ハッシュ関数SHA-224, SHA-512/224, SHA-512/256及びSHA-3 (Keccak)に関する実装評価

電気通信大学 大学院情報理工学研究科

崎山　一男

2014年2月

目 次

1 はじめに

本報告は，FIPS PUB 180-4 [36] で仕様が策定されているハッシュ関数と，FIPS Draft が公開予定である SHA-3 について，国際会議等で発表された文献や公開情報に基づき，主にハードウェア実装評価に関する研究動向調査を行うものである．

・SHA-224 (FIPS PUB 180-4)　・SHA-512/224 (FIPS PUB 180-4)

・SHA-512/256 (FIPS PUB 180-4)　・SHA-3

なお，SHA-3 は，FIPS Draft がまだ公開されていないため，Keccak の仕様に基づき研究調査を行う[1]．本報告では，特に処理性能と実装に必要となるコストに焦点を絞り調査を行う[2]．

SHA-224 は，その仕様から，SHA-256 とほぼ同等のスループット性能を有すると言えるが，演算結果（ハッシュ値）のデータ・サイズが小さい分，データ転送に要する時間を含めたレイテンシの短縮が期待できる．また，数多くのハッシュ値を保存するようなアプリケーションでは，ハッシュ値の記憶に必要となるメモリ容量が削減できることができるため，システム・コストの削減が期待できる．

SHA-512/224 と SHA-512/256 はそれぞれ SHA-224 と SHA-256 と比べ，64 bit CPU でのソフトウェア性能で一定の優位性があることが示されている．これは，64 bit 演算を高速で行う CPU の演算処理能に依るものである．つまり，全体の計算量は多くなるにも関わらず，64 bit 演算に基づくアルゴリズムの処理が，32 bit 演算に基づくアルゴリズムの処理よりも，64 bit CPU では高速に処理可能となる逆転現象が起きたためである．

Keccak のソフトウェア実装についての調査は，eBASH (ECRYPT Benchmarking of All Submitted Hashes) の “List of SHA-3 finalists measured” のデータを利用し，調査する．64 bit CPU における Keccak のソフトウェア実装の性能は，SHA-256 と同等以上であるが，SHA-512 と比べると約 1.5 倍の処理時間となることが報告されている．また，32 bit CPU でのソフトウェア実装は，SHA-512 と同等以上の性能を有するが，SHA-256 と比べ約 1.8 倍の処理時間となることが報告されている．

最後に，Keccak のハードウェア実装について，これまでに発表された文献や公開情報を調べ，処理性能と実装コストについて調査を行う．文献により，Xilinx 社の FPGA，Altera 社の FPGA，ASIC 実装といった異なるプラットフォームでハードウェアの性能評価が行われているため，単純な比較は困難であるが，Xilinx 社の Virtex-5 を用いたハードウェア実装については，20 件ほどの異なる実装評価結果が報告されている．そこで，本報告では，Virtex-5 上のハードウェア実装を中心に，高速実装と軽量実装の性能を比較する．Keccak のパラメータについては，主要なものが統一されたハードウェア実装結果を抽出し，実装評価環境の違いにより生じる多少の不公平さを容認した上で，Keccak のハードウェア実装の実力値を考察する．

[1] NIST の HP によると，FIPS Draft の公開は 2014 年 4〜6 月に予定されている [5].

[2] 本報告では安全性に関する評価は考慮しない．処理性能とコストのトレード・オフは，それぞれのハッシュ関数に対するものであり，ハッシュ関数の優劣を決定するものではない．

2　ハッシュ関数の実装評価

ハッシュ関数は，任意長の入力データに対して固定長の出力値（ハッシュ値）を返す関数である．ハードウェア実装の観点からは，データ圧縮装置の一種ととらえることができるが，実装性の高効率化と安全性の担保を同時に実現するために特定の処理を繰り返す設計が多い．

例えば，FIPS-180-4 [36] で仕様が策定されている SHA-256 のアルゴリズムでは，1 ブロック（512 bit）に対して合計 64 ステップの処理を繰り返しハッシュ値を出力する．1 サイクルで 1 ステップの処理を行う場合，合計で 64 サイクルが必要となり，2 ブロック以上の処理を行う場合，1 ブロック目のハッシュ値は 2 ブロック目のハッシュ値の計算における入力の一部となる．このため，一般に各ブロックの処理を並列実行することができず，メッセージ長に比例してレイテンシは長くなる．一方，スループットは 1 ブロックの処理に要する時間で決まるためメッセージ長に依存しない．以上の考察は，ほとんど全てのハッシュ関数アルゴリズムで適用可能である．

ハッシュ関数の実装性能を評価する際，メッセージ長が短い場合（ショート・メッセージ）とメッセージ長が長い場合（ロング・メッセージ）に分けて議論されることが多い．ショート・メッセージの場合，インターフェイスの影響を大きく受けてしまうため，ブロックの処理時間に対するメッセージ・データの転送時間は無視できない．従って，ショート・メッセージを入力とするハッシュ関数の性能の指標には，レイテンシを用いることが妥当である．一方，ロング・メッセージに対する性能を評価する場合には，スループットを指標とすることが多い．これは，メッセージ・データの転送とブロック処理とを並行処理することで，インターフェイスの影響を見かけ上小さくできるためである．文献 [32] でも同様のことが述べられているが，以下でメッセージ・サイズに対するスループット及びレイテンシへの影響を説明する．

ここで，

M :　パディング後のメッセージ・サイズ [bit],
B :　ブロック・サイズ [bit],
N_p :　メッセージ・パディング必要なクロックサイクル数,
N_i :　メモリ上のメッセージをコアに転送するのに必要なクロックサイクル数,
N_c :　ハッシュ関数処理コアに必要なクロックサイクル数,
N_f :　コアの最終処理に必要なクロックサイクル数,
N_o :　ハッシュ値をメモリに転送するのに必要なクロックサイクル数,
L :　ハッシュ値の生成におけるレイテンシ [sec],
T :　ハッシュ関数の入力データに対するスループット [bps],
f :　コア及びインターフェイスの最大動作周波数 [Hz].

とする．

メッセージのハッシュ処理に要する合計時間，つまりレイテンシは，

$$L = \left(N_p + \frac{M_p}{B} \cdot (N_i + N_c) + N_f + N_o \right) \cdot \frac{1}{f} \text{ [sec]}$$

である．ここで，たとえば，$M_p = B$ であるようなショート・メッセージを考える（メッセージが 1 ブロックの場合）．上記の式は，

$$L = \left(N_p + N_i + N_c + N_f + N_o \right) \cdot \frac{1}{f} \text{ [sec]}$$

となり，コア処理時間だけではなく，パディング処理，データの送受信及びハッシュ関数の最終処理時間に大きく影響されることが分かる．一方，スループット T は，

$$T = M_p \cdot f \cdot \left(N_p + \frac{M_p}{B} \cdot (N_i + N_c) + N_f + N_o \right)^{-1} \text{ [bps]}$$

となる．メッセージが十分長い場合には，パディング処理，データの送信及びハッシュ関数の最終処理時間を無視することができるため，スループットは

$$T \approx \frac{B \cdot f}{N_i + N_c} \text{ [bps]}$$

となり，メッセージ長に依存しないことが確認できる．

3　SHA-224 の実装評価

SHA-224 は，SHA-256 の処理において得られた 256 bit の出力値から 32 bit データ ($H_7^{(n)}$) を切り捨て，224 bit のハッシュ値を出力する[3]．多くの場合，実装形態に依らず処理性能は SHA-256 とほぼ同等と考える．ただし，最終データの転送に要する時間は 32 bit 分短縮できるため，ハッシュ値をメモリに転送する際に必要となるサイクル数 N_o に関して 12.5%程度の削減が期待できる．この効果は，メッセージ長で決まるロング・メッセージにおけるスループットには影響がほとんどないものの，ショート・メッセージにおけるレイテンシでは，僅かではあるが一定の効果が期待できる．また後述するが，HMAC のハードウェア実装においても，回路規模の削減とスループット性能の両方でその優位性が示されている．

実装コストについては，ハッシュ値の記憶に必要となるメモリ容量が削減できることが考えられる．特に，8 bit CPU 上でのソフトウェアプログラムや，RFID タグ等の低リソース向けのハードウェア実装で有効である．また，パスワード認証のように，膨大なハッシュ値の保管が必要となるようなアプリケーションでは，大幅なメモリの削減が期待できる．また，電力消費の観点でも多少の効果は期待できると考える．

