F	Area	Throughput @f_{max}	Power @1MHz	Energy
ASCON-fast 1 round	no	7.08 kGE	5,524 Mbps	43 μW	33 μJ/byte
	custom	7.95 kGE			
ASCON-fast 2 rounds	no	10.61 kGE	8,425 Mbps	72 μW	27 μJ/byte
	custom	11.48 kGE			
ASCON-fast 3 rounds	no	14.26 kGE	10,407 Mbps	102 μW	25 μJ/byte
	custom	15.13 kGE			
ASCON-fast 6 rounds	no	24.93 kGE	13,218 Mbps	184 μW	23 μJ/byte
	custom	25.80 kGE			
ASCON-64-bit	no	4.99 kGE	72 Mbps	32 μW	1,397 μJ/byte
	custom	5.86 kGE			
ASCON-x-low-area	no	2.57 kGE	14 Mbps	15 μW	5,706 μJ/byte
	custom	3.75 kGE			

ループット性能が見られている．なお，ASCON-fast 6 roundsでは消費電力は最大であるが，処理時間が短くなるためエネルギー効率は最も良くなっていることがわかる．

表3.3に記載のASCON-64-bitは，64ビットの算術論理ユニット(ALU)にもとづいてデーターパスを設計したものである．ソフトウェア実装のように，ラウンド処理を複数のサイクルに分けて実行するため処理性能は低下するが，面積コストは下がる． ASCON-x-low-areaは面積を小さくすることを狙った実装である， ASCON-64-bitよりもさらにデーターパスを軽量化し，3.75 kGEのコンパクト実装を実現している． ASCON-64-bitとASCON-x-low-areaは，どちらも消費電力を低く抑えることができるが，ASCON-fastと比べてエネルギー効率は大幅に増加することがわかる．処理時間が長くなるためである．

Dobraunigらは文献 [8]で，CAESARプロジェクトで使われたCAESAR Hardware API [13]を想定したコプロセッサ実装の結果を紹介している．インタフェース回路は，CAESAR HW APIである．表3.2にまとめたGroßらの結果と比較して面積コストが若干高くなっているが，これはCAESAR HW APIの構成が原因と考える．このように，ASCONの回路規模は，インタフェース回路の影響を大きくうけていることがわかる．

表3.4: DobraunigらによるASCONのASIC実装の結果 [8]

Design	I/F	Area	Throughput @f_{max}	Power	Energy
ASCON-128 1 round		9.42 kGE	4,888 Mbps	-	-
ASCON-128 2 rounds	CAESAR	12.99 kGE	8,482 Mbps	-	-
ASCON-128 3 rounds	HW API	16.59 kGE	10,343 Mbps	-	-
ASCON-128 6 rounds		27.28 kGE	12,261 Mbps	-	-
ASCON-128a 1 round	CAESAR	9.68 kGE	7,326 Mbps	-	-
ASCON-128a 2 rounds	HW API	13.25 kGE	11,743 Mbps	-	-
ASCON-128a 4 rounds		20.38 kGE	16,675 Mbps	-	-

他にも，Elsadekらは，GF 22 nm CMOS (GF22FDx)で合成したASCONのASIC実装につい

て，スループット性能とエネルギー効率を報告している [9]．インタフェースは考慮されておらず，コア部のみの実装性能評価となっている．表3.5にその結果をまとめる．

表3.5: ElsadekらによるASCONのASIC実装の結果 [8]

Design	I/F	Area	Data	Throughput @f$_{max}$	Energy Efficiency
ASCON-128 (Enc.)	no	11 kGE	Short/PT	39.1 Mbps	407.2 Mbit/mJ
			Short/AD	37.8 Mbps	371.7 Mbit/mJ
			Short/PT+AD	70.4 Mbps	640.8 Mbit/mJ
ASCON-128(Enc.)	no	11 kGE	Long/PT	522.0 Mbps	2,614.7 Mbit/mJ
			Long/AD	522.0 Mbps	2,531.3 Mbit/mJ
			Long/PT+AD	531.9 Mbps	2,600.4 Mbit/mJ

表中のShort/PTは，入力(PT)のデータ長が短い場合で，16バイトのデータを間隔を空けて送信する想定である． PTのみあるいはADのみの場合においては，40 Mbs程度の処理性能がでていることがわかる．PT+ADの評価では70 Mbps程度となっている．

入力データがLongの場合では，1536バイトの連続した入力データの処理を想定している．Shortと比べて，スループット性能は高くなり，500 Mbpsを超えている．

エネルギー効率については，著者らは以下の式を用いている．

$$\text{Energy Efficiency [bit/J]} = \frac{\text{Throughput [bit/sec]}}{\text{Power[W]}}$$

例えば，表3.5の 1 行目のデータに対する電力（平均）は，$39.1/407.2 = 0.0960$となるため，96μWと算出できる．また，1 ビットの処理に必要なエネルギーを求める場合は，エネルギー効率と処理データ数を用いて，$128/407.2 = 0.3144$となるため，0.3144μJ/bit（2.51μJ/byte)と算出できる． GroßらによるASIC実装 [11]で報告されているエネルギー消費の値と比べると大きな違いがあることがわかる．どちらの性能評価も合成結果後のシミュレーションによる見積もりであることが原因として考えられる．電力やエネルギー効率の正確な測定には，レイアウト後の正確なシミュレーションや実チップでの測定が不可欠と考える．

3.3 命令拡張

SteineggerとPrimasは，RISC-V (RI5CY Core)にASCON-p命令を追加し，命令拡張による実装性能評価を報告している [23]． ASCON-pはASCONのビルディングブロックであり，ASCON-p命令を実装することでソフトウェアの柔軟性を維持しつつ，処理性能の向上が得られるとしている．論文では，65 nm UMC low leakageライブラリ(umc065LL1P10M)を用いて，ASCON-p命令を追加したRISC-Vプロセッサのソフトウェア処理性能が報告されている．表3.6は，その結果をASCONを認証暗号として用いた場合とハッシュ関数として用いた場合に分けてまとめたものである．

ASCON-Cは，ASCON-p命令を使わないC言語プログラムをオプション(-O3, -Os)付きでコ

表3.6: SteineggerとPrimasによるASCON-p命令拡張によるASCONの実装結果 [23]

Design	Area overhead	Cycles/Byte			Binary size
		64 Byte	1,536 Byte	Long	
ASCON-C (-O3)	-	164.3	110.6	108.3	11,716 Byte
ASCON-C (-Os)	-	269.7	187.1	183.5	2,104 Byte
ASCON-ASM + Acc.	4.7 kGE	4.2	2.2	2.1	888 Byte
ASCONHASH-C (-O3)	-	306.9	208.0	203.8	20,244 Byte
ASCONHASH-C (-Os)	-	423.3	268.0	261.3	1528 Byte
ASCONHASH-ASM + Acc.	4.7 kGE	4.6	2.6	2.5	484 Byte

ンパイルした場合の結果である.. ASCON-ASM + Acc.は，拡張命令ASCON-pを用いている．プログラムは，アセンブリ言語で記述している．処理性能（1バイト処理に必要なサイクル数：Cycles/Byte)は，３つの入力データ長にわけて測定している．

ASCON-p命令をRSIC-Vプロセッサに追加することで，4.7 kGEの論理回路が追加で必要となることがわかる．命令拡張により，認証付き暗号，ハッシュ関数のどちらの処理性能も大幅に向上し，１バイトあたりに必要なサイクル数は大幅に少なくなることがわかる．結果として，50〜80倍の性能向上が4.7 kGEの追加回路で得られていることは特筆に値する．また，命令拡張により必要なプログラムサイズ(RAM)の削減にも寄与している．

Chengらもファイナリスト10候補に対してRISC-Vの命令拡張による実装性能評価の結果を発表している [7]. SteineggerとPrimasによる研究とは，特定のアルゴリズムに特化した命令拡張ではない点で大きく異なる．また，FPGA(Kintex-7)上の実装結果となる．表3.7は，論文の結果をCycles/Byteに変換しまとめたものである. Designの項目において，RV32GCはRISVC-Vのオリジナルのコアであり，+以降は命令拡張の種類を意味する．研究で用いられたRV32GCコアの面積コストは3,303 LUTsであり，性能評価で用いられた入力データ長はいずれも128バイトである．

表3.7: Chenらによる命令拡張によるASCONの実装結果 [7]

Design	Area	Cycles/Byte	Binary size
ASCON RV32GC (Enc.)	3,303 LUTs	336.0	-
ASCON RV32GC+Zbkb/x (Enc.)	3,764 LUTs	252.5	-
ASCON RV32GC+Zbkb/x+ν_0^{32} (Enc.)	4,234 LUTs	131.1	-
ASCON RV32GC (Dec.)	3,303 LUTs	269.7	-
ASCON RV32GC+Zbkb/x (Dec.)	3,764 LUTs	255.4	-
ASCON RV32GC+Zbkb/x+ν_0^{32} (Enc.)	4,234 LUTs	134.1	-

SteineggerとPrimasによる研究と比べて大きな性能向上はないものの，面積コストを抑えた汎用的な命令拡張で，数倍の性能向上が得られることがわかる．

3.4 ソフトウェア実装

文献 [8]で，著者らはAMD, Intel, Armプロセッサに ASCONを実装した際の実装性能の評価

結果を報告している．

表3.8: ASCONのソフトウェア実装結果 [8]

Design	Platform	Cycles/Byte			
		1 Byte	32 Byte	64 Byte	Long
ASCON-128	AMD Ryzen 7 1700	-	-	14.5	8.6
	Intel Xeon E5-2609 v4	-	-	17.3	10.5
	Arm Cortex-A53(Armv8)	-	-	18.3	11.0
	Arm Cortex-A15(Armv7)	-	-	69.8	34.6
ASCON-128(*)	Intel Core i5-6300U	367	23	17.6	11.4
	Intel Core i5-4200U	521	32	23.9	15.8
	Arm Cortex-A7(NEON)	2705	99	73.2	47.9
	Arm Cortex-A7(Armv7)	1871	115	86.6	57.2
	Arm1176JZF-S(Armv6)	2189	133	97.9	65.3
ASCON-128a	AMD Ryzen 7 1700	-	-	12.0	5.7
	Intel Xeon E5-2609 v4	-	-	14.1	6.9
	Arm Cortex-A57(Armv8)	-	-	15.1	7.3
	Arm Cortex-A15(Armv7)	-	-	60.3	23.8
ASCON-128a(*)	Intel Core i5-6300U	365	19	13.5	7.8
	Intel Core i5-4200U	519	27	18.8	10.6
	Arm Cortex-A7(NEON)	2705	83	57.6	32.6
	Arm Cortex-A7(Armv7)	1911	102	71.3	41.2
	Arm1176JZF-S(Armv6)	2267	120	84.4	50.2

(*)はECRYPTベンチマーキング(eBACS)からの引用である [5]

処理性能の高いCPUでは，メッセージ長がLongの場合において，1バイトあたり10サイクル未満の結果が得られている． Armv7やArmv6では，およそ1/5から1/4倍程度の性能となっていることがわかる．メッセージ長の短い場合の評価結果，特に1バイトの場合において必要なサイクル数が大幅に増加しており，性能が低い印象を与えるが，実際のレイテンシは無視できるほど低い値になる．

文献 [12]で，日良らはアルゴリズムの提出者が作成したリファレンスコードを低消費電力のプロセッサーであるArm Cortex-M0に移植し，実装性能としてレイテンシとROMサイズ（コードサイズ）を評価している． CPUの動作周波数は48MHzである．測定では，平文とADを0バイトから32バイトまで変化させ，暗号化と復号にかかるレイテンシを分けて測定している．テストベクトルは，平文とADのそれぞれ2バイトずつ変化させた17×17＝289通りの組み合わせで構成されている．表中の結果は，この289通りのテストベクトルの全ての処理にかかるレイテンシの総和となっている．カッコ内の数値はテストベクトル実行した際の 1 回の処理に必要なレイテンシの平均値である．表3.9に結果をまとめる．

また，文献 [25]では，Arm Cortex-M3とAVR ATmega 128でのソフトウェア実装性能を評価している． 16バイトのADと16バイトの入力データに対して，暗号化／復号の処理性能を評価している．動作周波数は，Arm Cortex-M3では84MHz，AVR ATmega 128では16MHzである．表3.10にこれらの結果をまとめる．

テストベクトルや動作周波数が異なるため一概に述べることは難しいが，いずれの結果でも数

表3.9: ASCONのArm Cortex-M0上でのレイテンシ評価 [12]

Design	0 to 32Bytes Encryption [msec]	0 to 32 Bytes Decryption [msec]	ROM [kByte]	Code [kByte]
ascon128av12	153 (0.529)	155 (0.536)	30.6	28.6
ascon128v12	183 (0.633)	185 (0.640)	31.4	29.4
ascon80pqv12	185 (0.640)	188 (0.650)	31.3	29.3

表3.10: ASCONのArm Cortex-M3とAVR ATmega上のレイテンシ評価 [25]

Design	Platform	16-byte AD + 16-byte Msg. [msec]	ROM [Byte]	RAM [Byte]
ASCON-128 (Enc.)	AVR ATmega	5.84	9,732	157
ASCON-128 (Dec.)	@16 Mhz	5.86		181
ASCON-128 (Enc.)	Arm Cortex-M3	0.30	4,764	196
ASCON-128 (Dec.)	@84 MHz	0.31		121

十バイト程度のデータであれば，数msec程度での処理が可能であることがわかる．文献 [25]では，コードサイズが最適化されていることが読み取れる．さらに実装性能を上げるためには，認証アルゴリズムの特徴を理解しCPUに合わせた最適化を図る必要があると考える．

4 Elephant

4.1 概要

Elephantの設計者（提出者）は以下のとおりである．

- Tim Beyne
- Yu Long Chen
- Christoph Dobraunig
- Bart Mennink

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://www.esat.kuleuven.be/cosic/elephant/`
- 仕様：`https://csrc.nist.gov/CSRC/media/Projects/lightweight-cryptography/documents/finalist-round/updated-spec-doc/elephant-spec-final.pdf`

4.2 ハードウェアアクセラレータ

4.2.1 コプロセッサ（FPGA実装）

ElephantのFPGA実装については，NIST主催のLightweight Cryptography Workshop 2022でのAbdulgadirらの報告がある [2]． Artix-7上にLWC HW APIを用いた実装で，1536バイトのデータを暗号化した際に214 Mbpsのスループットを達成している（表4.1）．また，Mohajeraniらの報告 [18]を表4.2にまとめる．同じ1,291 LUTsの面積コストで文献 [2]とスループット性能が異なる結果が見られるが，その理由は不明である．

表4.1: ElephantのFPGA実装の結果 [2]

Design	Platform	I/F	Area	Throughput
Elephant [2]	Artix-7	LWC HW API	1,291 LUTs	214.3 Mbps

表4.2: ElephantのFPGA実装の結果（つづき） [18]

Design	Platform	Data	Area	Throughput
Elephant-v1	Artix-7	AD+PT(Long)	1,291 LUTs	282.9 Mbps
		Hash(Long)	-	-
	Cyclone 10	AD+PT(Long)	2,056 LEs	201.5 Mbps
		Hash(Long)	-	-
	ECP5	AD+PT(Long)	2,368 LUTs	120.5 Mbps

Continued on next page

表 4.2 – continued from previous page

Design	Platform	Data	Area	Throughput
		Hash(Long)	-	-
	Artix-7	AD+PT(1536 Bytes)	1,291 LUTs	139.8 Mbps
		AD+PT(64 Bytes)		103.9 Mbps
		AD+PT(16 Bytes)		66.8 Mbps
	Cyclone 10	AD+PT(1536 Bytes)	2,056 LEs	99.6 Mbps
		AD+PT(64 Bytes)		74.0 Mbps
		AD+PT(16 Bytes)		47.5 Mbps
	ECP5	AD+PT(1536 Bytes)	2,368 LUTs	59.5 Mbps
		AD+PT(64 Bytes)		44.3 Mbps
		AD+PT(16 Bytes)		28.4 Mbps
Elephant-v2	Artix-7	AD+PT(Long)	1,884 LUTs	864.5 Mbps
		Hash(Long)	-	-
	Cyclone 10	AD+PT(Long)	2,729 LEs	540.4 Mbps
		Hash(Long)	-	-
	ECP5	AD+PT(Long)	3,073 LUTs	408.4 Mbps
		Hash(Long)	-	-
	Artix-7	AD+PT(1536 Bytes)	1,884 LUTs	426.8 Mbps
		AD+PT(64 Bytes)		313.1 Mbps
		AD+PT(16 Bytes)		194.7 Mbps
	Cyclone 10	AD+PT(1536 Bytes)	2,729 LEs	266.8 Mbps
		AD+PT(64 Bytes)		195.7 Mbps
		AD+PT(16 Bytes)		121.7 Mbps
	ECP5	AD+PT(1536 Bytes)	3,073 LUTs	201.6 Mbps
		AD+PT(64 Bytes)		147.9 Mbps
		AD+PT(16 Bytes)		92.0 Mbps
Elephant-v3	Artix-7	AD+PT(Long)	1,717 LUTs	810.1 Mbps
		Hash(Long)	-	-
	Cyclone 10	AD+PT(Long)	2,504 LEs	499.2 Mbps
		Hash(Long)	-	-
	ECP5	AD+PT(Long)	2,901 LUTs	357.9 Mbps
		Hash(Long)	-	-
	Artix-7	AD+PT(1536 Bytes)	1,717 LUTs	400.1 Mbps
		AD+PT(64 Bytes)		294.3 Mbps
		AD+PT(16 Bytes)		184.2 Mbps
	Cyclone 10	AD+PT(1536 Bytes)	2,504 LEs	246.5 Mbps
		AD+PT(64 Bytes)		181.3 Mbps
		AD+PT(16 Bytes)		113.5 Mbps
	ECP5	AD+PT(1536 Bytes)	2,901 LUTs	176.7 Mbps
		AD+PT(64 Bytes)		13-
		AD+PT(16 Bytes)		81.4 Mbps

Continued on next page

表 4.2 – continued from previous page

Design	Platform	Data	Area	Throughput
Elephant-v4	Artix-7	AD+PT(Long)	1,901 LUTs	990.1 Mbps
		Hash(Long)	-	-
	Cyclone 10	AD+PT(Long)	3,050 LEs	593.3 Mbps
		Hash(Long)	-	-
	ECP5	AD+PT(Long)	3,157 LUTs	367.6 Mbps
		Hash(Long)	-	-
	Artix-7	AD+PT(1536 Bytes)		483.4 Mbps
		AD+PT(64 Bytes)		308.1 Mbps
		AD+PT(16 Bytes)		149.6 Mbps
	Cyclone 10	AD+PT(1536 Bytes)	3,050 LEs	289.7 Mbps
		AD+PT(64 Bytes)		184.6 Mbps
		AD+PT(16 Bytes)		89.7 Mbps
	ECP5	AD+PT(1536 Bytes)	3,157 LUTs	179.5 Mbps
		AD+PT(64 Bytes)		114.4 Mbps
		AD+PT(16 Bytes)		55.5 Mbps
Elephant-v5	Artix-7	AD+PT(Long)	2,645 LUTs	1,543.1 Mbps
		Hash(Long)	-	-
	Cyclone 10	AD+PT(Long)	3,926 LEs	902.3 Mbps
		Hash(Long)	-	-
	ECP5	AD+PT(Long)	4,145 LUTs	640.6 Mbps
		Hash(Long)	-	-
	Artix-7	AD+PT(1536 Bytes)	2,645 LUTs	752.2 Mbps
		AD+PT(64 Bytes)		468.8 Mbps
		AD+PT(16 Bytes)		222.2 Mbps
	Cyclone 10	AD+PT(1536 Bytes)	3,926 LEs	439.8 Mbps
		AD+PT(64 Bytes)		274.1 Mbps
		AD+PT(16 Bytes)		129.9 Mbps
	ECP5	AD+PT(1536 Bytes)	4,145 LUTs	312.3 Mbps
		AD+PT(64 Bytes)		194.6 Mbps
		AD+PT(16 Bytes)		92.3 Mbps
LWC HW APIを用いている．認証暗号の性能は暗号化時のものである．				

Artix-7上で，2,645 LUTの面積コストで最大1.5 Gbps程度のスループット性能を達成している．論文での報告数は少ないが，ASCONと同様，面積コストとハードウェア性能のトレードオフが模索できる柔軟性を有しているといえる．

4.2.2 コプロセッサ（ASIC実装）

文献 [9]でのElephantのASIC実装は以下のとおりである． Elsadekらは，GF 22 nm CMOS

(GF22FDx)で合成したASIC実装について，スループット性能とエネルギー消費について報告している．インタフェース回路は考慮されておらず，コアのみの評価となっている．表4.3にその結果をまとめる．他の候補と比べて，比較的高いスループット性能を有していることがわかる．また，エネルギー効率の評価結果も良い．

表4.3: ElsadekらによるElephantのASIC実装の結果 [9]

Design	I/F	Area	Data	Throughput @f_{max}	Energy Efficiency
Elephant (Enc.)	no	17.3 kGE	Short/PT	14.2 Mbps	74.8 Mbit/mJ
			Short/PT	14.2 Mbps	79.7 Mbit/mJ
			Short/PT+AD	24.0 Mbps	133.5 Mbit/mJ
Elephant (Enc.)	no	17.3 kGE	Long/PT	52.2 Mbps	258.2 Mbit/mJ
			Long/AD	52.2 Mbps	460.2 Mbit/mJ
			Long/PT+AD	70.9 Mbps	341.0 Mbit/mJ

ライブラリ：GF 22 nm CMOS(GF22FDx)，インタフェース回路含まず
Short：16バイトデータを長い時間を空けて処理，Long：1536バイトのデータを処理.

4.3 命令拡張

表4.4は，ChengらによるRISC-Vの命令拡張におけるElephantの実装結果である [7]．命令拡張により，240倍を超える高速化が達成できていることがわかる．別の見方をすれば，既存の命令セットでElephantを高速化に処理することが難しいともいえる．

表4.4: Chenらの命令拡張によるElephantの実装結果 [7]

Design	Area	Cycles/Byte
Elephant RV32GC (Enc.)	3,303 LUTs	125,343.8
Elephant RV32GC+Zbkb/x (Enc.)	3,764 LUTs	3,137.1
Elephant RV32GC+Zbkb/x+ν_0^{32} (Enc.)	3,938 LUTs	508.7
Elephant RV32GC (Dec.)	3,303 LUTs	125,344.3
Elephant RV32GC+Zbkb/x (Dec.)	3,764 LUTs	3,146.8
Elephant RV32GC+Zbkb/x+ν_0^{32} (Dec.)	3,938 LUTs	508.4

入力データ長は128バイトである.

表4.5: ElephantのArm Cortex-M0のレイテンシ評価 [12]

Design	0 to 32Bytes Encryption [msec]	0 to 32 Bytes Decryption [msec]	ROM [kByte]	Code [kByte]
elephant200v1	111095 (38.39)	11095 (38.39)	17.0	14.9
elephant160v1	309186 (1069)	309187 (1069)	16.4	14.4
elephant176v1	36,2768 (1,255)	36,2769 (1,255)	16.4	14.4

RAMの使用量はコンパイル時の静的解析で約1 kByte程度，動作周波数は48MHz
レイテンシは289通りの入力データでの値でカッコ内は平均値.

4.4 ソフトウェア実装

文献 [12]では，Elephantについても軽量デバイス(Arm Cortex-M0)に実装し処理時のレイテンシを測定した．表4.5に示す通り処理性能は他の候補と比べて低い．高速化のためには，アクセラレータや命令拡張といったハードウェアのサポートが必要と考える．また，ハードウェア実装における性能評価の結果がよいことから，ELephantはハードウェア実装に向いたアルゴリズムと思われる．

5 GIFT-COFB

5.1 概要

GIFT-COFBの設計者（提出者）は以下のとおりである.

- Subhadeep Banik
- Avik Chakraborti
- Tetsu Iwata
- Kazuhiko Minematsu
- Mridul Nandi
- Thomas Peyrin
- Yu Sasaki
- Siang Meng Sim
- Yosuke Todo

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://www.isical.ac.in/~lightweight/COFB/`
- 仕様：`https://csrc.nist.gov/CSRC/media/Projects/lightweight-cryptography/documents/finalist-round/updated-spec-doc/gift-cofb-spec-final.pdf`

5.2 ハードウェアアクセラレータ

5.2.1 コプロセッサ（FPGA実装）

MohajeraniらによるFPGA実装の報告 [18]を表5.1にまとめる． Artix-7上で，1,000 LUTs程度の軽量実装から3 Gbpsを超える高速実装までさまざまな結果が得られていることがわかる..

表5.1: GIFT-COFBのFPGA実装の結果 [18]

Design	Platform	Data	Area	Throughput
GIFT-COFB-GMU-v1	Artix-7	AD+PT(Long)	1,223 LUTs	821.1 Mbps
		Hash(Long)	-	-
	Cyclone 10	AD+PT(Long)	1,903 LEs	499.0 Mbps
		Hash(Long)	-	-
	ECP5	AD+PT(Long)	2,727 LUTs	332.3 Mbps
		Hash(Long)	-	-
	Artix-7	AD+PT(1536 Bytes)	1,223 LUTs	407.4 Mbps
		AD+PT(64 Bytes)		346.2 Mbps
		AD+PT(16 Bytes)		235.4 Mbps
	Cyclone 10	AD+PT(1536 Bytes)	1,903 LEs	247.6 Mbps
		AD+PT(64 Bytes)		210.4 Mbps
		AD+PT(16 Bytes)		143.1 Mbps

Continued on next page

表 5.1 – continued from previous page

Design	Platform	Data	Area	Throughput
	ECP5	AD+PT(1536 Bytes)	2,727 LUTs	164.9 Mbps
		AD+PT(64 Bytes)		140.1 Mbps
		AD+PT(16 Bytes)		95.3 Mbps
GIFT-COFB-GMU-v2	Artix-7	AD+PT(Long)	1,380 LUTs	1,590.9 Mbps
		Hash(Long)	-	-
	Cyclone 10	AD+PT(Long)	2,111 LEs	954.0 Mbps
		Hash(Long)	-	-
	ECP5	AD+PT(Long)	2,628 LUTs	639.9 Mbps
		Hash(Long)	-	-
	Artix-7	AD+PT(1536 Bytes)	1,380 LUTs	787.4 Mbps
		AD+PT(64 Bytes)		639.4 Mbps
		AD+PT(16 Bytes)		402.5 Mbps
	Cyclone 10	AD+PT(1536 Bytes)	2,111 LEs	472.2 Mbps
		AD+PT(64 Bytes)		383.4 Mbps
		AD+PT(16 Bytes)		241.4 Mbps
	ECP5	AD+PT(1536 Bytes)	2,628 LUTs	316.7 Mbps
		AD+PT(64 Bytes)		257.2 Mbps
		AD+PT(16 Bytes)		161.9 Mbps
GIFT-COFB-GMU-v3	Artix-7	AD+PT(Long)	1,641 LUTs	2,897.5 Mbps
		Hash(Long)	-	-
	Cyclone 10	AD+PT(Long)	2,523 LEs	1,533.7 Mbps
		Hash(Long)	-	-
	ECP5	AD+PT(Long)	3,059 LUTs	869.6 Mbps
		Hash(Long)	-	-
	Artix-7	AD+PT(1536 Bytes)	1,641 LUTs	1,427.8 Mbps
		AD+PT(64 Bytes)		1,071.3 Mbps
		AD+PT(16 Bytes)		601.4 Mbps
	Cyclone 10	AD+PT(1536 Bytes)	2,523 LEs	755.7 Mbps
		AD+PT(64 Bytes)		567.1 Mbps
		AD+PT(16 Bytes)		318.3 Mbps
	ECP5	AD+PT(1536 Bytes)	3,059 LUTs	428.5 Mbps
		AD+PT(64 Bytes)		321.5 Mbps
		AD+PT(16 Bytes)		180.5 Mbps
GIFT-COFB-GMU-v4	Artix-7	AD+PT(Long)	1,730 LUTs	3,029.3 Mbps
		Hash(Long)	-	-
	Cyclone 10	AD+PT(Long)	2,609 LEs	1,575.5 Mbps
		Hash(Long)	-	-
	ECP5	AD+PT(Long)	3,311 LUTs	812.7 Mbps
		Hash(Long)	-	-
	Artix-7	AD+PT(1536 Bytes)	1,730 LUTs	1,489.7 Mbps