3.1　SHA-224 の実装性能評価 [25][24]

SHA-224 の実装性能を報告する既存研究として，文献 [25] と [24] がある．文献 [25] は，SHA-256 と SHA-224 を全く同じ仕様の共通 IP(Intellectual Property) コアとして設計しており，性能は全く同じであるとしている．文献 [24] では，HMAC 実装において SHA-256 と SHA-224 のハッシュ値のビット長の違いを厳密に評価し，ハードウェア実装の結果，FPGA に実装した HMAC/SHA-224 が HMAC/SHA-256 と比べて回路規模と処理性能の両方で優位としている．この結果は，ハードウェアのみならずソフトウェアでも同様の優位性が期待できると考える．以上の文献調査により，SHA-224 に特化した実装を行うことで，僅かではあるが SHA-256 より高速かつ軽量に実装できることが期待できる．

[3]SHA-224 で用いられる初期値も SHA-256 と異なるが，処理性能に影響はない．

4　SHA-512/224, SHA-512/256 の実装評価

SHA-256 と SHA-512 をハードウェア実装した場合，アルゴリズムにおける計算量の違いにより SHA-512 は SHA-256 と比べ回路規模が増加するか，スループット性能が低下する．文献 [6] では，SHA-256 と SHA-512 を同じ回路規模で FPGA に実装した場合，ロング・メッセージに対するスループットはそれぞれ 1008 Mbps@41.97 MHz と 806 Mbps@41.97 MHz としている．言い換えれば，SHA-512 は SHA-256 の 80%のスループット性能となることが示されている．また，文献 [35] における Virtex-2 を用いた実装評価結果では，SHA-256 と SHA-512 でそれぞれ 1009 Mbps@133.06 MHz と 1329 Mbps@109.03 MHz のスループット性能を得るのに，1,373 slices と 2,726 slices の回路規模が必要としている．つまり，SHA-512 は SHA-256 と同等以上（約 130%）のスループットを得るのに，およそ 2 倍の回路規模が必要としている．

SHA-512/224 及び SHA-512/256 は，SHA-512 に基づくアルゴリズムでメッセージを処理し，512 bit のハッシュ値をそれぞれ 224 bit と 256 bit に切り捨て（トランケート）し出力する．つまり，SHA-512/224 及び SHA-512/256 は，SHA-512 のアルゴリズムを利用し，SHA-256 あるいは SHA-224 の代替として利用することを想定したアルゴリズムである．SHA-512 の計算量は SHA-256 と比べて多くなるが，64 bit CPU にソフトウエア実装した場合には，SHA-512/224 及び SHA-512/256 に優位性が見られる．これは，SHA-512 の基本演算が 64 bit であり，64 bit CPU の特性を有効利用したソフトウェア実装では SHA-512 の処理速度が SHA-256 と比べ速くなるためである．

4.1　SHA-512/256 の実装上の有効性 [17]

ePrint に掲載された文献 [17] で，Gueron らは 64 bit CPU における SHA-256 と SHA-512/256 の処理性能の評価を行っている．Gueron らは，2.67 GHz 動作の Intel Xeon X5670 プロセッサに Linux (OpenSuse 11.1 64 bit) を組み合せたプラットフォームにソフトウェア実装を行い，SHA-512/256 が SHA-256 と比べて高速であるとしている．

ハードウェア実装による SHA-512/224 と SHA-512/256 の有効性に関する報告は見られない．この理由は，ハードウェア実装の場合では，任意のバス幅を用意することができるためである．一般的に，回路規模と処理性能のトレードオフにおいては，SHA-256 が効率的であると言える．

5　SHA-3 (Keccak) の実装評価

2012 年 10 月 2 日，NIST (National Institute of Standards and Technology) が公募した SHA-3 候補から，Keccak（ケチャック）が選定された．Keccak は，STMicroelectronics の Guido Bertoni，Joan Daemen 及び Gilles Van Assche と NXP Semiconductor の Michaël Peeters が設計したスポンジ構造を有するハッシュ関数である．以降，Keccak の詳細と，これまでに報告された実装性能の結果について議論する．

5.1　Keccak の仕様

5.1.1　スポンジ構造

スポンジ構造は，固定長の permutation（置換）と padding（パディング）に基づく，mode of operation（利用モード）の一種である．図 1 にスポンジ構造を示す．$b = r + c$ [bit] のステートの初期値は 0 である[4]．

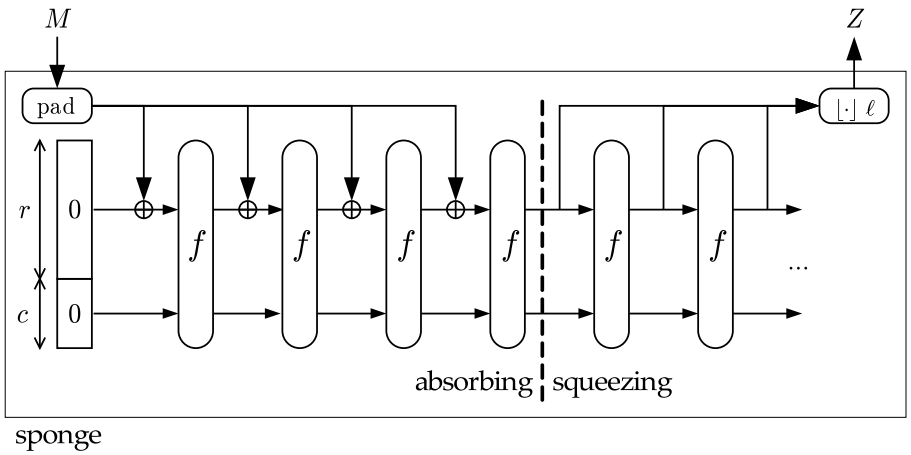


図 1: スポンジ構造（The Sponge Functions Corner, http://sponge.noekeon.org より引用）

スポンジ構造は，absorbing（吸収）と squeezing（搾出）の大きく 2 つのフェーズに分かれている．吸収のフェーズでは，パディング後のメッセージデータ M_p を r [bit] のデータに分割し，ステートの r [bit] のデータとの XOR 演算の後に関数 f に入力する．つまり，

$$S_r^{i+1} \,||S_c^{i+1} = f\big((S_r^i \oplus M_p^i \,||\, S_c^i\big)$$

となる．ここで，S_r^i と S_c^i は，f 関数を i 回処理した後のステートにおけるビットレートとキャパシティである（初期のステートでは $i = 0$）．既存研究における Keccak の実装評価は，この構造に基づくものである．

[4] ブロック・サイズ r を bitrate（ビットレート），c を capacity（キャパシティ）と呼ぶ．

5.1.2 デュプレックス構造

Keccak には，前述のスポンジ構造において吸収と搾出を交互に行うデュプレックス構造（図 2）も用意されている．参考までに掲載する．

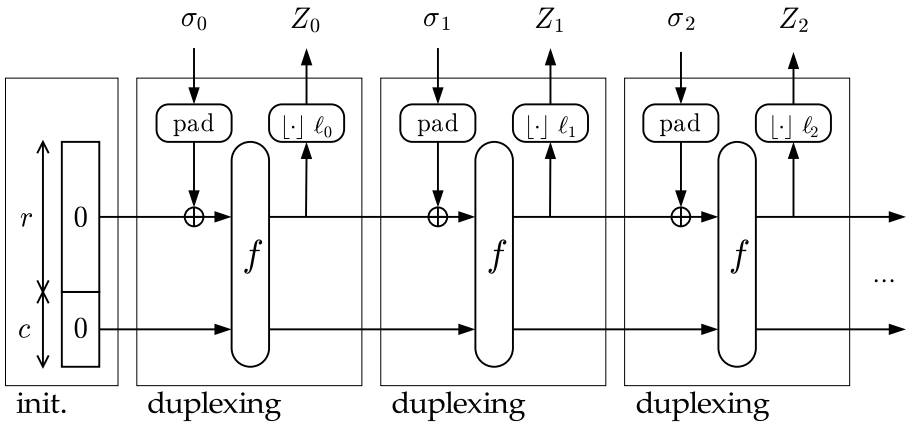
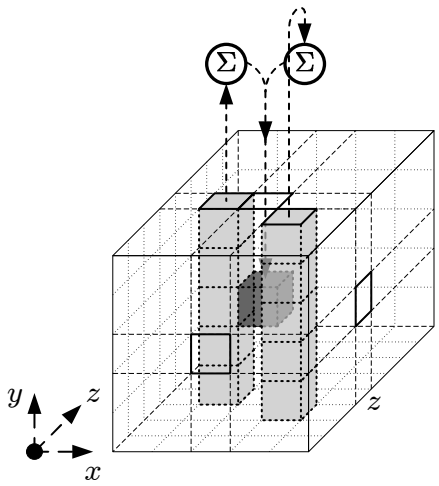


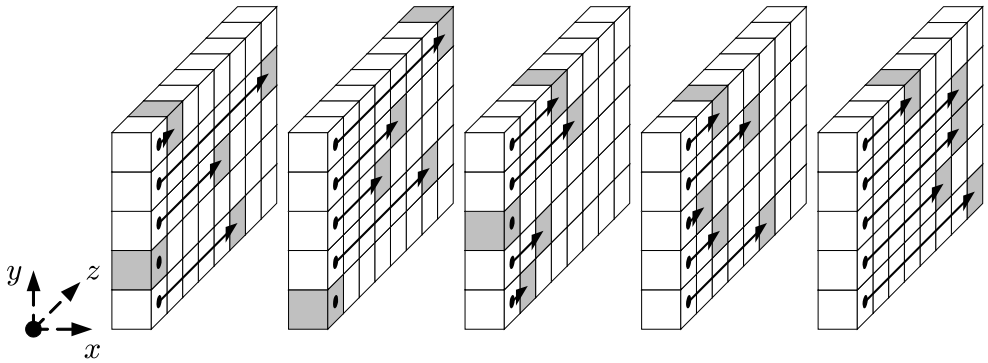
図 2: デュプレックス構造（The Sponge Functions Corner, http://sponge.noekeon.org より引用）

5.1.3 Keccak-f 置換関数

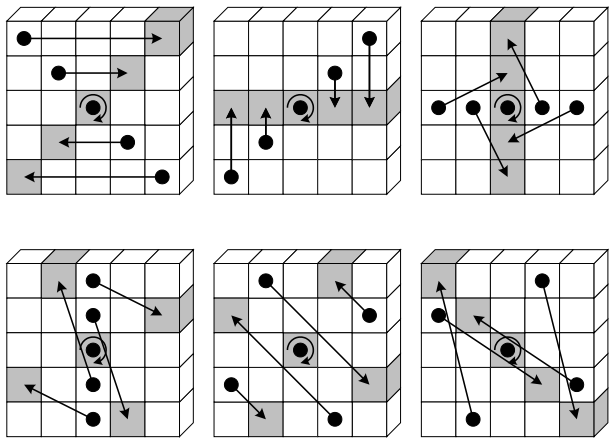
Keccak-f 置換関数は，図 3(a)〜3(d) に示すように，θ，ρ，π，χ の 4 つのステップとラウンド定数との XOR 処理を行う ι ステップにより，3 次元のステートが計算される．ハードウェア実装の観点からは，θ ステップにおけるパリティ処理 Σ と χ ステップの NOT と AND 演算を除き，配線だけで処理を実現できる．ステップ処理の詳細については [11] を参照されたい．



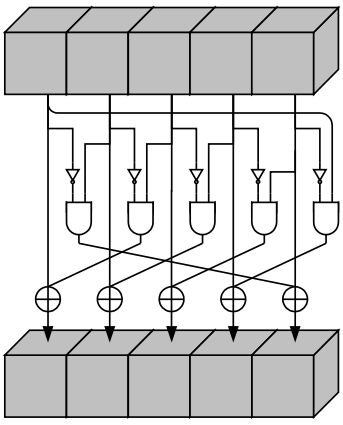
(a) θ ステップ



(b) ρ ステップ



(c) π ステップ



(d) χ ステップ

図 3: Keccak-f 置換 (The Sponge Functions Corner, http://sponge.noekeon.org より引用)

5.2 Keccak のソフトウェア実装

Keccak のソフトウェア実装の性能は，eBASH (ECRYPT Benchmarking of All Submitted Hashes) の “List of SHA-3 finalists measured” に詳細が掲載されている [1]．eBASH とは，ECRYPT（European Network of Excellence for Cryptology）における VAMPIRE lab. のプロジェクト名である．Keccak も eBASH プロジェクトにおけるソフトウェア性能評価の対象として，ベンチマークが実施されている．Keccak に関しては，様々なセキュリティパラメータでベンチマーク評価がなされているが，本報告では，Keccak-512（Keccak$[r = 1088, c = 512]$）に焦点を絞り，考察を行うことにする．

表 1: Keccak$[r = 1088, c = 512]$ のソフトウェア実装性能比較（ [1] の中央値）．単位は clock/byte）．

		メッセージ長 [bytes]					
CPU	Hash	Long	4,096	1,536	576	64	8
amd64 Haswell	Keccak-512	10.60	10.52	10.57	12.01	24.02	189.50
	SHA-256	11.72	11.92	12.68	13.64	25.64	108.75
	SHA-512	7.70	8.26	8.72	9.10	17.55	139.88
armeabi Cortex A15	Keccak-512	36.19	38.05	40.65	48.34	127.12	1012.75
	SHA-256	20.04	21.17	23.13	28.24	91.70	592.12
	SHA-512	44.41	46.50	50.18	54.82	138.55	1094.88

表 1 は [1] の公表結果から，64 bit CPU である「amd 64 Haswell」と，スマートフォン等で使われている 32 bit CPU の「armeabi Cortex A15」のソフトウェア実装の結果をまとめたものである．参考値として，SHA-256 と SHA-512 の性能を並記した．64 bit CPU の「amd64 Haswell」に実装した場合，Keccak は SHA-256 と同等以上の性能となるが，SHA-512 と比べ約 1.5 倍の処理時間を要する．一方，32 bit CPU の「armeabi Cortex A15」に実装した場合は，SHA-512 と同等以上の性能となるが，SHA-256 と比べ約 1.8 倍の処理時間を要する．この性能の優劣は，メッセージ長に大きく左右されないことが分かる．

また，欧州の複数の大学からなる研究チームが，8 bit CPU の AVR における性能評価結果 [8] を報告している．計算能力に乏しい CPU を用いた実装実験では，コード・サイズや，処理に必要となる一時記憶容量について，統一的な条件で比較を行わなければならない．この点において， [8] の結果の取り扱いについては，さらなる検討が必要と思われる．

5.3 Keccak のハードウェア実装

ここでは，Keccak の提案者らによるソフトウェア／ハードウェア実装の概要 [12] に基づき議論を進める．Keccak 提案者らによるハードウェア実装は，High-speed core 向け，Mid-range core 向け及び Low-area coprocessor 向けの 3 つのカテゴリに分けられ，それぞれ回路規模と処理性能が評価されている．High-speed core 向けの実装は，1 サイクルで 1 ラウンド (single round per cycle) あるいは 1 サイクルで複数回 (multiple rounds per cycle) の Keccak-f を実行し，Mid-range core 向けと Low-area coprocessor 向けの実装は，ラウンド関数 1 回に相当する演算を複数サイクルで実行するものである．ここでは，議論を簡単にするために，High-speed core 向けの実装とそれ以外（Mid-range core 向けならびに Low-area coprocessor 向け実装）の 2 つに分けて既存研究の整理し，実装性能の比較を行う．

5.3.1 High-speed core 実装

High-speed core 実装は，組み合せ回路で構成された Keccak-f ラウンドを，ループ・アーキテクチャの基本演算として設計するものである．1 サイクルでラウンド関数である Keccak-f[1600] を 1 回実行するハードウェアを構成した場合，例えば Keccak$[r = 1024, c = 576]$ は合計 24 サイクルで処理され，回路規模は High-speed core 実装の中では最小となる[5]．文献 [12] では，1 サイクルで Keccak$[r = 1024, c = 576]$ のラウンド関数を 1 回あるいは複数回実行するハードウェアの処理性能と回路規模について見積りを与えている[6]．見積りのために回路はいくつかの部品に分られて合成されている．合成には，Synopsys の Design Compiler が用いられ，STM 180 nm のスタンダード・セル・ライブラリの worst case が使用された．以下が [12] で報告された数値である．

1) 1 つのラウンド関数のゲート・サイズは 19 kgate
2) 1 つのラウンド関数の信号遅延時間は 1.1 ns
3) ラウンド関数を除く回路（順序回路や制御回路等）は合計 29 kgate
4) ラウンド関数を除く回路の信号遅延時間は 0.8 ns

これにより，例えば，1 サイクルで Keccak-f[1600] ラウンドを 6 回実行する Keccak$[r = 1024, c = 576]$ ハードウェアのゲート・サイズは 143 kgate でスループットは 34.6 Gbps と見積もることができる．1 サイクルで処理するラウンドの数に比例して，最大信号遅延時間と回路規模は増加するが，上記 4) の遅延時間への影響が小さくなるため，スループットは改善される．ただし，このような見積りはレイアウトが理想的な場合にのみ成立する．実際には回路規模が増大すると，配線容量などの影響により，スループット低下の要因が増える結果となる．