Continued on next page

表 5.1 – continued from previous page

Design	Platform	Data	Area	Throughput
		AD+PT(64 Bytes)		1,079.8 Mbps
		AD+PT(16 Bytes)		580.1 Mbps
	Cyclone 10	AD+PT(1536 Bytes)	2,609 LEs	774.8 Mbps
		AD+PT(64 Bytes)		561.6 Mbps
		AD+PT(16 Bytes)		301.7 Mbps
	ECP5	AD+PT(1536 Bytes)	3,311 LUTs	399.6 Mbps
		AD+PT(64 Bytes)		289.7 Mbps
		AD+PT(16 Bytes)		155.6 Mbps
GIFT-COFB-GMU-v5	Artix-7	AD+PT(Long)	2,051 LUTs	2,922.7 Mbps
		Hash(Long)	-	-
	Cyclone 10	AD+PT(Long)	4,828 LEs	1,097.6 Mbps
		Hash(Long)	-	-
	ECP5	AD+PT(Long)	3,821 LUTs	777.9 Mbps
		Hash(Long)	-	-
	Artix-7	AD+PT(1536 Bytes)	2,051 LUTs	1,429.1 Mbps
		AD+PT(64 Bytes)		947.9 Mbps
		AD+PT(16 Bytes)		461.5 Mbps
	Cyclone 10	AD+PT(1536 Bytes)	4,828 LEs	536.7 Mbps
		AD+PT(64 Bytes)		356.0 Mbps
		AD+PT(16 Bytes)		173.3 Mbps
	ECP5	AD+PT(1536 Bytes)	3,821 LUTs	380.4 Mbps
		AD+PT(64 Bytes)		252.3 Mbps
		AD+PT(16 Bytes)		122.8 Mbps
GIFT-COFB-GMU-v6	Artix-7	AD+PT(Long)	2,363 LUTs	2,816.0 Mbps
		Hash(Long)	-	-
	Cyclone 10	AD+PT(Long)	6,630 LEs	951.6 Mbps
		Hash(Long)	-	-
	ECP5	AD+PT(Long)	-	-
		Hash(Long)	-	-
	Artix-7	AD+PT(1536 Bytes)	2,363 LUTs	1,369.5 Mbps
		AD+PT(64 Bytes)		840.6 Mbps
		AD+PT(16 Bytes)		380.5 Mbps
	Cyclone 10	AD+PT(1536 Bytes)	6,630 LEs	462.8 Mbps
		AD+PT(64 Bytes)		284.0 Mbps
		AD+PT(16 Bytes)		128.6 Mbps
	ECP5	AD+PT(1536 Bytes)	-	-
		AD+PT(64 Bytes)	-	-
		AD+PT(16 Bytes)	-	-
GIFT-COFB-VT-v1	Artix-7	AD+PT(Long)	1,041 LUTs	733.3 Mbps
		Hash(Long)	-	-

Continued on next page

表 5.1 – continued from previous page

Design	Platform	Data	Area	Throughput
	Cyclone 10	AD+PT(Long)	1,877 LEs	491.7 Mbps
		Hash(Long)	-	-
	ECP5	AD+PT(Long)	2,214 LUTs	304.8 Mbps
		Hash(Long)	-	-
	Artix-7	AD+PT(1536 Bytes)	1,041 LUTs	364.3 Mbps
		AD+PT(64 Bytes)		317.1 Mbps
		AD+PT(16 Bytes)		225.6 Mbps
	Cyclone 10	AD+PT(1536 Bytes)	1,877 LEs	244.3 Mbps
		AD+PT(64 Bytes)		212.6 Mbps
		AD+PT(16 Bytes)		151.3 Mbps
	ECP5	AD+PT(1536 Bytes)	2,214 LUTs	151.4 Mbps
		AD+PT(64 Bytes)		131.8 Mbps
		AD+PT(16 Bytes)		93.8 Mbps

LWC HW APIを用いている．認証暗号の性能は暗号化時のものである．

5.2.2 コプロセッサ（ASIC実装）

文献 [9]でのGIFT-COFBのASIC実装の結果を表5.2に示す．また，文献 [6]でのGIFT-COFBのASIC実装は表5.3のとおりである． STM 90 nmライブラリで合成した際の面積と，10MHzでの電力とエネルギーを測定している．表には掲載しなかったが，TSMC 28 nmライブラリとhigh-leakage NanGate 45 nmの結果もあわせて紹介している． GIFT-COFB-SER-Sは，ビットシリアル実装においてswap-and-rotate [4, 3]と呼ばれる手法を用いて面積最小を狙ったものであり，GIFT-COFB-SER-Fはパイプライン処理でのアイドルサイクルを無くし高速化したものである．

表5.2: ElsadekらによるGIFT-COFBのASIC実装の結果 [9]

Design	I/F	Area	Data	Throughput @f_{max}	Energy Efficiency
GIFT-COFB (Enc.)	no	8.1 kGE	Short/PT	25.4 Mbps	299.2 Mbit/mJ
			Short/AD	30.5 Mbps	354.0 Mbit/mJ
			Short/PT+AD	50.4 Mbps	587.4 Mbit/mJ
GIFT-COFB (Enc.)	no	8.1 kGE	Long/PT	156.6 Mbps	1,258.9 Mbit/mJ
			Long/AD	156.6 Mbps	1,265.7 Mbit/mJ
			Long/PT+AD	159.6 Mbps	1,278.9 Mbit/mJ

ライブラリ：GF 22 nm CMOS(GF22FDx)，インタフェース回路含まず
Short：16バイトデータを長い時間を空けて処理，Long：1536バイトのデータを処理

表5.3: CaforioらによるGIFT-COFBのハードウェア実装結果 [9]

Design	I/F	Area	Latency Cycles	Power @10 MHz	Energy
GIFT-COFB (Enc.)	custom	3,927 GE	400	156.3 μW	6.254 nJ
GIFT-COFB-SER-S (Enc.)	custom	1,443 GE	54,784	50.11 μW	275.8 nJ
GIFT-COFB-SER-F (Enc.)	custom	1,485 GE	51,328	62.15 μW	311.9 nJ

ライブラリ：STM 90 nm，128ビットADと1,024bitメッセージを処理

5.3 命令拡張

表5.4は，ChengらによるRISC-Vの命令拡張におけるGIFT-COFBの実装結果である [7]．表中の(FS)と(BS)はそれぞれfix-slicingとbit-slicingを意味する．いずれの実装でも2命令拡張により20倍を超える性能向上がえられていることがわかる．

表5.4: Chenらの命令拡張によるGIFT-COFBの実装結果 [7]

Design	Area	Cycles/Byte
GIFT-COFB(BS) RV32GC (Enc.)	3,303 LUTs	5372.0
GIFT-COFB(BS) RV32GC+Zbkb/x (Enc.)	3,764 LUTs	328.5
GIFT-COFB(BS) RV32GC+Zbkb/x+ν_0^{32} (Enc.)	3,906 LUTs	217.0
GIFT-COFB(BS) RV32GC (Dec.)	3,303 LUTs	5371.4
GIFT-COFB(BS) RV32GC+Zbkb/x (Dec.)	3,764 LUTs	328.9
GIFT-COFB(BS) RV32GC+Zbkb/x+ν_0^{32} (Dec.)	3,906 LUTs	217.3
GIFT-COFB(FS) RV32GC (Enc.)	3,303 LUTs	5372.0
GIFT-COFB(FS) RV32GC+Zbkb/x (Enc.)	3,764 LUTs	327.2
GIFT-COFB(FS) RV32GC+Zbkb/x+ν_0^{32} (Enc.)	4,370 LUTs	263.8
GIFT-COFB(FS) RV32GC (Dec.)	3,303 LUTs	5371.4
GIFT-COFB(FS) RV32GC+Zbkb/x (Dec.)	3,764 LUTs	326.2
GIFT-COFB(FS) RV32GC+Zbkb/x+ν_0^{32} (Dec.)	4,370 LUTs	262.8

入力データ長は128バイトである．

5.4 ソフトウェア実装

表5.5: GIFT-COFBのArm Cortex-M0上でのレイテンシ評価 [12]

Design	0 to 32Bytes Encryption [msec]	0 to 32 Bytes Decryption [msec]	ROM [kByte]	Code [kByte]
giftcofb128v1	1806 (6.25)	1807 (6.25)	17.1	15.1

RAMの使用量はコンパイル時の静的解析で約1 kByte程度，動作周波数は48MHz
レイテンシは289通りの入力データでの値でカッコ内は平均値.

文献 [12]での，GIFT-COFBのArm Cortex-M0実装の結果を示す．約6 msecのレイテンシ性能は他の候補と比べて低いとはいえないが，十分実用的な処理性能はあるといえる．

6 Grain128-AEAD

6.1 概要

Grain128-AEADの設計者（提出者）は以下のとおりである．

- Martin Hell
- Thomas Johansson
- Willi Meier
- Jonathan Sönnerup
- Hirotaka Yoshida
- Alexander Maximov

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://grain-128aead.github.io/`
- 仕様：`https://csrc.nist.gov/CSRC/media/Projects/lightweight-cryptography/documents/finalist-round/updated-spec-doc/grain-128aead-spec-final.pdf`

6.2 ハードウェアアクセラレータ

6.2.1 コプロセッサ（FPGA実装）

表6.1は，岡部によるGrain128-AEADのFPGA実装の結果である [27]． FPGAデバイスの性能が比較的低いこともあり，スループット値はそれほど高くはない．また，インタフェース回路部がないためと思われるが，面積コストは小さい値となっている．

表6.1: Grain128-AEADのFPGA実装の結果 [27]

Design	Platform	I/F	Area	Throughput
Grain128-AEAD	Spartan-3	no	161 LUTs	150.2 Mbps
	Spartan-6	no	174 LUTs	196.8 Mbps

なお，Mohajeraniらの報告 [18]にGrain128-AEADは含まれてない．

6.2.2 コプロセッサ（ASIC実装）

文献 [9]でのElsadekらによるGrain128-AEADのASIC実装性能評価の結果は表6.2のとおりである． XoodyakやTinyJambuと比べてエネルギー効率はかなり低い値となっている．これは，スループット性能に起因するものであるため，低い動作周波数を用いて比較した場合では，これほどの差はでないと予想する．

SönnerupらのASIC実装の結果 [21]を表6.3と6.4にまとめる．使用したライブラリは，ST Microelectronics 65 nm (stm065v536)である．表6.3は，低電力実装の結果である．インタフェースはカスタムに最適化したものを使用している． Unroll数を，1 (Unrollなし)から64まで変化させ，合計で7種類を実装している．電力値は，低電力トランジスタHVTを使用し，100 kHzで

表6.2: ElsadekらによるGrain128-AEADのASIC実装の結果 [9]

Design	I/F	Area	Data	Throughput @f_{max}	Energy Efficiency
Grain128-AEAD (Enc.)	no	4.3 kGE	Short/PT	6.0 Mbps	66.5 Mbit/mJ
			Short/AD	6.0 Mbps	70.8 Mbit/mJ
			Short/PT+AD	9.6 Mbps	106.8 Mbit/mJ
Grain128-AEAD (Enc.)	no	4.3 kGE	Long/PT	17.4 Mbps	258.2 Mbit/mJ
			Long/AD	17.4 Mbps	230.1 Mbit/mJ
			Long/PT+AD	17.7 Mbps	255.8 Mbit/mJ

ライブラリ：GF 22 nm CMOS(GF22FDx)，インタフェース回路含まず
Short：16バイトデータを長い時間を空けて処理，Long：1536バイトのデータを処理.

表6.3: SönnerupらのGrain128-AEADのASIC実装(低電力)の結果 [9]

Design	I/F	Unroll	Area	Power
Grain128-AEAD (Enc.)	custom	1	2,337 GE	0.26 μW
		2	2,511 GE	0.30 μW
		4	2,862 GE	0.32 μW
		8	3,594 GE	0.35 μW
		16	5,053 GE	0.39 μW
		32	7,950 GE	0.46 μW
		64	13,800 GE	0.63 μW

表6.4: SönnerupらによるGrain128-AEADのASIC実装（高速）の結果 [9]

Design	I/F	Unroll	Area	Throughput	Power
Grain128-AEAD (Enc.)	custom	1	2,645 GE	1.25 Gbps	0.25 mW@2.5 GHz
		2	2,695 GE	2.32 Gbps	0.23 mW@2.32 GHz
		4	3,199 GE	4.16 Gbps	0.29 mW@2.08 GHz
		8	4,448 GE	8.68 Gbps	0.67 mW@2.17 GHz
		16	7,118 GE	16.64 Gbps	1.55 mW@2.08 GHz
		32	9,206 GE	24.96 Gbps	1.78 mW@1.56 GHz
		64	16,958 GE	33.6 Gbps	2.76 mW@1.05 GHz

動作させた時の数値である．最も電力の低いもので0.26 μWである．使用したライブラリにより電力値は大きく変わるが，インタフェースを含めた値としては極めて小さいと考える．

表6.4は，高速実装の結果である．アンロール数が大きくなるほど，面積コストと電力は大きくなるが，スループット性能も向上することがわかる．アンロール数が64の場合で，30 Gbpsを超える結果となっている．この時，8,000バイトのデータを約5 nJのエネルギーで処理できるとしている．低電力実装も高速実装も，合成後のシミュレーションの結果であるため，特に電力とエネルギー効率に関しては，実チップでの検証が必要と思われる．

6.3 命令拡張

表6.5は，ChengらによるRISC-Vの命令拡張におけるGrain128-AEADの実装性能評価の結果である [7]．拡張した命令がGrain128-AEADの処理の短縮に繋がらなかったため，スループット

性能の向上は他のアルゴリズムと比べて小さい.