[5] クリティカル・パス遅延を低減するために，データの読み込みに 1 サイクル追加し，合計 25 サイクルとする実装もある [45][7].
[6] Chapter 4, Sect. 4.2, Table 4.1 を参照のこと

第三者による既存研究の多くは，single round per cycle 実装による評価である．以降，文献毎に簡単な説明を加え，表 2〜5 に文献で示された結果をまとめハードウェア性能の比較する．

5.3.2 第三者による High-speed core 実装の性能評価

High-speed core 実装の処理性能は，主にスループットで評価されているが，これまでの文献の多くはデータ転送等によるサイクル数を考慮しておらず理想的なデータ・インタフェースを想定している（つまり，single round per cycle 実装では，24 サイクルで実行が可能としている）．本報告でも，特に断りの無い限り，理想的なデータ・インタフェースを想定することとする[7]．

Stömbergson は文献 [41] で，Keccak の提案者らにより公開されたリファレンス・コードを 4 種類の FPGA に実装し，それぞれ処理性能を評価した．その結果，single round per cycle 実装を行った場合のスループット性能は 5〜10 Gbps 程度であった．

Graz University of Technology の Tillich らは，UMC 180 nm の CMOS スタンダード・セル・ライブラリを用いて Keccak$[r = 1088, c = 512]$ を実装し，56.3 kgates の回路規模で，21.2 Gbps のスループット性能を達成した [45]．彼らの実装では，データのアクセスに 1 サイクル使用しているため，1 ブロックの計算に要するレイテンシは 25 サイクルであった．スループットは 21.2 Gbps と報告されている．回路の合成には Cadence PKS-Shell (v05.16) が用いられ，速度優先の条件で合成を行った（レイアウトは行っていない）．彼らは加えて，同じハードウェア設計を異なる条件で合成し，回路規模とスループットのトレード・オフを調べ，最小の回路規模で約 40 kgates，スループットは 11〜13 Gbps という結果を示した[8]．後に Tillich らは文献 [44] で，UMC 180 nm を用いたレイアウト後の実装結果を示し，21.2 Gbps のスループット性能の妥当性を評価した．

George Mason University (GMU) の Gaj らは，国際会議 CHES2010 での報告 [14] で，7 種類の FPGA 実装結果を示した．Xilinx 社の Virtex-5 と Altera 社の Stratix III でそれぞれ 10.8 Gbps と 13.4 Gbps の高いスループットを実現している．後の文献では，Virtex-6, Cyclone IV 及び Stratix IV がプラットフォームとして用いられ，追加の実験結果が示された [23]．Virtex-6 と Stratix IV においても 13 Gbps のスループットを達成している．FPGA 実装の結果は，文献 [40][39][22][15] で更新されている．

ETH Zurich の Henzen らは CHES2010 で UMC 90 nm を用いた ASIC 実装結果を示した [21]．彼らは，合成／レイアウトの条件として，スループットとして 20 Gbps と 0.2 Gbps を設定し，回路規模は 50.0 kgates と 27.5 kgates でスループット性能は 43.0 Gbps と 6.8 Gbps の結果を得ている．

文献 [7] で Akin らは，FPGA 実装と Synopsys 90 nm を用いた ASIC 実装の結果を示した．彼らは，通常の single round per cycle 実装に加えて，1 ラウンドを複数ステージに分けたパイプライン・アーキテクチャを提案している．一般に，パイプライン・アーキテ

[7] 一般的には，2 章で述べたようにデータ・インタフェースによってはスループットが低下する場合がある．
[8] グラフから読み取った値．

表 2: Keccak-f[1600] のラウンド関数を 1 サイクルで 1 回実行する (single round per cycle) ハードウェアの性能比較〜その 1〜.

文献	ブロック・サイズ [bit]	プラットフォーム (ASIC/FPGA)	回路規模	動作周波数 [MHz]	スループット [Gbps]
[12]	1,024	STM 180 nm	48 kgates	526	22.4
[41]	1,024	Cyclone III	5,842 LEs	123	7.0
		Stratix III	4,550 ALUTs	176	10.0
		Spartan-3A	3,393 slices	85	4.8
		Virtex-5	1,483 slices	118	6.7
[45]	1,088	UMC 180 nm	56.3 kgate	488	21.2
[14]	1,088	Spartan-3	3,326 slices	96	4.4
		Virtex-4	3,343 slices	202	9.2
		Virtex-5	1,229 slices	238	10.8
		Cyclone II	6,239 LEs	165	7.5
		Cyclone III	5,983 LEs	174	7.9
		Stratix II	4,086 ALUTs	199	9.0
		Stratix III	4,458 ALUTs	296	13.4
[23]	1,088	Spartan-3	3,371 slices	109	4.9
		Virtex-4	3,457 slices	230	10.4
		Virtex-5	1,272 slices	283	12.8
		Virtex-6	1,207 slices	286	13.0
		Cyclone II	6,419 LEs	134	6.1
		Cyclone III	6,823 LEs	156	7.1
		Cyclone IV	6,469 LEs	156	7.1
		Stratix II	4,245 ALUTs	210	9.5
		Stratix III	4,213 ALUTs	273	12.4
		Stratix IV	4,219 ALUTs	286	13.0

表 3: Keccak-f[1600] のラウンド関数を 1 サイクルで 1 回実行する (single round per cycle) ハードウェアの性能比較〜その 2〜.

文献	ブロック・サイズ [bit]	プラットフォーム (ASIC/FPGA)	回路規模	動作周波数 [MHz]	スループット [Gbps]
[21]	1,088	UMC 90 nm	50.0 kgates	949	43.0
			27.5 kgates	149	6.8
[7]	1,088	Spartan-3	2,024 slices	81	3.5
		(pipeline)	4,536 slices	338	14.9
		Virtex-2	2,024 slices	137	5.8
		(pipeline)	4,536 slices	342	15.0
		Virtex-4	2,024 slices	143	6.1
		(pipeline)	4,356 slices	509	22.3
		Synopsys 90 nm	10.5 kgates	455	19.3
		(pipeline)	23.2 kgates	1,695	74.4
[9]	1,088	Virtex-5	1,971 slices	196	6.3
[34]	1,024	Virtex-5	1,433 slices	205	8.4 (1.0)
[18]	1,024	UMC 130 nm	35.0 kgates	161	6.6 (0.8)
			47.4 kgates	377	15.5 (1.8)
[44]	1,088	UMC 180 nm	56.7 kgates	267	11.6
			56.3 kgates	488	21.2
[19]	1,024	IBM 130 nm	42.5 kgates	267	10.7
[40]/[39]	1,088	Virtex-5	1,352/1,369 slices	299/297	13.5
		Virtex-5	1,338 slices + 1 BRAM	248	11.3
		Stratix III	4,221 ALUTs	303	13.7
		Stratix III	4,277 ALUTs + 2k Mem	311/307	14.1/13.9

表 4: Keccak-f[1600] のラウンド関数を 1 サイクルで 1 回実行する (single round per cycle) ハードウェアの性能比較〜その 3〜.

文献	ブロック・サイズ [bit]	プラットフォーム (ASIC/FPGA)	回路規模	動作周波数 [MHz]	スループット [Gbps]
[22]/[15]	1,088	Virtex-5	1,395/1,369 slices	不明	12.8/13.3
		Virtex-5	1,980/1,950 slices		15.4/16.1
		Virtex-5	3,849/- slices		12.7/-
		Virtex-6	1,165/1,086 slices		11.8
		Virtex-6	1,446/1,474 slices		16.2/18.8
		Virtex-6	2,785/- slices		13.2/-
		Stratix III	3,909/3,531 ALUTs		13.0/15.5
		Stratix III	4,955/4,810 ALUTs		19.2/20.0
		Stratix III	5,391/- ALUTs		16.0/-
		Stratix IV	4,129/3,471 ALUTs		13.2/16.1
		Stratix IV	4,953/4,294 ALUTs		18.6/21.2
		Stratix IV	5,402/- ALUTs		17.9/-
[20]	1,088	UMC 65nm	46.3 kgates	485	22.0
			80.7 kgates	599	27.2
[33]	1,088	Virtex-5	1,333 slices	230	12.5
		Virtex-6	915 slices	283	13.7
		Virtex-7	1,161 slices	286	13.3

表 5: Keccak-f[1600] のラウンド関数を 1 サイクルで 1 回実行する (single round per cycle) ハードウェアの性能比較〜その 4〜.

文献	ブロック・サイズ [bit]	プラットフォーム (ASIC/FPGA)	回路規模	動作周波数 [MHz]	スループット [Gbps]
[16]	1,088	Stratix III	14,402 ALUTs	212	13.6
	1,088	Virtex-5	2,636 slices	84	5.4
[32]	1,024	STM 90 nm	50.7 kgates	781	33.3
			33.7 kgates	541	23.1
			29.5 kgates	355	15.1
[37]	1,024	Virtex-5	2,640 slices	122	5.2
			3,117 slices	452	7.7
[28]	1,088	Virtex-5	1,305 slices	195	8.5 (5.1)

クチャは，複数のメッセージに対するハッシュ値の計算を高速に実行する際には，非常に有効な方法である．クリティカル・パスの遅延を低減することで，動作周波数を上げることができ，高いスループットが期待できる．ただし，パイプライン・ステージが増すと 1 回のハッシュ処理を細分して処理するため，レイテンシの低下が懸念される．文献 [7] では，1 ラウンドを 5 段に分けた 5 ステージ・パイプライン・アーキテクチャを FPGA 及び ASIC に実装し，回路規模が約 2.5 倍増加したものの，4 倍程度のスループットの向上が得られた．Keccak のパイプライン・アーキテクチャ実装が，アプリケーションによっては有効であることを示した．

University College Cork の Baldwin は，RMIT University と Queen's University Belfast との文献 [9] で，Keccak を Virtex-5 に実装し 6.3 Gbps のスループット性能を得た．

Matsuo らは，Keccak の提案者らが提供しているサンプル・コードを用いて，Virtex-5 上のハードウェア性能の評価を行った [34] ．彼らが行った実装の特徴は，インターフェイスにおけるデータの受け渡し時間を考慮している点にある．多くの文献は，アクセラレータ・モジュール単体の性能を評価しているが，インターフェイスのバンド幅によりシステム全体としてのスループット性能は律速されてしまうことがある．本文献では，データ幅を 16 bit としたバス・インターフェイスを用い，アクセラレータ・モジュールとしての使用を想定している．ハードウェアの性能評価の結果は 1.0 Gbps であった（表 3 の括弧内のスループット値）．理想的なインターフェイスを用いた場合の性能（8.4 Gbps）と比べ，約 1/8 となることが示された．

文献 [18] でも同様の議論がなされている．Virginia Tech の Guo らは，文献 [34] と同じインタフェースを用いて，UMC 180 nm で Keccak のハードウェア実装を行った．合成条件を回路規模最小とした場合に 35.0 kgates の回路規模で 0.8 Gbps， 速度優先とした場合に 47.4 kgates の回路規模で 1.8 Gbps のスループット性能が得られている．いずれもインターフェイスにおけるデータの受け渡し時間が考慮されている．インターフェイスを理想的なものとした場合には，それぞれ 6.6 Gbps と 15.5 Gbps のスループット性能である（表 3 の括弧内のスループット値）．彼らは後に，IBM 180 nm での合成結果として 42.5 kgates の回路規模で 10.7 Gbps のスループット性能を得ている [19].

ETH Zurich の Gürkaynak は GMU の Gaj らとの共同研究で UMC 65nm の実装結果を報告している [20]．回路規模が 46.3 kgates の場合スループット性能は 22.0 Gbps であり，80.7 kgates では 27.2 Gbps であった．レイアウト前後で合成結果に差異があることを示し，正確な見積りにおけるレイアウト実装の必要を明らかにした．

他にも Keccak の高速ハードウェアが実装が多く報告されている [33][16][32][37][28] ．デバイスの微細化によるスループット性能は向上しており，FPGA 実装では 10〜20 Gbps 程度，ASIC 実装では 20 Gbps を超える実装結果が得られている．いずれにせよ，高速実装の課題として，性能を律速しないインタフェースの実装が挙げられる．

5.3.3 Mid-range core/Low-area coprocessor 実装

Mid-range core/Low-area coprocessor 実装は，組み合せ回路で構成された Keccak-f ラウンド処理を複数のサイクルに分けて実装するものである．文献 [12] で述べられている Mid-range core 実装の特徴は，以下のようにまとめられる．

1) ステートを N_b ブロックに分割し，ラウンド関数を複数サイクルに分けて実行
2) N_b の値を大きくするほど，回路規模の低減が可能
3) 高い動作周波数の実現が可能
4) フリップ・フロップから構成されるモジュールにより高速なデータ・アクセスを実現

つまり，High-speed core 実装よりも回路規模を小さくするために，ラウンド関数を細分化し，組み合せ回路を削減していると言い換えることができる．

Low-area coprocessor 実装は，SRAM 等のシステム・メモリの使用を想定しており，フリップ・フロップ数が少なくなる．そのため，Mid-range core 実装と比べ SRAM アクセス数が追加されるため，ラウンド関数の処理により多くのクロック数が必要となる．例えば，Keccak の提案者らによる Virtex-5 への実装では，448 slices の回路規模で，1 ブロックの処理に要するレイテンシは 5,160 サイクルであった [12]．High-speed core と比較し，回路規模を 1/3 程度に抑えて実装できることが示されている．

本報告では, Mid-range core 実装と Low-area coprocessor 実装をまとめて軽量実装 (single round per multiple cycles) と位置づけ，既存研究のハードウェア性能を調査する．

5.3.4 第三者による Mid-range core/Low-area coprocessor 実装の性能評価

Kavun らは，文献 [29] で，130 nm CMOS テクノロジを用いて Keccak の軽量実装を行い，回路規模と処理性能の評価を行った．実装の結果，回路規模は 20.8 kgates となり，レイテンシは 1,200 サイクルであった．後に [30] で，90 nm CMOS テクノロジを用いた性能評価を行った結果，レイテンシは文献 [29] と同じく 1,200 サイクルであったが，回路規模は 15.2 kgates となった．回路規模の指標となるゲート・カウントはライブラリによっては異なり，特に順序回路でのばらつきが大きいため，上記の結果が得られたものと推測する．

Keccak の軽量実装は FPGA でも評価されている [31][27][38][26][28] ．その中でも特徴的なのは，Block RAM を用いて slice 数の低減を実現した [38] や，回路規模とレイテンシのトレード・オフを詳細に調べた [28] である．アプリケーションにより Keccak ハードウェアの柔軟性を示す結果と言える．

表 6: Keccak-f[1600] の軽量実装 (single round per multiple cycles) ハードウェアの性能比較.

文献	ブロック・サイズ [bit]	プラットフォーム (ASIC/FPGA)	回路規模	レイテンシ [cycles]
[12]	1,024	Virtex-5	448 slices	5160
[29]	1,088	130 nm	20.8 kgates	1200
[30]	1,088	90 nm	15.2 kgates	1200
[31]	1,088	Virtex-6	144 slices	2137
[27]	1,088	Virtex-5	393 slices	200
[38]	1,024	Virtex-5	151 slices + 3 BRAM	1062
[26]	1,088	Spartan-3	1665 slices	200
		Spartan-6	420 slices	
		Virtex-6	397 slices	
[28]	1,088	Virtex-5	164 slices	1600
			195 slices	800
			222 slices	400
			301 slices	200
			489 slices	100
			914 slices	50

5.3.5 Keccak のハードウェア実装性能評価のまとめ

図 4, 5 はそれぞれ Virtex-5 と ASIC 実装の結果をまとめたグラフである．横軸は回路規模であり，縦軸は高速実装ハードウェアに対してはスループットを，軽量実装ハードウェアに対してはレイテンシを示す．高速実装ハードウェアではスループット性能が最も重要である．Virtex-5 実装の結果から，多くの実装結果において回路規模が 1,000〜2,000slice であり，スループット性能は 6〜16 Gbps であった．ASIC 実装の結果はばらつきが大きいが，回路規模は 30〜60 kgates でスループット性能はおおよそ 10〜40 Gbps であった．一方，軽量実装の場合，回路規模が最重要項目であるが，軽量化に伴う処理性能の低下を評価するためにレイテンシを併せて指標として用いる．例えば，Jungk らによる Virtex-5 実装の報告 [28] から，軽量化に伴いレイテンシが増加する傾向が読み取れる．高速実装と比べて，柔軟に回路規模の削減が可能であり，この点は，Keccak ハードウェア実装の大きな特長のひとつと言える．