表6.5: Chenらの命令拡張によるGrain128-AEADの実装結果 [7]

Design	Area	Cycles/Byte
Grain128-AEADv2 RV32GC (Enc.)	3,303 LUTs	685.0
Grain128-AEADv2 RV32GC+Zbkb/x (Enc.)	3,764 LUTs	670.5
Grain128-AEADv2 RV32GC+Zbkb/x+ν_0^{32} (Enc.)	4,271 LUTs	500.6
Grain128-AEADv2 RV32GC (Dec.)	3,303 LUTs	677.0
Grain128-AEADv2 RV32GC+Zbkb/x (Dec.)	3,764 LUTs	663.3
Grain128-AEADv2 RV32GC+Zbkb/x+ν_0^{32} (Dec.)	4,271 LUTs	493.3

入力データ長は128バイトである.

6.4 ソフトウェア実装

文献 [12]での，Grain128-AEADのArm Cortex-M0実装の結果を表6.6に示す．また，文献[25]で報告されているAVR ATmega 128とArm Cortex-M3でのソフトウェア実装の結果を表6.7に示す．表6.6の結果と比べて，レイテンシ値は低くなっているが，これはアルゴリズムとプラットフォームを考慮した最適実装がなされたためと思われる.

表6.6: Grain128-AEADをArm Cortex-M0上でのレイテンシ評価 [12]

Design	0 to 32Bytes Encryption [msec]	0 to 32 Bytes Decryption [msec]	ROM [kByte]	Code [kByte]
grain128aead	23854 (82.54)	23,831 (82.46)	17.8	15.8

RAMの使用量はコンパイル時の静的解析で約1 kByte程度，動作周波数は48MHz
レイテンシは289通りの入力データでの値でカッコ内は平均値.

表6.7: Grain-128AEADv2のArm Cortex-M3とAVR ATmegaのレイテンシ評価 [25]

Design	Platform	16-byte AD + 16-byte Msg. [msec]	ROM [kByte]	RAM [Byte]
Grain-128AEADv2 (Enc.)	AVR ATmega	7.81	6,098	145
Grain-128AEADv2 (Dec.)	@16 Mhz	7.83		147
Grain-128AEADv2 (Enc.)	Arm Cortex-M3	0.59	6,680	224
Grain-128AEADv2 (Dec.)	@84 MHz	0.59		232

7 ISAP

7.1 概要

ISAPの設計者（提出者）は以下のとおりである.

- Christoph Dobraunig
- Maria Eichlseder
- Stefan Mangard
- Florian Mendel
- Bart Mennink
- Robert Primas
- Thomas Unterluggauer

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

7.2 ハードウェアアクセラレータ

7.2.1 コプロセッサ（FPGA実装）

MohajeraniらのFPGA実装の結果 [18]を表7. 1 にまとめる．最大で約830 Mbpsのスループット性能が得られていることがわかる．面積コストと処理性能は他のアルゴリズムと比べ良い結果とはなっていない.

表7.1: ISAPのFPGA実装の結果 [18]

Design	Platform	Data	Area	Throughput
ISAP-v1	Artix-7	AD+PT(Long)	3,491 LUTs	829.6 Mbps
		Hash(Long)	-	-
	Cyclone 10	AD+PT(Long)	4,589 LEs	544.2 Mbps
		Hash(Long)	-	-
	ECP5	AD+PT(Long)	6,701 LUTs	262.6 Mbps
		Hash(Long)	-	-
	Artix-7	AD+PT(1536 Bytes)	3,491 LUTs	389.4 Mbps
		AD+PT(64 Bytes)		163.9 Mbps
		AD+PT(16 Bytes)		60.1 Mbps
	Cyclone 10	AD+PT(1536 Bytes)	4,589 LEs	255.4 Mbps
		AD+PT(64 Bytes)		107.5 Mbps
		AD+PT(16 Bytes)		39.4 Mbps

Continued on next page

表 7.1 – continued from previous page

Design	Platform	Data	Area	Throughput
	ECP5	AD+PT(1536 Bytes)	6,701 LUTs	123.3 Mbps
		AD+PT(64 Bytes)		51.9 Mbps
		AD+PT(16 Bytes)		19.0 Mbps
ISAP-v2	Artix-7	AD+PT(Long)	2,157 LUTs	609.5 Mbps
		Hash(Long)	-	-
	Cyclone 10	AD+PT(Long)	3,852 LEs	415.7 Mbps
		Hash(Long)	-	-
	ECP5	AD+PT(Long)	5,708 LUTs	207.1 Mbps
		Hash(Long)	-	-
	Artix-7	AD+PT(1536 Bytes)	2,157 LUTs	290.6 Mbps
		AD+PT(64 Bytes)		140.7 Mbps
		AD+PT(16 Bytes)		53.8 Mbps
	Cyclone 10	AD+PT(1536 Bytes)	3,852 LEs	198.2 Mbps
		AD+PT(64 Bytes)		95.9 Mbps
		AD+PT(16 Bytes)		36.7 Mbps
	ECP5	AD+PT(1536 Bytes)	5,708 LUTs	98.8 Mbps
		AD+PT(64 Bytes)		47.8 Mbps
		AD+PT(16 Bytes)		18.3 Mbps
ISAP-v3	Artix-7	AD+PT(Long)	2,182 LUTs	808.1 Mbps
		Hash(Long)	-	-
	Cyclone 10	AD+PT(Long)	3,767 LEs	567.0 Mbps
		Hash(Long)	-	-
	ECP5	AD+PT(Long)	5,703 LUTs	282.2 Mbps
		Hash(Long)	-	-
	Artix-7	AD+PT(1536 Bytes)	2,182 LUTs	378.5 Mbps
		AD+PT(64 Bytes)		156.3 Mbps
		AD+PT(16 Bytes)		56.8 Mbps
	Cyclone 10	AD+PT(1536 Bytes)	3,767 LEs	265.5 Mbps
		AD+PT(64 Bytes)		109.6 Mbps
		AD+PT(16 Bytes)		39.8 Mbps
	ECP5	AD+PT(1536 Bytes)	5,703 LUTs	132.1 Mbps
		AD+PT(64 Bytes)		54.6 Mbps
		AD+PT(16 Bytes)		19.8 Mbps
ISAP-v4	Artix-7	AD+PT(Long)	-	-
		Hash(Long)	-	-
	Cyclone 10	AD+PT(Long)	3,026 LEs	551.1 Mbps
		Hash(Long)	-	-
	ECP5	AD+PT(Long)	3,623 LUTs	238.9 Mbps
		Hash(Long)	-	-
	Artix-7	AD+PT(1536 Bytes)	-	-

Continued on next page

表 7.1 – continued from previous page

Design	Platform	Data	Area	Throughput
		AD+PT(64 Bytes)	-	-
		AD+PT(16 Bytes)	-	-
	Cyclone 10	AD+PT(1536 Bytes)	3,026 LEs	261.3 Mbps
		AD+PT(64 Bytes)		119.3 Mbps
		AD+PT(16 Bytes)		44.2 Mbps
	ECP5	AD+PT(1536 Bytes)	3,623 LUTs	113.3 Mbps
		AD+PT(64 Bytes)		51.7 Mbps
		AD+PT(16 Bytes)		19.2 Mbps
LWC HW APIを用いている．認証暗号の性能は暗号化時のものである．				

7.2.2 コプロセッサ（ASIC実装）

文献 [9]でのISAPのASIC実装の結果は表7.2のとおりである．スループット性能が高くないため，エネルギー効率もファイナリストの中では高くない．

表7.2: ElsadekらによるISAPのASIC実装の結果 [9]

Design	I/F	Area	Data	Throughput @f_{max}	Energy Efficiency
ISAP (Enc.)	no	15.4 kGE	Short/PT	12.5 Mbps	49.9 Mbit/mJ
			Short/AD	18.5 Mbps	70.8 Mbit/mJ
			Short/PT+AD	23.2 Mbps	93.5 Mbit/mJ
ISAP (Enc.)	no	15.4 kGE	Long/PT	156.6 Mbps	548.8 Mbit/mJ
			Long/AD	156.6 Mbps	863.0 Mbit/mJ
			Long/PT+AD	195.0 Mbps	682.1 Mbit/mJ

ライブラリ：GF 22 nm CMOS(GF22FDx)，インタフェース回路含まず
Short：16バイトデータを長い時間を空けて処理，Long：1536バイトのデータを処理.

7.3 命令拡張

SteineggerとPrimasによるRISC-Vの命令拡張としてのASCON-*p*命令の実装を報告している [23]は，ISAP-A-128AとISAP-A-128に適用できる．表7.3は，その実装性能評価をまとめたものである．入力データ長により異なるが，命令拡張により40倍から80倍程度の性能向上が得られていることがわかる．また，命令拡張後のISAP-A-128a-ASM + Acc.において，64バイトのデータ処理の場合は，Longの場合と比べて 1 バイトあたりに必要なサイクル数が約7倍となっている．入力データ長が短くなるようなアプリケーションでは命令拡張による効果は低いといえる．

表7.3: SteineggerとPrimasによるASCON-p命令拡張によるISAPの実装結果 [23]

Design	Area overhead	Cycles/Byte			Binary size
		64 Byte	1,536 Byte	Long	
ISAP-A-128a-C (-O3)	-	1,184.3	386.9	352.3	11,052 Byte
ISAP-A-128a-C (-Os)	-	2,024.1	616.0	554.8	3,744 Byte
ISAP-A-128a-ASM + Acc.	4.7 kGE	29.1	5.2	4.2	1,844 Byte
ISAP-A-128-ASM + Acc.	4.7 kGE	73.6	7.7	5.0	2,522 Byte

7.4 ソフトウェア実装

文献 [12]でのISAPのArm Cortex-M0上の実装結果を示す.

表7.4: ISAPのArm Cortex-M0上でのレイテンシ評価 [12]

Design	0 to 32Bytes Encryption [msec]	0 to 32 Bytes Decryption [msec]	ROM [kByte]	Code [kByte]
isapa128av20	2,791 (9.66)	2,793 (9.66)	16.5	14.5
isapa128v20	11,414 (39.49)	11,416 (39.50)	16.5	14.5
isapk128av20	46794 (161.9)	46796 (161.9)	16.6	14.5
isapk128v20	39,4973 (1,366)	39,4975 (1,366)	16.6	14.5

RAMの使用量はコンパイル時の静的解析で約1 kByte程度，動作周波数は48MHz
レイテンシは289通りの入力データでの値でカッコ内は平均値.

8 PHOTON-Beetle

8.1 概要

PHOTON-Beetleの設計者（提出者）は以下のとおりである.

- Zhenzhen Bao
- Avik Chakraborti
- Nilanjan Datta
- Jian Guo
- Mridul Nandi
- Thomas Peyrin
- Kan Yasuda

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://www.isical.ac.in/~lightweight/beetle/`
- 仕様：`https://csrc.nist.gov/CSRC/media/Projects/lightweight-cryptography/documents/finalist-round/updated-spec-doc/photon-beetle-spec-final.pdf`

8.2 ハードウェアアクセラレータ

8.2.1 コプロセッサ（FPGA実装）

MohajeraniらのFPGA実装の結果 [18]を表8.1にまとめる.

表8.1: PHOTON-BeetleのFPGA実装の結果 [18]

Design	Platform	Data	Area	Throughput
PHOTON-Beetle-v1	Artix-7	AD+PT(Long)	2,065 LUTs	747.0 Mbps
		Hash(Long)		227.8 Mbps
	Cyclone 10	AD+PT(Long)	3,602 LEs	526.4 Mbps
		Hash(Long)		160.6 Mbps
	ECP5	AD+PT(Long)	3,294 LUTs	425.7 Mbps
		Hash(Long)		129.8 Mbps
	Artix-7	AD+PT(1536 Bytes)	2,065 LUTs	370.4 Mbps
		AD+PT(64 Bytes)		311.0 Mbps
		AD+PT(16 Bytes)		207.1 Mbps
		Hash(1536 Bytes)		228.6 Mbps
		Hash(64 Bytes)		249.0 Mbps
		Hash(16 Bytes)		345.2 Mbps
	Cyclone 10	AD+PT(1536 Bytes)	3,602 LEs	261.0 Mbps
		AD+PT(64 Bytes)		219.2 Mbps

Continued on next page

表 8.1 – continued from previous page

Design	Platform	Data	Area	Throughput
		AD+PT(16 Bytes)		146.0 Mbps
		Hash(1536 Bytes)		161.1 Mbps
		Hash(64 Bytes)		175.5 Mbps
		Hash(16 Bytes)		243.3 Mbps
	ECP5	AD+PT(1536 Bytes)	3,294 LUTs	211.1 Mbps
		AD+PT(64 Bytes)		177.3 Mbps
		AD+PT(16 Bytes)		118.0 Mbps
		Hash(1536 Bytes)		130.3 Mbps
		Hash(64 Bytes)		141.9 Mbps
		Hash(16 Bytes)		196.7 Mbps
LWC HW APIを用いている．認証暗号の性能は暗号化時のものである．				

8.2.2 コプロセッサ（ASIC実装）

文献 [9]でのPHOTON-BeetleのASIC実装は表8.2のとおりである．

表8.2: ElsadekらによるPHOTON-BeetleのASIC実装の結果 [9]

Design	I/F	Area	Data	Throughput @f_{max}	Energy Efficiency
PHOTON-Beetle (Enc.)	no	12.6 kGE	Short/PT	42.6 Mbps	265.9 Mbit/mJ
			Short/AD	42.6 Mbps	265.5 Mbit/mJ
			Short/PT+AD	78.4 Mbps	413.9 Mbit/mJ
PHOTON-Beetle (Enc.)	no	12.6 kGE	Long/PT	522.0 Mbps	1,065.2 Mbit/mJ
			Long/AD	522.0 Mbps	1,035.5 Mbit/mJ
			Long/PT+AD	531.9 Mbps	1,065.8 Mbit/mJ

ライブラリ：GF 22 nm CMOS(GF22FDx)，インタフェース回路含まず
Short：16バイトデータを長い時間を空けて処理，Long：1536バイトのデータを処理．

8.3 命令拡張

表8.3はChengらによるRISC-Vの命令拡張におけるPHOTON-Beetleの実装結果である [7]．命令拡張による処理性能の向上はあまりないことがわかる．

8.4 ソフトウェア実装

表8.4に文献 [12]でのPHOTON-BeetleのArm Cortex-M0上の実装結果を示す．本評価の中では，レイテンシ性能は悪い方であるが，最適化により改善されると思われる．

表8.3: Chenらの命令拡張によるPHOTON-Beetleの実装結果 [7]

Design	Area	Cycles/Byte
PHOTON-Beetle RV32GC (Enc.)	3,303 LUTs	685.0
PHOTON-Beetle RV32GC+Zbkb/x (Enc.)	3,764 LUTs	670.5
PHOTON-Beetle RV32GC+Zbkb/x+ν_0^{32} (Enc.)	4,271 LUTs	500.6
PHOTON-Beetle RV32GC (Dec.)	3,303 LUTs	677.0
PHOTON-Beetle RV32GC+Zbkb/x (Dec.)	3,764 LUTs	663.3
PHOTON-Beetle RV32GC+Zbkb/x+ν_0^{32} (Dec.)	4,271 LUTs	493.3

入力データ長は128バイトである.

表8.4: PHOTON-BeetleのArm Cortex-M0上でのレイテンシ評価 [12]

Design	0 to 32Bytes Encryption [msec]	0 to 32 Bytes Decryption [msec]	ROM [kByte]	Code [kByte]
photonbeetleaead128rate128v1	12,251 (42.39)	12,253 (42.40)	17.5	15.4
photonbeetleaead128rate32v1	29,655 (102.6)	29,658 (102.6)	17.6	15.5

RAMの使用量はコンパイル時の静的なメモリサイズで約1 kByte程度である.

9 Romulus

9.1 概要

Romulusの設計者（提出者）は以下のとおりである.

- Tetsu Iwata
- Mustafa Khairallah
- Kazuhiko Minematsu
- Thomas Peyrin
- Chun Guo

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://romulusae.github.io/romulus/`
- 仕様：`https://csrc.nist.gov/CSRC/media/Projects/lightweight-cryptography/documents/finalist-round/updated-spec-doc/romulus-spec-final.pdf`

9.2 ハードウェアアクセラレータ

9.2.1 コプロセッサ（FPGA実装）

MohajeraniらのFPGA実装の結果 [18]を表9.1にまとめる.

表9.1: RomulusのFPGA実装の結果 [18]

Design	Platform	Data	Area	Throughput
Romulus-v1	Artix-7	AD+PT(Long)	953 LUTs	637.2 Mbps
		Hash(Long)	-	-
	Cyclone 10	AD+PT(Long)	1,735 LEs	398.6 Mbps
		Hash(Long)	-	-
	ECP5	AD+PT(Long)	1,998 LUTs	224.0 Mbps
		Hash(Long)	-	-
	Artix-7	AD+PT(1536 Bytes)	953 LUTs	315.7 Mbps
		AD+PT(64 Bytes)		261.7 Mbps
		AD+PT(16 Bytes)		209.4 Mbps
	Cyclone 10	AD+PT(1536 Bytes)	1,735 LEs	197.5 Mbps
		AD+PT(64 Bytes)		163.7 Mbps
		AD+PT(16 Bytes)		131.0 Mbps
	ECP5	AD+PT(1536 Bytes)	1,998 LUTs	111.0 Mbps
		AD+PT(64 Bytes)		92.0 Mbps

Continued on next page

表 9.1 – continued from previous page

Design	Platform	Data	Area	Throughput
		AD+PT(16 Bytes)		73.6 Mbps
Romulus-v2	Artix-7	AD+PT(Long)	1,280 LUTs	1,095.7 Mbps
		Hash(Long)	-	-
	Cyclone 10	AD+PT(Long)	2,086 LEs	725.5 Mbps
		Hash(Long)	-	-
	ECP5	AD+PT(Long)	2,353 LUTs	419.8 Mbps
		Hash(Long)	-	-
	Artix-7	AD+PT(1536 Bytes)	1,280 LUTs	542.0 Mbps
		AD+PT(64 Bytes)		434.8 Mbps
		AD+PT(16 Bytes)		326.1 Mbps
	Cyclone 10	AD+PT(1536 Bytes)	2,086 LEs	358.9 Mbps
		AD+PT(64 Bytes)		287.9 Mbps
		AD+PT(16 Bytes)		215.9 Mbps
	ECP5	AD+PT(1536 Bytes)	2,353 LUTs	207.7 Mbps
		AD+PT(64 Bytes)		166.6 Mbps
		AD+PT(16 Bytes)		125.0 Mbps
Romulus-v3	Artix-7	AD+PT(Long)	1,824 LUTs	1,085.8 Mbps
		Hash(Long)	-	-
	Cyclone 10	AD+PT(Long)	2,407 LEs	70-
		Hash(Long)	-	-
	ECP5	AD+PT(Long)	3,847 LUTs	397.2 Mbps
		Hash(Long)	-	-
	Artix-7	AD+PT(1536 Bytes)	1,824 LUTs	535.6 Mbps
		AD+PT(64 Bytes)		408.9 Mbps
		AD+PT(16 Bytes)		281.1 Mbps
	Cyclone 10	AD+PT(1536 Bytes)	2,407 LEs	345.3 Mbps
		AD+PT(64 Bytes)		263.6 Mbps
		AD+PT(16 Bytes)		181.3 Mbps
	ECP5	AD+PT(1536 Bytes)	3,847 LUTs	195.9 Mbps
		AD+PT(64 Bytes)		149.6 Mbps
		AD+PT(16 Bytes)		102.9 Mbps
Romulus-v4	Artix-7	AD+PT(Long)	2,602 LUTs	802.6 Mbps
		Hash(Long)	-	-
	Cyclone 10	AD+PT(Long)	3,409 LEs	558.5 Mbps
		Hash(Long)	-	-
	ECP5	AD+PT(Long)	5,086 LUTs	298.9 Mbps
		Hash(Long)	-	-
	Artix-7	AD+PT(1536 Bytes)	2,602 LUTs	394.4 Mbps
		AD+PT(64 Bytes)		282.8 Mbps
		AD+PT(16 Bytes)		176.8 Mbps

Continued on next page

表 9.1 – continued from previous page

Design	Platform	Data	Area	Throughput
	Cyclone 10	AD+PT(1536 Bytes)	3,409 LEs	274.5 Mbps
		AD+PT(64 Bytes)		196.8 Mbps
		AD+PT(16 Bytes)		123.0 Mbps
	ECP5	AD+PT(1536 Bytes)	5,086 LUTs	146.9 Mbps
		AD+PT(64 Bytes)		105.3 Mbps
		AD+PT(16 Bytes)		65.8 Mbps
Romulus-v5	Artix-7	AD+PT(Long)	887 LUTs	27.9 Mbps
		Hash(Long)	-	-
	Cyclone 10	AD+PT(Long)	1,960 LEs	17.0 Mbps
		Hash(Long)	-	-
	ECP5	AD+PT(Long)	1,961 LUTs	1-
		Hash(Long)	-	-
	Artix-7	AD+PT(1536 Bytes)	887 LUTs	13.8 Mbps
		AD+PT(64 Bytes)		11.8 Mbps
		AD+PT(16 Bytes)		10.2 Mbps
	Cyclone 10	AD+PT(1536 Bytes)	1,960 LEs	8.4 Mbps
		AD+PT(64 Bytes)		7.2 Mbps
		AD+PT(16 Bytes)		6.2 Mbps
	ECP5	AD+PT(1536 Bytes)	1,961 LUTs	4.9 Mbps
		AD+PT(64 Bytes)		4.2 Mbps
		AD+PT(16 Bytes)		3.6 Mbps
LWC HW APIを用いている．認証暗号の性能は暗号化時のものである．				

9.2.2 コプロセッサ（ASIC実装）

文献 [9]でのPHOTON-BeetleのASIC実装は表9.2のとおりである．

表9.2: ElsadekらによるRomulusのASIC実装の結果 [9]

Design	I/F	Area	Data	Throughput @f_{max}	Energy Efficiency
Romulus (Enc.)	no	9.8 kGE	Short/PT	31.0 Mbps	257.6 Mbit/mJ
			Short/AD	31.0 Mbps	256.7 Mbit/mJ
			Short/PT+AD	61.6 Mbps	520.7 Mbit/mJ
Romulus (Enc.)	no	9.8 kGE	Long/PT	156.6 Mbps	1,033.0 Mbit/mJ
			Long/AD	156.6 Mbps	1,956.0 Mbit/mJ
			Long/PT+AD	212.8 Mbps	1,364.2 Mbit/mJ

ライブラリ：GF 22 nm CMOS (GF22FDx)，インタフェース回路含まず
Short：16バイトデータを長い時間を空けて処理，Long：1536バイトのデータを処理．

文献 [6]でのRomulusのASIC実装の結果は表9.3のとおりである．

表9.3: CaforioらによるRomulusのASIC実装の結果 [9]

Design	I/F	Area	Latency Cycles	Power @10 MHz	Energy
Romulus (Enc.)	custom	1,778 GE	55,431	82.28 μW	456.1 nJ

ライブラリ：STM 90 nm，128ビットADと1,024bitメッセージを処理

9.3 命令拡張

表9.4はChengらによるRISC-Vの命令拡張におけるRomulusの実装結果である [7]．表中のFSとTBは，それぞれfix-slicingとlook-up tablesを意味する．命令拡張により一定の高速化が達成できているといえる．特にTB実装では，30倍程度の高速化が得られている．

表9.4: Chenらの命令拡張によるRomulusの実装結果 [7]

Design	Area	Cycles/Byte
Romulus(TB) RV32GC (Enc.)	3,303 LUTs	7956.0
Romulus(TB) RV32GC+Zbkb/x (Enc.)	3,764 LUTs	1665.5
Romulus(TB) RV32GC+Zbkb/x+ν_0^{32} (Enc.)	3.998 LUTs	256.9
Romulus(TB) RV32GC (Dec.)	3,303 LUTs	7953.0
Romulus(TB) RV32GC+Zbkb/x (Dec.)	3,764 LUTs	1667.5
Romulus(TB) RV32GC+Zbkb/x+ν_0^{32} (Dec.)	3.998 LUTs	258.2
Romulus(FS) RV32GC (Enc.)	3,303 LUTs	1383.1
Romulus(FS) RV32GC+Zbkb/x (Enc.)	3,764 LUTs	1589.7
Romulus(FS) RV32GC+Zbkb/x+ν_0^{32} (Enc.)	4,205 LUTs	315.2
Romulus(FS) RV32GC (Dec.)	3,303 LUTs	1385.4
Romulus(FS) RV32GC+Zbkb/x (Dec.)	3,764 LUTs	1589.4
Romulus(FS) RV32GC+Zbkb/x+ν_0^{32} (Dec.)	4,205 LUTs	322.3

入力データ長は128バイトである．

9.4 ソフトウェア実装

表9.5に文献 [12]でのRomulusのArm Cortex-M0上の実装結果を示す． 10 msecを超えるレイテンシであり，他の候補と比べて悪い結果である．さらなる最適化の余地があるか検討が必要と考える．

表9.5: RomulusをArm Cortex-M0上でのレイテンシ評価 [12]

Design	0 to 32Bytes Encryption [msec]	0 to 32 Bytes Decryption [msec]	ROM [kByte]	Code [kByte]
romulusn3v12	3,214 (11.12)	3,216 (11.13)	19.5	16.9
romulusm3v12	4,185 (14.48)	4,188 (14.49)	19.7	17.1
romulusn2v12	4,943 (17.10)	4,944 (17.11)	19.7	17.1
romulusn1v12	4,959 (17.16)	4,962 (17.17)	19.6	17.0
romulusm1v12	6,126 (21.20)	6,128 (21.20)	19.5	17.0
romulusm2v12	6,376 (22.06)	6,378 (22.07)	19.8	17.2

RAMの使用量はコンパイル時の静的解析で約1 kByte程度，動作周波数は48MHz
レイテンシは289通りの入力データでの値でカッコ内は平均値.

10 SPARKLE

10.1 概要

SPARKLEの設計者（提出者）は以下のとおりである.

- Christof Beierle
- Alex Biryukov
- Luan Cardoso dos Santos
- Johann Großschädl
- Léo Perrin
- Aleksei Udovenko
- Vesselin Velichkov
- Qingju Wang
- Amir Moradi
- Aein Rezaei Shahmirzadi

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://sparkle-lwc.github.io/`
- 仕様：`https://csrc.nist.gov/CSRC/media/Projects/lightweight-cryptography/documents/finalist-round/updated-spec-doc/sparkle-spec-final.pdf`

10.2 ハードウェアアクセラレータ

10.2.1 コプロセッサ（FPGA実装）

MohajeraniらのFPGA実装の結果 [18]を表10.1にまとめる.