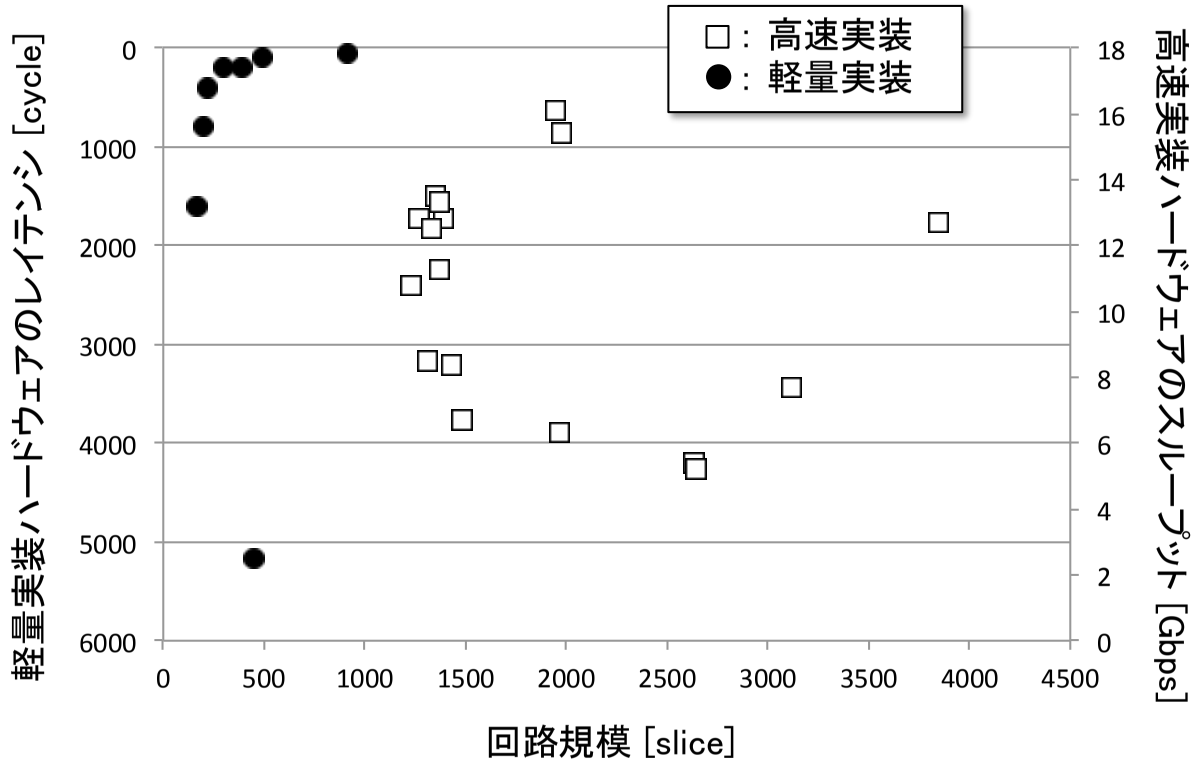


図 4: 既存研究における Virtex-5 FPGA を用いた高速実装と軽量実装の性能結果の比較.

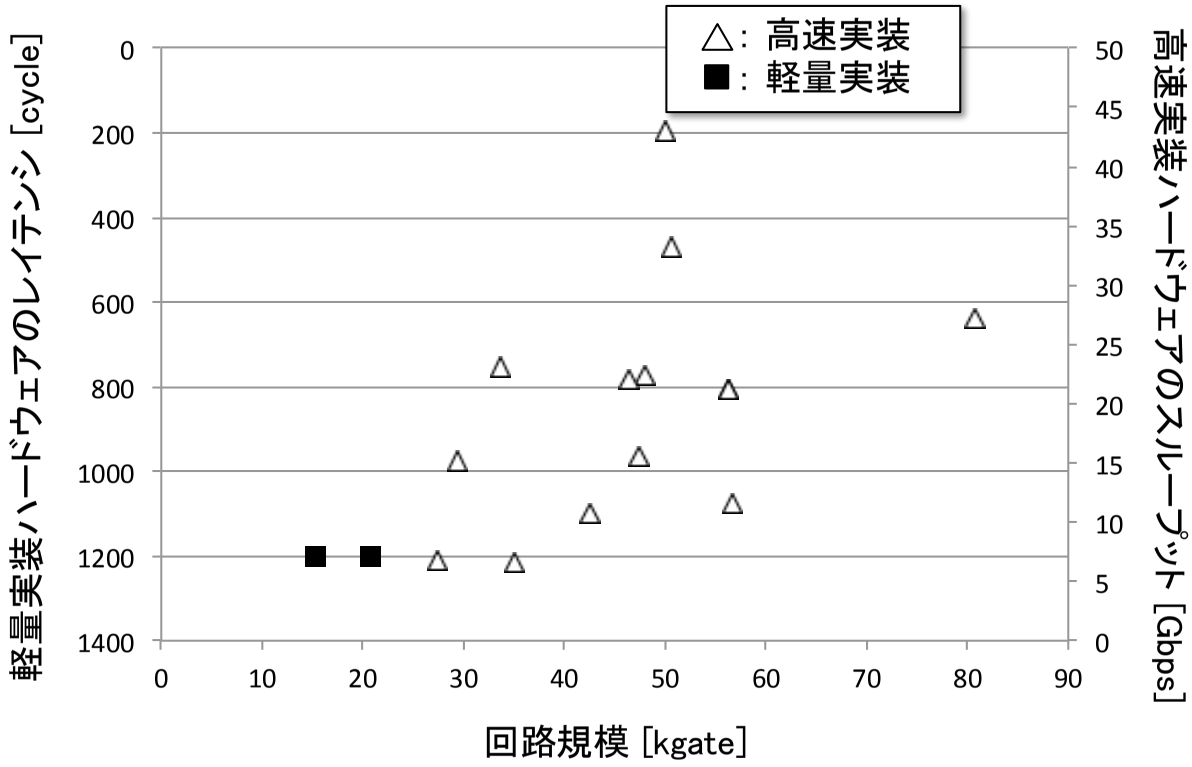


図 5: 既存研究における CMOS スタンダード・セルを用いた高速実装と軽量実装の性能結果の比較.

5.4 Keccak のハードウェア実装に対するサイドチャネル耐性

国際会議 HASP2012 で Bertoni らは，Keccak ハードウェア実装のサイドチャネル対策について議論した [10]．彼らは秘密分散法の考え方に基づき，Keccak ハードウェア回路を 3 つに分割し（シェア数 3），サイドチャネル情報と Mac-Keccak 処理における中間処理値との相関を低くし，鍵復元攻撃を困難にすることに成功した．評価に用いられた Mac-Keccak に対する攻撃は，秘密鍵 Key に対して K =Keccak-f(Key$||0^c$) の計算処理中に漏洩するサイドチャネル情報から K の復元を試みるものである．鍵がレート r より小さい場合，メッセージ M の一部が Keccak-f の入力の一部となるが，同様の解析手法で安全性評価ができるとしている．

Technische Universität Darmstadt の Zonhner らの [46] や Virginia Tech の Taha らの [43][42] では，対策が施されていない実装ではあるが，攻撃研究の報告がある．いずれも，MAC-Keccak の実装に対してサイドチャネル解析攻撃を行い，鍵の復元が可能であることを示すものである．Zonhner らによる国際会議 DATE2012 で発表された [46] では，MAC-Keccak のメッセージ入力時の XOR 演算と θ ステップにおける XOR 演算を利用し，鍵復元の可能性を議論している．一般に，線形関数である多ビットの XOR 処理を選択関数とする場合，漏洩するサイドチャネル情報から中間処理値を復元することは難しいが，文献 [46] における ATMega-256-1 を用いた実験の結果，鍵長やパラメータに制限[9]があるものの，鍵の復元が可能であるとしている．

Taha らの国際会議 IWSEC2013 の発表 [42] では，任意の鍵長の MAC-Keccak から鍵復元が可能であることを示した．彼らは，Keccak[$r = 1088, c = 512$] を 32 bit CPU に実装し，鍵長を $768, 832, 896, 960, 1024$ bit に分けて安全性評価を行った．鍵がレート r より小さいため，メッセージの一部 M が Keccak-f の入力の一部となり，攻撃者は K =Keccak-f(Key$||M$) を処理する計算により生じるサイドチャネル情報を利用して，鍵の復元を試みるものである．鍵長により攻撃手順は異なるが，非線形処理のみだけでなく XOR 処理に生じるサイドチャネル情報を利用している．結論として，鍵長が長くなるほど鍵復元の成功にはより多くのサイドチャネル情報が必要となるものの，いずれの鍵長でも数万波形程度のサイドチャネル情報で鍵復元攻撃が可能であることを示した．

Katholieke Universiteit Leuven/University of Twente の Bilgin らは，秘密分散に基づく Threshold Implementation (TI) によるサイドチャネル攻撃対策を行い，1 次差分電力解析に対して，有効な対策となることを国際会議 COSADE2013 で示した [13]．3 つの異なるプロットフォーム（UMC 180 nm，UMC 130 nm 及び NANGATE 45 nm）を用いて，サイドチャネル対策によるコストの増加を見積もっている．高速実装ハードウェアの場合の回路規模は，シェア数 3 で約 4 倍となり，シェア数 4 で約 5 倍が必要となる．また，軽量実装ハードウェアの場合も同様に，シェア数 3 で約 4 倍となり，シェア数 4 で約 5 倍の回路規模が必要としている[10]．

[9] 5.1.1 章で述べた b と r に対して，鍵長が $\frac{b}{5}$ 以下でかつビットレート・データを $\frac{b}{5}$ 以上復元できる場合としている．

[10] 詳細は [13] の Table 1 を参照されたい．

本報告では紹介しなかったが，今後はスポンジ構造を有する Keccak に対する攻撃と対策の研究が活発に進められ，サイドチャネル解析研究を牽引するものと考える．故障を利用した解析攻撃も考えられる．これまで，攻撃対象として SPN 構造を有する AES 暗号が用いられ，サイドチャネル解析の研究が発展してきた．暗号実装上の安全性評価において，SPN 構造とスポンジ構造による本質的な違いを理解することが必要と考える．

6　まとめ

本報告は，FIPS PUB 180-4 [36] で仕様が策定されているハッシュ関数 SHA-224，SHA-512/224 及び SHA-512/256 について，関連文献を調査し，実装性能評価に関する考察を行った．また，FIPS Draft が未公開の SHA-3 については Keccak の仕様に基づき，ハードウェア実装性能評価をの研究動向調査を行った．

SHA-224 に関する調査の結果，SHA-256 とほぼ同等の性能であると考える．最終データの転送に要する時間を短縮できるため，僅かではあるがレイテンシの短縮が達成できることが期待できる．さらに，ハッシュ値のメモリ容量を削減できるため，数多くのハッシュ値を保存するようなアプリケーションにおいて，コスト削減が期待できる．

SHA-512/224 と SHA-512/256 に関する調査の結果, SHA-224 や SHA-256 と比べ, 64 bit CPU におけるソフトウェア実装性能において一定の優位性を有すると考える．これは，64 bit 演算を高速で行う CPU の登場によるもので，今後の暗号アルゴリズムの設計や実装研究に大きな意味を持つと考える．

Keccak のソフトウェア実装に関する文献調査の結果，64 bit CPU の場合の処理性能は，SHA-256 と同等以上だが，SHA-512 より低く，32 bit CPU の場合は，SHA-512 と同等以上だが，SHA-256 より低くいことが分かった．SHA-256 と SHA-512 だけの比較ではあるが，64 bit CPU，32 bit CPU のどちらでも一定の性能が出るように設計されている．

Keccak のハードウェア実装に関する調査では，高速実装／軽量実装に分けて，回路規模に対するスループット，レイテンシを調べた．プラットフォームとして最も用いられていた Xilinx 社の Virtex-5 の実装結果では，回路規模が 1,000〜2,000 slices で 6〜16 Gbps の性能であった．SHA-256 と比べ，回路規模は増加（約 2〜3 倍）するが，スループット性能は 4 倍程度，レイテンシは約 1/2 の時間を実現している [32]．Keccak の ASIC 実装による性能評価では，FPGA よりも高いスループット性能を有することが分かった．また，軽量実装については，数 100 slices で実装可能であることが分かった．

今後，SHA-3 Keccak は，次第に市場に普及すると思われる．Keccak は，柔軟な実装が可能である点が最大の魅力であるが，数 kgates 程度で実装可能な既存の軽量暗号と比べ，比較的回路規模が大きい．軽量実装に向けた実装研究は進展すると思われるが，Keccak の仕様を効率良く実現する画期的な処理アルゴリズムが登場することは考えにくい．高速実装においては，半導体プロセスの微細化に伴う性能向上が幾分期待できる．また，Keccak アクセラレータを最大限活用できるシステム・アーキテクチャ設計については，継続的な研究が必要と考える．

サイドチャネル攻撃対策の研究については，それほど多くの文献を見つけることはできなかったが，対策が施されていない MAC-Keccak 実装に対する鍵復元攻撃攻撃は可能であることが分かった．一方で，ハードウェアを複数の回路に分割することで，MAC-Keccak の鍵復元攻撃耐性の向上に繋がる報告が見られた．回路の分割数によりサイドチャネル対策のコスト増加は，シュア数 3 の場合で約 4 倍，シェア数 4 の場合で約 5 倍の回路規模が必要となり，現実的な解としては改善が必要と考える．

本報告では触れなかったが，文献調査の段階で GPU を用いた高速実装評価が数件見られた．また Keccak が広く普及すれば，汎用 CPU のインストラクション・セットが拡張されることも考えられる．様々なプラットフォームでの Keccak の実装性能に関して，今後の研究動向を注視する必要がある．実装評価とサイドチャネル攻撃対策についての研究も活発に進められることが予想される．

参考文献

[1] List of SHA-3 finalists measured. `http://bench.cr.yp.to/primitives-sha3.html`.

[2] The Second SHA-3 Candidate Conference, Santa Barbara, USA, August 23–24, 2010. `http://csrc.nist.gov/groups/ST/hash/sha-3/Round2/Aug2010/`.

[3] ECRYPT II Hash Workshop 2011, Tallinn, Estonia, May 19–20, 2011. `http://www.ecrypt.eu.org/hash2011/`.

[4] The Third SHA-3 Candidate Conference, Washington, DC, USA, March 22–23, 2012. `http://csrc.nist.gov/groups/ST/hash/sha-3/Round3/March2012/`.

[5] TENTATIVE SHA-3 STANDARD (FIPS XXX) DEVELOPMENT TIMELINE, Dec. 2013. `http://csrc.nist.gov/groups/ST/hash/sha-3/timeline_fips.html`.

[6] I. Ahmad and A.S. Das. Hardware implementation analysis of SHA-256 and SHA-512 algorithms on FPGAs. *Journal of Computers and Electrical Engineering*, 31(6):345–360, 2005.

[7] A. Akin, A. Aysu, O.C. Ulusel, and E. Savaş. Efficient Hardware Implementations of High Throughput SHA-3 Candidates Keccak, Luffa and Blue Midnight Wish for Single- and Multi-message Hashing. In *Proceedings of the 3rd International Conference on Security of Information and Networks, SIN 2010*, pages 168–177. ACM, 2010.

[8] J. Balasch, B. Ege, T. Eisenbarth, B. Gérard, Z. Gong, T. Güneysu, S. Heyse, S. Kerckhof, F. Koeune, T. Plos, T. Pöppelmann, F. Regazzoni, F.-X. Standaert, G. Van Assche, R. Van Keer, L. Van Oldeneel tot Oldenzeel, and I. Von Maurich. Compact Implementation and Performance Evaluation of Hash Functions in ATtiny Devices. volume 7771 of *Lecture Notes in Computer Science*, pages 158–172. Springer, 2012.

[9] B. Baldwin, N. Hanley, M. Hamilton, L. Lu, A. Byne, M. O'Neill, and W.P. Marnane. FPGA Implementations of the Round Two SHA-3 Candidates. In *Conference Record of [2].*

[10] G. Bertoni, J. Daemen, N. Debande, T.-H. Le, M. Peeters, and G.V. Assche. Power analysis of hardware implementations protected with secret sharing. In *45th Annual IEEE/ACM International Symposium on Microarchitecture, MICRO 2012*, pages 9–16. IEEE Computer Society, 2012.

[11] G. Bertoni, J. Daemen, M. Peeters, and G.V. Assche. The Keccak reference (Version 3.0), 2011. `http://keccak.noekeon.org/`.

[12] G. Bertoni, J. Daemen, M. Peeters, G.V. Assche, and R.V. Keer. Keccak implementation overview (version 3.2), 2012. `http://keccak.noekeon.org/`.

[13] B. Bilgin, S. Nikova, V. Rijmen, V. Nikov, J. Daemen, and G.V. Assche. Efficient and First-Order DPA Resistant Implementations of KECCAK. In *12th Smart Card Research and Advanced Application Conference, CARDIS 2013*, Lecture Notes in Computer Science. Springer, 2013.