表10.1: SPARKLEのFPGA実装の結果 [18]

Design	Platform	Data	Area	Throughput
SCHWAEMM-v1	Artix-7	AD+PT(Long)	3,071 LUTs	813.2 Mbps
		Hash(Long)	-	-
	Cyclone 10	AD+PT(Long)	4,713 LEs	492.4 Mbps
		Hash(Long)	-	-
	ECP5	AD+PT(Long)	4,685 LUTs	399.6 Mbps
		Hash(Long)	-	-
	Artix-7	AD+PT(1536 Bytes)	3,071 LUTs	396.8 Mbps
		AD+PT(64 Bytes)		255.1 Mbps
		AD+PT(16 Bytes)		94.9 Mbps
	Cyclone 10	AD+PT(1536 Bytes)	4,713 LEs	240.3 Mbps
		AD+PT(64 Bytes)		154.5 Mbps
		AD+PT(16 Bytes)		57.5 Mbps
	ECP5	AD+PT(1536 Bytes)	4,685 LUTs	195.0 Mbps

Continued on next page

表 10.1 – continued from previous page

Design	Platform	Data	Area	Throughput
		AD+PT(64 Bytes)		125.3 Mbps
		AD+PT(16 Bytes)		46.7 Mbps
SCHWAEMM-v2	Artix-7	AD+PT(Long)	3,740 LUTs	783.1 Mbps
		Hash(Long)		489.4 Mbps
	Cyclone 10	AD+PT(Long)	5,773 LEs	516.5 Mbps
		Hash(Long)		322.8 Mbps
	ECP5	AD+PT(Long)	5,947 LUTs	384.2 Mbps
		Hash(Long)		240.1 Mbps
	Artix-7	AD+PT(1536 Bytes)	3,740 LUTs	382.1 Mbps
		AD+PT(64 Bytes)		245.6 Mbps
		AD+PT(16 Bytes)		91.4 Mbps
		Hash(1536 Bytes)		481.2 Mbps
		Hash(64 Bytes)		346.7 Mbps
		Hash(16 Bytes)		184.9 Mbps
	Cyclone 10	AD+PT(1536 Bytes)	5,773 LEs	252.0 Mbps
		AD+PT(64 Bytes)		162.0 Mbps
		AD+PT(16 Bytes)		60.3 Mbps
		Hash(1536 Bytes)		317.3 Mbps
		Hash(64 Bytes)		228.6 Mbps
		Hash(16 Bytes)		121.9 Mbps
	ECP5	AD+PT(1536 Bytes)	5,947 LUTs	187.5 Mbps
		AD+PT(64 Bytes)		120.5 Mbps
		AD+PT(16 Bytes)		44.9 Mbps
		Hash(1536 Bytes)		236.1 Mbps
		Hash(64 Bytes)		170.1 Mbps
		Hash(16 Bytes)		90.7 Mbps

LWC HW APIを用いている．認証暗号の性能は暗号化時のものである．

10.2.2 コプロセッサ（ASIC実装）

文献 [9]でのSPARKLEのASIC実装は表10.2のとおりである．

10.3 命令拡張

表10.3は，ChengらによるRISC-Vの命令拡張におけるSparkleの実装性能評価の結果である [7].

表10.2: ElsadekらによるSPARKLEのASIC実装の結果 [9]

Design	I/F	Area	Data	Throughput @f_{max}	Energy Efficiency
Sparkle (Enc.)	no	39.5 kGE	Short/PT	43.0 Mbps	141.3 Mbit/mJ
			Short/AD	43.0 Mbps	141.6 Mbit/mJ
			Short/PT+AD	80.0 Mbps	253.7 Mbit/mJ
Sparkle (Enc.)	no	39.5 kGE	Long/PT	1,740.0 Mbps	1,581.7 Mbit/mJ
			Long/AD	1,740.0 Mbps	1,956.0 Mbit/mJ
			Long/PT+AD	1,773.0 Mbps	1,492.1 Mbit/mJ

ライブラリ：GF 22 nm CMOS (GF22FDx)，インタフェース回路含まず
Short：16バイトデータを長い時間を空けて処理，Long：1536バイトのデータを処理.

表10.3: Chenらの命令拡張によるSparkleの実装結果 [7]

Design	Area	Cycles/Byte
Sparkle RV32GC (Enc.)	3,303 LUTs	234.6
Sparkle RV32GC+Zbkb/x (Enc.)	3,764 LUTs	100.6
Sparkle RV32GC+Zbkb/x+ν_0^{32} (Enc.)	3.998 LUTs	76.0
Sparkle RV32GC+Zbkb/x+ν_1^{32} (Enc.)	3.986 LUTs	75.9
Sparkle RV32GC+Zbkb/x+ν_2^{32} (Enc.)	3.963 LUTs	40.8
Sparkle RV32GC (Dec.)	3,303 LUTs	234.8
Sparkle RV32GC+Zbkb/x (Dec.)	3,764 LUTs	100.9
Sparkle RV32GC+Zbkb/x+ν_0^{32} (Dec.)	3.998 LUTs	76.2
Sparkle RV32GC+Zbkb/x+ν_1^{32} (Dec.)	3.986 LUTs	76.2
Sparkle RV32GC+Zbkb/x+ν_2^{32} (Dec.)	3.963 LUTs	41.2

入力データ長は128バイトである.

10.4 ソフトウェア実装

表10.4に文献 [12]でのSPARKLEのArm Cortex-M0実装の結果を示す．本評価において，ファイナリスト中では低レイテンシの結果が得られている.

表10.4: SparkleのArm Cortex-M0上でのレイテンシ評価 [12]

Design	0 to 32Bytes Encryption [msec]	0 to 32 Bytes Decryption [msec]	ROM [kByte]	Code [kByte]
schwaemm128128v1	221	222	16.9	14.9
	(0.76)	(0.77)		
schwaemm256128v1	269	269	17.1	15.1
	(0.93)	(0.93)		
schwaemm192192v1	333	313	17	15
	(1.152)	(1.083)		
schwaemm256256v1	353	354	17.2	15.1
	(1.221)	(1.225)		

RAMの使用量はコンパイル時の静的なメモリサイズで約1 kByte程度である.

11 TinyJambu

11.1 概要

TinyJambuの設計者（提出者）は以下のとおりである.

- Hongjun Wu
- Tao Huang

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：n/a
- 仕様：`https://csrc.nist.gov/CSRC/media/Projects/lightweight-cryptography/documents/finalist-round/updated-spec-doc/tinyjambu-spec-final.pdf`

11.2 ハードウェアアクセラレータ

11.2.1 コプロセッサ（FPGA実装）

TinyJambuのFPGA実装については，NIST主催のLightweight Cryptography Workshop 2022でのAbdulgadirらの報告がある [2]． Artix-7上にLWC HW APIを用いた実装で，1536バイトのデータを暗号化した際に250 Mbps程度のスループットとなっている（表11.1）．また，他のFPGA実装の結果として，Mohajeraniらの報告 [18]を表11.2にまとめる．同じ591 LUTsの面積コストで文献 [2]とスループット性能が異なる異なる結果が見られるが，その理由は不明である.

表11.1: TinyJambuのFPGA実装の結果 [2]

Design	Platform	I/F	Area	Throughput
TinyJambu	Artix-7	LWC HW API	591 LUTs	250.4 Mbps

表11.2: TinyJambuのFPGA実装の結果（つづき） [18]

Design	Platform	Data	Area	Throughput
TinyJAMBU-GMU-v1	Artix-7	AD+PT(Long)	591 LUTs	354.7 Mbps
		Hash(Long)	-	-
	Cyclone 10	AD+PT(Long)	856 LEs	262.4 Mbps
		Hash(Long)	-	-
	ECP5	AD+PT(Long)	720 LUTs	166.4 Mbps
		Hash(Long)	-	-

Continued on next page

表 11.2 – continued from previous page

Design	Platform	Data	Area	Throughput
	Artix-7	AD+PT(1536 Bytes)	591 LUTs	176.1 Mbps
		AD+PT(64 Bytes)		151.3 Mbps
		AD+PT(16 Bytes)		105.1 Mbps
	Cyclone 10	AD+PT(1536 Bytes)	856 LEs	130.3 Mbps
		AD+PT(64 Bytes)		112.0 Mbps
		AD+PT(16 Bytes)		77.8 Mbps
	ECP5	AD+PT(1536 Bytes)	720 LUTs	82.6 Mbps
		AD+PT(64 Bytes)		71.0 Mbps
		AD+PT(16 Bytes)		49.3 Mbps
TinyJAMBU-GMU-v2	Artix-7	AD+PT(Long)	564 LUTs	186.4 Mbps
		Hash(Long)	-	-
	Cyclone 10	AD+PT(Long)	841 LEs	136.5 Mbps
		Hash(Long)	-	-
	ECP5	AD+PT(Long)	908 LUTs	89.3 Mbps
		Hash(Long)	-	-
	Artix-7	AD+PT(1536 Bytes)	564 LUTs	92.6 Mbps
		AD+PT(64 Bytes)		80.0 Mbps
		AD+PT(16 Bytes)		56.1 Mbps
	Cyclone 10	AD+PT(1536 Bytes)	841 LEs	67.8 Mbps
		AD+PT(64 Bytes)		58.5 Mbps
		AD+PT(16 Bytes)		41.0 Mbps
	ECP5	AD+PT(1536 Bytes)	908 LUTs	44.3 Mbps
		AD+PT(64 Bytes)		38.3 Mbps
		AD+PT(16 Bytes)		26.8 Mbps
TinyJAMBU-GMU-v3	Artix-7	AD+PT(Long)	537 LUTs	12.6 Mbps
		Hash(Long)	-	-
	Cyclone 10	AD+PT(Long)	817 LEs	8.7 Mbps
		Hash(Long)	-	-
	ECP5	AD+PT(Long)	1,277 LUTs	4.9 Mbps
		Hash(Long)	-	-
	Artix-7	AD+PT(1536 Bytes)	537 LUTs	6.3 Mbps
		AD+PT(64 Bytes)		5.4 Mbps
		AD+PT(16 Bytes)		3.8 Mbps
	Cyclone 10	AD+PT(1536 Bytes)	817 LEs	4.3 Mbps
		AD+PT(64 Bytes)		3.7 Mbps
		AD+PT(16 Bytes)		2.6 Mbps
	ECP5	AD+PT(1536 Bytes)	1,277 LUTs	2.4 Mbps
		AD+PT(64 Bytes)		2.1 Mbps
		AD+PT(16 Bytes)		1.5 Mbps
TinyJAMBU-TJT-v1	Artix-7	AD+PT(Long)	446 LUTs	104.3 Mbps

Continued on next page

表 11.2 – continued from previous page

Design	Platform	Data	Area	Throughput
		Hash(Long)	-	-
	Cyclone 10	AD+PT(Long)	686 LEs	72.2 Mbps
		Hash(Long)	-	-
	ECP5	AD+PT(Long)	580 LUTs	4-
		Hash(Long)	-	-
	Artix-7	AD+PT(1536 Bytes)	446 LUTs	51.8 Mbps
		AD+PT(64 Bytes)		44.4 Mbps
		AD+PT(16 Bytes)		30.8 Mbps
	Cyclone 10	AD+PT(1536 Bytes)	686 LEs	35.8 Mbps
		AD+PT(64 Bytes)		30.8 Mbps
		AD+PT(16 Bytes)		21.3 Mbps
	ECP5	AD+PT(1536 Bytes)	580 LUTs	19.9 Mbps
		AD+PT(64 Bytes)		17.0 Mbps
		AD+PT(16 Bytes)		11.8 Mbps
TinyJAMBU-TJT-v2	Artix-7	AD+PT(Long)	461 LUTs	438.3 Mbps
		Hash(Long)	-	-
	Cyclone 10	AD+PT(Long)	777 LEs	273.0 Mbps
		Hash(Long)	-	-
	ECP5	AD+PT(Long)	689 LUTs	174.5 Mbps
		Hash(Long)	-	-
	Artix-7	AD+PT(1536 Bytes)	461 LUTs	217.5 Mbps
		AD+PT(64 Bytes)		186.0 Mbps
		AD+PT(16 Bytes)		128.0 Mbps
	Cyclone 10	AD+PT(1536 Bytes)	777 LEs	135.5 Mbps
		AD+PT(64 Bytes)		115.9 Mbps
		AD+PT(16 Bytes)		79.7 Mbps
	ECP5	AD+PT(1536 Bytes)	689 LUTs	86.6 Mbps
		AD+PT(64 Bytes)		74.1 Mbps
		AD+PT(16 Bytes)		51.0 Mbps
TinyJAMBU-TJT-v3	Artix-7	AD+PT(Long)	576 LUTs	1,396.4 Mbps
		Hash(Long)	-	-
	Cyclone 10	AD+PT(Long)	1,021 LEs	929.1 Mbps
		Hash(Long)	-	-
	ECP5	AD+PT(Long)	1,092 LUTs	671.4 Mbps
		Hash(Long)	-	-
	Artix-7	AD+PT(1536 Bytes)	576 LUTs	691.1 Mbps
		AD+PT(64 Bytes)		561.1 Mbps
		AD+PT(16 Bytes)		353.1 Mbps
	Cyclone 10	AD+PT(1536 Bytes)	1,021 LEs	459.9 Mbps
		AD+PT(64 Bytes)		373.3 Mbps

Continued on next page

表 11.2 – continued from previous page

Design	Platform	Data	Area	Throughput
		AD+PT(16 Bytes)		234.9 Mbps
	ECP5	AD+PT(1536 Bytes)	1,092 LUTs	332.3 Mbps
		AD+PT(64 Bytes)		269.8 Mbps
		AD+PT(16 Bytes)		169.8 Mbps
LWC HW APIを用いている．認証暗号の性能は暗号化時のものである．				

11.2.2 コプロセッサ（ASIC実装）

ElsadekらによるTinyJambuのASIC実装性能評価の結果は表11.3のとおりである [9]． 16バイトデータを長い時間をかけて処理する場合 (Short)に対して，TinyJambuはファイナリストの中で最もエネルギー効率の高い結果が得られたと報告している．また，1536バイトのデータを連続して処理をする場合 (Long)に対しても，Xoodyakの次に高いエネルギー効率が得られたとしている．

表11.3: ElsadekらによるTinyJambuのASIC実装の結果 [9]

Design	I/F	Area	Data	Throughput @f_{max}	Energy Efficiency
TinyJambu (Enc.)	no	3.6 kGE	Short/PT	33.1 Mbps	831.0 Mbit/mJ
			Short/AD	35.3 Mbps	885.0 Mbit/mJ
			Short/PT+AD	60.0 Mbps	1,335.0 Mbit/mJ
			Long/PT	191.4 Mbps	2,808.4 Mbit/mJ
			Long/AD	191.4 Mbps	4,429.8 Mbit/mJ
			Long/PT+AD	248.2 Mbps	3,453.0 Mbit/mJ

ライブラリ：GF 22 nm CMOS (GF22FDx)，インタフェース回路含まず
Short：16バイトデータを長い時間を空けて処理，Long：1536バイトのデータを処理.

11.3 命令拡張

表11.4は，ChengらによるRISC-Vの命令拡張におけるTinyJambuの実装結果である [7]．提案された命令拡張により，２倍以上の処理性能が可能であることがわかる．

11.4 ソフトウェア実装

表11.5は，文献 [12]でのTinyJambuのArm Cortex-M0上の実装結果である．また，文献 [25]で報告されているAVR ATmega 128とArm Cortex-M3でのソフトウェア実装の結果を表11.6に示す．

表11.4: Chenらの命令拡張によるTinyJambuの実装結果 [7]

Design	Area	Cycles/Byte
TinyJambu RV32GC (Enc.)	3,303 LUTs	311.3
TinyJambu RV32GC+Zbkb/x (Enc.)	3,764 LUTs	262.3
TinyJambu RV32GC+Zbkb/x+ν_0^{32} (Enc.)	3.953 LUTs	149.4
TinyJambu RV32GC+Zbkb/x+ν_1^{32} (Enc.)	3.863 LUTs	149.4
TinyJambu RV32GC (Dec.)	3,303 LUTs	315.9
TinyJambu RV32GC+Zbkb/x (Dec.)	3,764 LUTs	265.9
TinyJambu RV32GC+Zbkb/x+ν_0^{32} (Dec.)	3.953 LUTs	152.8
TinyJambu RV32GC+Zbkb/x+ν_1^{32} (Dec.)	3.863 LUTs	152.8

入力データ長は128バイトである.

表11.5: TinyJambuのArm Cortex-M0上でのレイテンシ評価 [12]

Design	0 to 32Bytes Encryption [msec]	0 to 32 Bytes Decryption [msec]	ROM [kByte]	Code [kByte]
tinyjambu128	114	115	15.7	13.7
	(0.394)	(0.398)		
tinyjambu256	128	129	15.7	13.7
	(0.443)	(0.446)		
tinyjambu192	1,338	1,339	15.7	13.7
	(4.630)	(4.633)		

RAMの使用量はコンパイル時の静的解析で約1 kByte程度，動作周波数は48MHz
レイテンシは289通りの入力データでの値でカッコ内は平均値.

表11.6: TinyJambuのArm Cortex-M3とAVR ATmega上でのレイテンシ評価 [25]

Design	Platform	16-byte AD + 16-byte Msg. [msec]	ROM [kByte]	RAM [Byte]
TinyJAMBU-128 (Enc.)	AVR ATmega	10.20	3,890	117
TinyJAMBU-128 (Dec.)	@16 Mhz	10.20		119
TinyJAMBU-128 (Enc.)	Arm Cortex-M3	0.34	2,096	140
TinyJAMBU-128 (Dec.)	@84 MHz	0.34		140

12 Xoodyak

12.1 概要

Xoodyakの設計者（提出者）は以下のとおりである．

- Joan Daemen
- Seth Hoffert
- Michaël Peeters
- Gilles Van Assche
- Ronny Van Keer
- Silvia Mella

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://keccak.team/xoodyak.html`
- 仕様：`https://csrc.nist.gov/CSRC/media/Projects/lightweight-cryptography/documents/finalist-round/updated-spec-doc/xoodyak-spec-final.pdf`

12.2 ハードウェアアクセラレータ

12.2.1 コプロセッサ（FPGA実装）

XoodyakのFPGA実装については，NIST主催のLightweight Cryptography Workshop 2022でのAbdulgadirらの報告がある [2]． Artix-7上にLWC HW APIを用いた実装で，1536バイトのデータを暗号化した際に1.7 Gbpsを超えるスループットを達成している（表12.1）．極めて高速な実装結果といえる．

表12.1: XoodyakのFPGA実装の結果 [2]

Design	Platform	I/F	Area	Throughput
Xoodyak	Artix-7	LWC HW API	1,808 LUTs	1717.9 Mbps

また，他のFPGA実装の結果として，Mohajeraniらの報告 [18]を表12.2にまとめる．同じ1,801 LUTsの面積コストで文献 [2]とスループット性能が異なる異なる結果が見られるが，ElephantとTinyJambuと同様，その理由は不明である．

表12.2: XoodyakのFPGA実装の結果（つづき） [18]

Design	Platform	Data	Area	Throughput
Xoodyak-GMU-v1	Artix-7	AD+PT(Long)	1,808 LUTs	2,150.7 Mbps
		Hash(Long)		1,280.0 Mbps
	Cyclone 10	AD+PT(Long)	3,135 LEs	1,351.0 Mbps

Continued on next page

表 12.2 – continued from previous page

Design	Platform	Data	Area	Throughput
		Hash(Long)		804.1 Mbps
	ECP5	AD+PT(Long)	3,172 LUTs	936.2 Mbps
		Hash(Long)		557.2 Mbps
	Artix-7	AD+PT(1536 Bytes)		996.2 Mbps
		AD+PT(64 Bytes)		626.2 Mbps
		AD+PT(16 Bytes)		286.3 Mbps
		Hash(1536 Bytes)		1,261.4 Mbps
		Hash(64 Bytes)		946.1 Mbps
		Hash(16 Bytes)		530.7 Mbps
	Cyclone 10	AD+PT(1536 Bytes)	3,135 LEs	625.8 Mbps
		AD+PT(64 Bytes)		393.4 Mbps
		AD+PT(16 Bytes)		179.9 Mbps
		Hash(1536 Bytes)		792.4 Mbps
		Hash(64 Bytes)		594.3 Mbps
		Hash(16 Bytes)		333.4 Mbps
	ECP5	AD+PT(1536 Bytes)	3,172 LUTs	433.6 Mbps
		AD+PT(64 Bytes)		272.6 Mbps
		AD+PT(16 Bytes)		124.6 Mbps
		Hash(1536 Bytes)		549.1 Mbps
		Hash(64 Bytes)		411.8 Mbps
		Hash(16 Bytes)		231.0 Mbps
Xoodyak-GMU-v2	Artix-7	AD+PT(Long)	1,234 LUTs	173.4 Mbps
		Hash(Long)		83.0 Mbps
	Cyclone 10	AD+PT(Long)	5,871 LEs	79.5 Mbps
		Hash(Long)		38.1 Mbps
	ECP5	AD+PT(Long)	2,316 LUTs	77.2 Mbps
		Hash(Long)		37.0 Mbps
	Artix-7	AD+PT(1536 Bytes)	1,234 LUTs	77.8 Mbps
		AD+PT(64 Bytes)		46.7 Mbps
		AD+PT(16 Bytes)		20.4 Mbps
		Hash(1536 Bytes)		82.1 Mbps
		Hash(64 Bytes)		65.5 Mbps
		Hash(16 Bytes)		40.0Mbps
	Cyclone 10	AD+PT(1536 Bytes)	5,871 LEs	35.7 Mbps
		AD+PT(64 Bytes)		21.4 Mbps
		AD+PT(16 Bytes)		9.4 Mbps
		Hash(1536 Bytes)		37.7 Mbps
		Hash(64 Bytes)		30.0 Mbps
		Hash(16 Bytes)		18.4 Mbps
	ECP5	AD+PT(1536 Bytes)	2,316 LUTs	34.6 Mbps

Continued on next page

表 12.2 – continued from previous page

Design	Platform	Data	Area	Throughput
		AD+PT(64 Bytes)		20.8 Mbps
		AD+PT(16 Bytes)		9.1 Mbps
		Hash(1536 Bytes)		36.5 Mbps
		Hash(64 Bytes)		29.1 Mbps
		Hash(16 Bytes)		17.8 Mbps
Xoodyak-GMU2-v1	Artix-7	AD+PT(Long)	1,608 LUTs	6,569.8 Mbps
		Hash(Long)		3,091.7 Mbps
	Cyclone 10	AD+PT(Long)	2,575 LEs	3,563.2 Mbps
		Hash(Long)		1,676.8 Mbps
	ECP5	AD+PT(Long)	3,248 LUTs	3,148.5 Mbps
		Hash(Long)		1,481.6 Mbps
	Artix-7	AD+PT(1536 Bytes)	1,608 LUTs	2,892.4 Mbps
		AD+PT(64 Bytes)		1,435.4 Mbps
		AD+PT(16 Bytes)		550.6 Mbps
		Hash(1536 Bytes)		3,012.0 Mbps
		Hash(64 Bytes)		1,891.4 Mbps
		Hash(16 Bytes)		873.7 Mbps
	Cyclone 10	AD+PT(1536 Bytes)	2,575 LEs	1,568.7 Mbps
		AD+PT(64 Bytes)		778.5 Mbps
		AD+PT(16 Bytes)		298.6 Mbps
		Hash(1536 Bytes)		1,633.6 Mbps
		Hash(64 Bytes)		1,025.8 Mbps
		Hash(16 Bytes)		473.9 Mbps
	ECP5	AD+PT(1536 Bytes)	3,248 LUTs	1,386.1 Mbps
		AD+PT(64 Bytes)		687.9 Mbps
		AD+PT(16 Bytes)		263.9 Mbps
		Hash(1536 Bytes)		1,443.5 Mbps
		Hash(64 Bytes)		906.4 Mbps
		Hash(16 Bytes)		418.7 Mbps
Xoodyak-GMU2-v2	Artix-7	AD+PT(Long)	2,322 LUTs	5,697.7 Mbps
		Hash(Long)		3,638.9 Mbps
	Cyclone 10	AD+PT(Long)	5,058 LEs	2,782.7 Mbps
		Hash(Long)		1,777.2 Mbps
	ECP5	AD+PT(Long)	4,058 LUTs	1,994.8 Mbps
		Hash(Long)		1,274.0 Mbps
	Artix-7	AD+PT(1536 Bytes)	2,322 LUTs	2,714.0 Mbps
		AD+PT(64 Bytes)		1,306.3 Mbps
		AD+PT(16 Bytes)		489.8 Mbps
		Hash(1536 Bytes)		3,523.5 Mbps
		Hash(64 Bytes)		2,037.8 Mbps

Continued on next page

表 12.2 – continued from previous page

Design	Platform	Data	Area	Throughput
		Hash(16 Bytes)		878.3 Mbps
	Cyclone 10	AD+PT(1536 Bytes)	5,058 LEs	1,325.5 Mbps
		AD+PT(64 Bytes)		638.0 Mbps
		AD+PT(16 Bytes)		239.2 Mbps
		Hash(1536 Bytes)		1,720.9 Mbps
		Hash(64 Bytes)		995.2 Mbps
		Hash(16 Bytes)		429.0 Mbps
	ECP5	AD+PT(1536 Bytes)	4,058 LUTs	950.2 Mbps
		AD+PT(64 Bytes)		457.3 Mbps
		AD+PT(16 Bytes)		171.5 Mbps
		Hash(1536 Bytes)		1,233.6 Mbps
		Hash(64 Bytes)		713.4 Mbps
		Hash(16 Bytes)		307.5 Mbps
Xoodyak-XT-v1	Artix-7	AD+PT(Long)	1,355 LUTs	2,960.4 Mbps
		Hash(Long)	-	-
	Cyclone 10	AD+PT(Long)	2,231 LEs	1,724.7 Mbps
		Hash(Long)	-	-
	ECP5	AD+PT(Long)	2,402 LUTs	1,210.7 Mbps
		Hash(Long)	-	-
	Artix-7	AD+PT(1536 Bytes)	1,355 LUTs	1,371.2 Mbps
		AD+PT(64 Bytes)		861.9 Mbps
		AD+PT(16 Bytes)		394.1 Mbps
	Cyclone 10	AD+PT(1536 Bytes)	2,231 LEs	798.9 Mbps
		AD+PT(64 Bytes)		502.2 Mbps
		AD+PT(16 Bytes)		229.6 Mbps
	ECP5	AD+PT(1536 Bytes)	2,402 LUTs	560.8 Mbps
		AD+PT(64 Bytes)		352.5 Mbps
		AD+PT(16 Bytes)		161.2 Mbps
Xoodyak-XT-v2	Artix-7	AD+PT(Long)	2,025 LUTs	3,299.1 Mbps
		Hash(Long)	-	-
	Cyclone 10	AD+PT(Long)	3,541 LEs	1,558.8 Mbps
		Hash(Long)	-	-
	ECP5	AD+PT(Long)	4,077 LUTs	1,234.2 Mbps
		Hash(Long)	-	-
	Artix-7	AD+PT(1536 Bytes)	2,025 LUTs	1,549.4 Mbps
		AD+PT(64 Bytes)		992.3 Mbps
		AD+PT(16 Bytes)		462.8 Mbps
	Cyclone 10	AD+PT(1536 Bytes)	3,541 LEs	732.1 Mbps
		AD+PT(64 Bytes)		468.9 Mbps
		AD+PT(16 Bytes)		218.7 Mbps