[14] K. Gaj, E. Homsirikamol, and M. Rogawski. Fair and Comprehensive Methodology for Comparing Hardware Performance of Fourteen Round Two SHA-3 Candidates Using FPGAs. In S. Mangard and F.-X. Standaert, editors, *Cryptographic Hardware and Embedded Systems, CHES 2010*, volume 6225 of *Lecture Notes in Computer Science*, pages 264–278. Springer, 2010.

[15] K. Gaj, E. Homsirikamol, M. Rogawski, R. Shahid, and M.U. Sharif. Comprehensive Evaluation of High-Speed and Medium-Speed Implementations of Five SHA-3 Finalists Using Xilinx and Altera FPGAs. In *Conference Record of [4]*.

[16] A. Gholipour and S. Mirzakuchaki. Throughput Optimum Architecture of KECCAK Hash Function. *International Journal of Computer and Electrical Engineering*, 4(6):937–939, 2012.

[17] S. Gueron, S. Johnson, and J. Walker. SHA-512/256. IACR Cryptology ePrint Archive, Report 2010/548, 2010.

[18] X. Guo, S. Huang, L. Nazhandali, and P. Schaumont. Fair and Comprehensive Performance Evaluation of 14 Second Round SHA-3 ASIC Implementations. In *Conference Record of [2]*.

[19] X. Guo, M. Srivastav, S. Huang, L. Nazhandali, and P. Schaumont. Silicon Implementation of SHA-3 Finalists: BLAKE, Grøstl, JH, Keccak and Skein. In *Workshop Record of [3]*.

[20] F. Gürkaynak, K. Gaj, B. Muheim, E. Homsirikamol, C. Keller, M. Rogawski, Hubert Kaeslin, and J.-P. Kaps. Lessons Learned from Designing a 65nm ASIC for Evaluating Third Round SHA-3 Candidates. In *Conference Record of [4]*.

[21] L. Henzen, P. Gendotti, P. Guillet, E. Pargaetzi, M. Zoller, and F.K. Gürkaynak. Developing a Hardware Evaluation Method for SHA-3 Candidates. In S. Mangard and F.-X. Standaert, editors, *Cryptographic Hardware and Embedded Systems,*

CHES 2010, volume 6225 of *Lecture Notes in Computer Science*, pages 248–263. Springer, 2010.

[22] E. Homsirikamol, M. Rogawski, and K. Gaj. Comparing Hardware Performance of Round 3 SHA-3 Candidates using Multiple Hardware Architecture in Xilinx and Altera FPGAs. In *Workshop Record of [3]*.

[23] E. Homsirikamol, M. Rogawski, and K. Gaj. Comparing Hardware Performance of Fourteen Round Two SHA-3 Candidates Using FPGAs. IACR Cryptology ePrint Archive, Report 2010/445, 2010.

[24] M. Juliato and C. Gebotys. FPGA implementation of an HMAC processor based on the SHA-2 family of hash functions. University of Waterloo, Tech. Rep., 2011.

[25] Y. Jun, D. Jun, L. Na, and G. Yixiong. FPGA Implementation of SHA-224/256 Algorithm Oriented Digital Signature. In *Challenges in Environmental Science and Computer Engineering, CESCE 2010*, pages 63–66. IEEE, 2010.

[26] B. Jungk. Evaluation Of Compact FPGA Implementations For All SHA-3 Finalists. In *Conference Record of [4]*.

[27] B. Jungk and J. Apfelbeck. Area-Efficient FPGA Implementations of the SHA-3 Finalists. In P.M. Athanas, J. Becker, and R.Cumplido, editors, *2011 International Conference on Reconfigurable Computing and FPGAs, ReConFig 2011*, pages 235–241. IEEE Computer Society, 2011.

[28] B. Jungk and M. Stöttinger. Among slow dwarfs and fast giants: A systematic design space exploration of KECCAK. In *2013 8th International Workshop on Reconfigurable and Communication-Centric Systems-on-Chip (ReCoSoC)*, pages 1–8. IEEE, 2013.

[29] E.B. Kavun and T. Yalçin. A Lightweight Implementation of Keccak Hash Function for Radio-Frequency Identification Applications. In S.B. Örs Yalçin, editor, *Radio Frequency Identification: Security and Privacy Issues - 6th International Workshop, RFIDSec 2010*, volume 6370 of *Lecture Notes in Computer Science*, pages 258–269. Springer, 2010.

[30] E.B. Kavun and T. Yalcin. On the Suitability of SHA-3 Finalists for Lightweight Applications. In *Conference Record of [4]*.

[31] S. Kerckhof, F. Durvaux, N. Veyrat-Charvillon, F. Regazzoni, G.M. De Dormale, and F.-X. Standaert. Compact FPGA Implementations of the Five SHA-3 Finalists. In *Workshop Record of [3]*.

[32] M. Knežvić, K. Kobayashi, J. Ikegami, S. Matsuo, A. Satoh, Ü. Koçabas, J. Fan, T. Katashita, T. Sugawara, K. Sakiyama, I. Verbauwhede, K. Ohta, N. Homma, and T. Aoki. Fair and Consistent Hardware Evaluation of Fourteen Round Two SHA-3 Candidates. *IEEE Trans. VLSI Syst.*, 20(5):827–840, 2012.

[33] K. Latif, M.M. Rao, A. Aziz, and A. Mahboob. Efficient Hardware Implementations and Hardware Performance Evaluation of SHA-3 Finalists. In *Conference Record of [4].*

[34] S. Matsuo, M. Knežvić, P. Schaumont, I. Verbauwhede, A. Satoh, K. Sakiyama, and K. Ota. How Can We Conduct Fair and Consistent Hardware Evaluation for SHA-3 Candidate. In *Conference Record of [2].*

[35] R.P. McEvoy, F.M. Crowe, C.C. Murphy, and W.P. Marnane. Optimisation of the SHA-2 Family of Hash Functions on FPGAs. In *2006 Emerging VLSI Technologies and Architectures, ISVLSI 2006*, pages 317–322. IEEE, 2006.

[36] National Institute of Standards and Technology. Secure Hash Standard (SHS) (FIPS PUB 180-4), March 2012.

[37] F.D. Pereira, E.D.M. Ordonez, I.D. Sakai, and A.M. de Souza. Exploiting Parallelism on Keccak: FPGA and GPU Comparison. *Parallel & Cloud Computing*, 2(1):1–6, 2013.

[38] I. San and N. At. Compact Keccak Hardware Architecture for Data Integrity and Authentication on FPGAs. *Inf. Sec. J.: A Global Perspective*, 21(5):231–242, 2012.

[39] R. Shahid, M.U. Sharif, M. Rogawski, and K. Gaj. Use of Embedded FPGA Resources in Implementations of 14 Round 2 SHA-3 Candidates. In R. Tessier, editor, *2011 International Conference on Field-Programmable Technology, FPT 2011*, pages 1–9. IEEE, 2011.

[40] M.U. Sharif, R. Shahid, M. Rogawski, and K. Gaj. Use of Embedded FPGA Resources in Implementations of Five Round Three SHA-3 Candidates. In *Workshop Record of [3].*

[41] J. Strömbergson. Implementation of the Keccak Hash Function in FPGA Devices, 2009. `http://www.strombergson.com/files/Keccak_in_FPGAs.pdf`.

[42] M. Taha and P. Schaumont. Differential Power Analysis of MAC-Keccak at Any Key-Length. In K. Sakiyama and M. Terada, editors, *Advances in Information and Computer Security - 8th International Workshop on Security, IWSEC 2013*, volume 8231 of *Lecture Notes in Computer Science*, pages 68–82. Springer, 2013.

[43] M. Taha and P. Schaumont. Side-channel Analysis of MAC-Keccak. In *IEEE International Symposium on Hardware-Oriented Security and Trust, HOST 2013*, pages 125–130. IEEE, 2013.

[44] S. Tillich, M. Feldhofer, M. Kirschbaum, T. Plos, J.-M. Schmidt, and A. Szekely. Uniform Evaluation of Hardware Implementations of the Round-Two SHA-3 Candidates. In *Conference Record of [2]*.

[45] S. Tillich, M. Feldhofer, M. Kirschbaum, T. Plos, J.-M. Schmidt, and A. Szekely. High-Speed Hardware Implementations of BLAKE, Blue Midnight Wish, CubeHash, ECHO, Fugue, Grøstl, Hamsi, JH, Keccak, Luffa, Shabal, SHAvite-3, SIMD, and Skein. IACR Cryptology ePrint Archive, Report 2009/510, 2009.

[46] M. Zohner, M. Kasper, M. Stöttinger, and S.A. Huss. Side Channel Analysis of the SHA-3 Finalists. In *Proceedings of Design, Automation and Test in Europe Conference, DATE 2012*, pages 1012–1017, 2012.