Continued on next page

表 12.2 – continued from previous page

Design	Platform	Data	Area	Throughput
	ECP5	AD+PT(1536 Bytes)	4,077 LUTs	579.6 Mbps
		AD+PT(64 Bytes)		371.2 Mbps
		AD+PT(16 Bytes)		173.1 Mbps
Xoodyak-XT-v7	Artix-7	AD+PT(Long)	1,392 LUTs	2,859.2 Mbps
		Hash(Long)		1,701.6 Mbps
	Cyclone 10	AD+PT(Long)	2,272 LEs	1,626.1 Mbps
		Hash(Long)		967.8 Mbps
	ECP5	AD+PT(Long)	2,489 LUTs	1,118.6 Mbps
		Hash(Long)		665.7 Mbps
	Artix-7	AD+PT(1536 Bytes)	1,392 LUTs	1,324.3 Mbps
		AD+PT(64 Bytes)		832.5 Mbps
		AD+PT(16 Bytes)		380.6 Mbps
		Hash(1536 Bytes)		1,677.0 Mbps
		Hash(64 Bytes)		1,257.7 Mbps
		Hash(16 Bytes)		705.6 Mbps
	Cyclone 10	AD+PT(1536 Bytes)	2,272 LEs	753.2 Mbps
		AD+PT(64 Bytes)		473.4 Mbps
		AD+PT(16 Bytes)		216.5 Mbps
		Hash(1536 Bytes)		953.7 Mbps
		Hash(64 Bytes)		715.3 Mbps
		Hash(16 Bytes)		401.3 Mbps
	ECP5	AD+PT(1536 Bytes)	2,489 LUTs	518.1 Mbps
		AD+PT(64 Bytes)		325.7 Mbps
		AD+PT(16 Bytes)		148.9 Mbps
		Hash(1536 Bytes)		656.1 Mbps
		Hash(64 Bytes)		492.1 Mbps
		Hash(16 Bytes)		276.0 Mbps
Xoodyak-XT-v8	Artix-7	AD+PT(Long)	2,143 LUTs	3,176.3 Mbps
		Hash(Long)		2,106.2 Mbps
	Cyclone 10	AD+PT(Long)	3,630 LEs	1,578.5 Mbps
		Hash(Long)		1,046.7 Mbps
	ECP5	AD+PT(Long)	4,121 LUTs	1,251.2 Mbps
		Hash(Long)		829.7 Mbps
	Artix-7	AD+PT(1536 Bytes)	2,143 LUTs	1,491.7 Mbps
		AD+PT(64 Bytes)		955.4 Mbps
		AD+PT(16 Bytes)		445.5 Mbps
		Hash(1536 Bytes)		2,070.9 Mbps
		Hash(64 Bytes)		1,494.7 Mbps
		Hash(16 Bytes)		798.9 Mbps
	Cyclone 10	AD+PT(1536 Bytes)	3,630 LEs	741.3 Mbps

Continued on next page

表 12.2 – continued from previous page

Design	Platform	Data	Area	Throughput
		AD+PT(64 Bytes)		474.8 Mbps
		AD+PT(16 Bytes)		221.4 Mbps
		Hash(1536 Bytes)		1,029.1 Mbps
		Hash(64 Bytes)		742.8 Mbps
		Hash(16 Bytes)		397.0 Mbps
	ECP5	AD+PT(1536 Bytes)	4,121 LUTs	587.6 Mbps
		AD+PT(64 Bytes)		376.3 Mbps
		AD+PT(16 Bytes)		175.5 Mbps
		Hash(1536 Bytes)		815.8 Mbps
		Hash(64 Bytes)		588.8 Mbps
		Hash(16 Bytes)		314.7 Mbps
LWC HW APIを用いている．認証暗号の性能は暗号化時のものである．				

Xoodyak-XT-v8において，最高で3 Gbpsを超える認証付き暗号の処理性能が得られたとしている．高速なデータインタフェースが実現できることが前提であるが，アルゴリズムのコア性能として極めて高い処理能力を持つことがわかる．ただし，入力データが短い場合には450 Mbpsまでスループット性能が低下しており，短いデータをアイドル期間を空けて送信するような性能ではASCONなどの他のアルゴリズムと大きな差はない．

12.2.2 コプロセッサ（ASIC実装）

文献 [9]でのXoodyakのASIC実装は表12.3のとおりである．

表12.3: ElsadekらによるXoodyakのASIC実装の結果 [9]

Design	I/F	Area	Data	Throughput @f_{max}	Energy Efficiency
Xoodyak (Enc.)	no	11.9 kGE	Short/PT	39.6 Mbps	357.3 Mbit/mJ
			Short/AD	39.6 Mbps	354.0 Mbit/mJ
			Short/PT+AD	79.2 Mbps	720.9 Mbit/mJ
Xoodyak (Enc.)	no	11.9 kGE	Long/PT	783.0 Mbps	3,228.0 Mbit/mJ
			Long/AD	783.0 Mbps	5,753.0 Mbit/mJ
			Long/PT+AD	1,028.3 Mbps	4,263.0 Mbit/mJ

ライブラリ：GF 22 nm CMOS(GF22FDx)，インタフェース回路含まず
Short：16バイトデータを長い時間を空けて処理，Long：1536バイトのデータを処理.

12.3 命令拡張

表12.4は，ChengらによるRISC-Vの命令拡張におけるXoodyakの実装結果である [7]．命令拡張により，15倍程度の高速化が可能であることがわかる．拡張した命令がXoodyakの処理に適し

ていたためと思われる．

表12.4: Chenらの命令拡張によるXoodyakの実装結果 [7]

Design	Area	Cycles/Byte
Xoodyak RV32GC (Enc.)	3,303 LUTs	1,502.6
Xoodyak RV32GC+Zbkb/x (Enc.)	3,764 LUTs	113.9
Xoodyak RV32GC+Zbkb/x+ν_0^{32} (Enc.)	3,814 LUTs	106.4
Xoodyak RV32GC (Dec.)	3,303 LUTs	1,501.2
Xoodyak RV32GC+Zbkb/x (Dec.)	3,764 LUTs	112.5
Xoodyak RV32GC+Zbkb/x+ν_0^{32} (Dec.)	3,814 LUTs	104.9

入力データ長は128バイトである．

12.4 ソフトウェア実装

表12.5に文献 [12]でXoodyakのArm Cortex-M0実装の結果を示す．

表12.5: XoodyakのArm Cortex-M0上でのレイテンシ評価 [12]

Design	0 to 32Bytes Encryption [msec]	0 to 32 Bytes Decryption [msec]	ROM [kByte]	Code [kByte]
xoodyakv1	173 (0.599)	174 (0.602)	16.3	14.3

RAMの使用量はコンパイル時の静的解析で約1 kByte程度，動作周波数は48MHz
レイテンシは289通りの入力データでの値でカッコ内は平均値．

また，文献 [25]で報告されているAVR ATmega 128とArm Cortex-M3でのソフトウェア実装の結果を表12.6に示す． 16 MHz動作のATmegaでも3 msec程度のレイテンシであり，実用性は十分あるものと考える．

表12.6: XoodyakのAVR ATmegaとArm Cortex-M3上でのレイテンシ評価 [25]

Design	Platform	16-byte AD + 16-byte Msg. [msec]	ROM [kByte]	RAM [Byte]
Xoodyak (Enc.)	AVR ATmega	3.25	4,306	167
Xoodyak (Dec.)	@16 Mhz	3.23		183
Xoodyak (Enc.)	Arm Cortex-M3	0.19	3,572	208
Xoodyak (Dec.)	@84 MHz	0.19		232

参考文献

[1] Lightweight cryptography in hardware and embedded systems evaluation of finalists in the NIST LWC process. `https://cryptography.gmu.edu/athena/index.php?id=LWC`.

[2] Abubakr Abdulgadir, Richard Haeussler, Sammy Lin, Jens-Peter Kaps, and Kris Gaj. Side-Channel Resistant Implementations of Three Finalists of the NIST Lightweight Cryptography Standardization Process: Elephant, TinyJAMBU, and Xoodyak. *Lightweight Cryptography Workshop 2022*, pages 1–7, 2022.

[3] Fatih Balli, Andrea Caforio, and Subhadeep Banik. The area-latency symbiosis: Towards improved serial encryption circuits. *IACR Trans. Cryptogr. Hardw. Embed. Syst.*, 2021(1):239–278, 2021.

[4] Subhadeep Banik, Fatih Balli, Francesco Regazzoni, and Serge Vaudenay. Swap and rotate: Lightweight linear layers for spn-based blockciphers. *IACR Trans. Symmetric Cryptol.*, 2020(1):185–232, 2020.

[5] Daniel J. Bernstein and Tanja Lange (editors). eBACS: ECRYPT benchmarking of cryptographic systems. `https://bench.cr.yp.to`.

[6] Andrea Caforio, Daniel Collins, Subhadeep Banik, and Francesco Regazzoni. A small GIFT-COFB: lightweight bit-serial architectures. *IACR Cryptol. ePrint Arch., Paper 2022/955*, 2022.

[7] Hao Cheng, Johann Großschädl, Ben Marshall, Dan Page, and Thinh Pham. RISC-V Instruction Set Extensions for Lightweight Symmetric Cryptography. *Lightweight Cryptography Workshop 2022*, pages 1–50, 2022.

[8] Christoph Dobraunig, Maria Eichlseder, Florian Mendel, and Martin Schläffer. Ascon v1.2: Lightweight authenticated encryption and hashing. *J. Cryptol.*, 34(3):33, 2021.

[9] Islam Elsadek, Sohrab Aftabjahani, Doug Gardner, Erik MacLean, John Ross Wallrabenstein, and Eslam Yahya Tawfik. Hardware and energy efficiency evaluation of NIST lightweight cryptography standardization finalists. In *IEEE International Symposium on Circuits and Systems, ISCAS 2022*, pages 133–137. IEEE, 2022.

[10] Farnoud Farahmand, William Diehl, Abubakr Abdulgadir, Jens-Peter Kaps, and Kris Gaj. Improved lightweight implementations of caesar authenticated ciphers. In *2018 IEEE 26th Annual International Symposium on Field-Programmable Custom Computing Machines (FCCM)*, pages 29–36, 2018.

[11] Hannes Groß, Erich Wenger, Christoph Dobraunig, and Christoph Ehrenhöfer. Suit up! - made-to-measure hardware implementations of ASCON. In *2015 Euromicro Confer-*

ence on Digital System Design, DSD 2015, Madeira, pages 645–652. IEEE Computer Society, 2015.

[12] Ryota Hira, Tomoaki Kitahara, Daiki Miyahara, Yuko Hara-Azumi, Yang Li, and Kazuo Sakiyama. Software evaluation for second round candidates in NIST lightweight cryptography. *IACR Cryptol. ePrint Arch., Paper 2022/591*, 2022.

[13] Ekawat Homsirikamol, William Diehl, Ahmed Ferozpuri, Farnoud Farahmand, Panasayya Yalla, Jens-Peter Kaps, and Kris Gaj. CAESAR hardware API. *IACR Cryptol. ePrint Arch., Paper 2016/626*, 2016.

[14] IACR. Cryptology ePrint Archive. `https://eprint.iacr.org`.

[15] IACR. IACR online proceedings. `https://www.iacr.org/archive/`.

[16] IEEE. IEEE Xplore. `https://ieeexplore.ieee.org/Xplore/home.jsp`.

[17] Jens-Peter Kaps, William Diehl, Michael Tempelmeier, Ekawat Homsirikamol, and Kris Gaj. Hardware API for lightweight cryptography. `https://cryptography.gmu.edu/athena/LWC/LWC_HW_API.pdf`.

[18] Kamyar Mohajerani, Richard Haeussler, Rishub Nagpal, Farnoud Farahmand, Abubakr Abdulgadir, Jens-Peter Kaps, and Kris Gaj. FPGA benchmarking of round 2 candidates in the NIST lightweight cryptography standardization process: Methodology, metrics, tools, and results. *IACR Cryptol. ePrint Arch., Paper 2020/1207*, 2020.

[19] NIST. NIST lightweight cryptography. `https://csrc.nist.gov/projects/lightweight-cryptography`.

[20] Behnaz Rezvani and William Diehl. Hardware implementations of NIST lightweight cryptographic candidates: A first look. *IACR Cryptol. ePrint Arch., Paper 2019/824*, 2019.

[21] Jonathan Sönnerup, Martin Hell, Mattias Sönnerup, and Ripudaman Khattar. Efficient hardware implementations of Grain-128AEAD. In Feng Hao, Sushmita Ruj, and Sourav Sen Gupta, editors, *Progress in Cryptology - INDOCRYPT 2019 - 20th International Conference on Cryptology in India, Hyderabad, India, December 15-18, 2019, Proceedings*, volume 11898 of *Lecture Notes in Computer Science*, pages 495–513. Springer, 2019.

[22] Springer. Springer link. `https://link.springer.com`.

[23] Stefan Steinegger and Robert Primas. A fast and compact RISC-V accelerator for Ascon and friends. In Pierre-Yvan Liardet and Nele Mentens, editors, *Smart Card Research and Advanced Applications - 19th International Conference, CARDIS 2020, Virtual Event, November 18-19, 2020, Revised Selected Papers*, volume 12609 of *Lecture Notes in Computer Science*, pages 53–67. Springer, 2020.

[24] Michael Tempelmeier, Fabrizio De Santis, Georg Sigl, and Jens-Peter Kaps. The CAESAR-API in the real world - towards a fair evaluation of hardware CAESAR candidates. In *2018 IEEE International Symposium on Hardware Oriented Security and Trust, HOST 2018, Washington, DC, USA, April 30 - May 4, 2018*, pages 73–80. IEEE Computer Society, 2018.

[25] Yuhei Watanabe, Hideki Yamamoto, and Hirotaka Yoshida. Performance evaluation of NIST LWC finalists on AVR atmega and ARM Cortex-M3 microcontrollers. *IACR Cryptol. ePrint Arch., Paper 2022/1071*, 2022.

[26] Panasayya Yalla and Jens-Peter Kaps. Evaluation of the CAESAR hardware API for lightweight implementations. In *2017 International Conference on ReConFigurable Computing and FPGAs (ReConFig)*, pages 1–6, 2017.

[27] 岡部　忠. 軽量ストリーム暗号のハードウェア実装　～ FPGAを対象デバイスとした実装性能の比較～. 情報処理学会講演論文集, 2022.