ストリーム暗号 *Enocoro*

評価書

株式会社 日立製作所

2010 年 2 月 2 日

目次

1 序

本稿は、ストリーム暗号 *Enocoro* の評価書である。ストリーム暗号の実装性能や安全性は、鍵ストリームを生成する疑似乱数生成器によって特徴付けられる。以下、本稿では、*Enocoro* の鍵ストリームを生成する疑似乱数生成器について評価した結果を紹介する。また、本稿では、ストリーム暗号と疑似乱数生成器を特に区別せず、どちらも *Enocoro* と呼ぶ。

一般に、暗号プリミティブにおいては、十分な評価が行われた後に、はじめて安全である (良い暗号である) と認められる。本評価書の目的は、*Enocoro* に関する安全性および実装に関する評価結果をまとめることで、*Enocoro* が「十分に評価された」「良い暗号」であると主張することにある。また、*Enocoro* に興味を持った人が、自分で評価を行う際に、本稿が良きサーベイとなっていることを期待する。

以下、本稿の構成を述べる。本稿では、まず 2 節で *Enocoro* の設計方針について述べる。次に、3 節で *Enocoro* の安全性評価に関する結果を紹介する。4 節では、*Enocoro* のソフトウェアおよびハードウェア実装評価の結果を紹介する。また、5 節で、既存の暗号との比較を試み、*Enocoro* の存在意義について論じる。以上の評価内容を以って、我々は *Enocoro* が「十分に良い」ストリーム暗号であると信じる。

なお、本稿では *Enocoro*-128v2 の仕様、および仕様書 [20] で定義されている記号に関する知識を前提としている。また、一般的な安全性評価技術については、本稿で改めて厳密な定義を紹介することはせず、その概要に触れるだけに留めた。詳細については、末尾に挙げた文献を参照いただきたい。

2 設計の指針

Enocoro はハードウェアでの軽量性を目標として設計されたストリーム暗号向けの疑似乱数生成器であり、Daemen と Clapp が 1998 年に提案した Panama [7] を源流とする。*Enocoro* はハードウェア向けの疑似乱数生成器ではあるが、ソフトウェアでの実装可能性を保つため、*Enocoro* のアルゴリズムはバイト単位処理の組み合わせで構成されている。これは、eSTREAM [9] の portfolio に掲載されているハードウェア向け疑似乱数生成器とは異なる *Enocoro* の特徴である。バイト単位の処理構造は、ブロック暗号 (特に AES [11]) の安全性評価を通じて育まれてきた truncation 評価技術を適用できるという利点もあり、安全性評価のコストを比較的小さく抑えることができるという点でも有意義であると考える。また、要素関数の選択では、軽量性を最優先事項としている。

2.1 内部状態

Enocoro は、ハードウェア実装における軽量性を実現するために、内部状態を最小限に留めている。実際、*Enocoro*-128v2 では、128 ビットの鍵長に対して、Time-Memory-Trade-Off 攻撃が定める最小の内部状態 (256 ビット) をわずかに超える 272 ビットの内部状態を持つために、$n_b = 32$ とした。

2.2 Sbox

Enocoro は、MUGI [19] と同じく非線形要素として 8 ビット Sbox を採用している。*Enocoro* の Sbox は AES や MUGI とは異なり、4 ビット Sbox と線形変換の合成として定義される。このような構成では、最大差分確率や最大線形確率が最良となる Sbox を得ることはできないが、Sbox1 個当たりの実装コストを削減することができる。

2.3 拡散構造

PANAMA や MUGI では、非線形処理を行う ρ 関数は、SPN (Substitution-Permutation Network) 構造を採用している。これに対し、*Enocoro* は線形変換 L を Sbox S_8 でサンドイッチする SPS (Substitution-Permutation-Substitution) 構造を採用しており、これにより効率的に状態更新関数の非線形性を高めることができると考える。さらに、線形相関攻撃に対して強度が変わらないように SPS 構造を等価変形することで、処理の並列性を高め、ハードウェア実装における高い処理性能を実現している。

2.4 パラメータの選択

Enocoro の結線構造はパラメータ化されている。*Enocoro*-128v2 の結線構造は、線形特性を利用した乱数識別攻撃に対する耐性を最適化するように選ばれている。

2.5 バージョンによるアルゴリズムの差違

Enocoro のアルゴリズムは、これまでにいくつかのバージョンが公開されてきた。本稿で取り扱うのは基本的に *Enocoro*-128v2 であるが、混乱を避けるため、他のバージョンとの差違について表 1 に簡単にまとめておく。これまでに公開された *Enocoro* のパラメータ $n_b, k_1, k_2, k_3, k_4, q_1, p_1, q_2, p_2, q_3, p_3$ は、いずれのバージョンも以下の条件を満たす。

- $1 \leq i \leq 3$ について $q_i = k_i$ を満たす。
- $1 \leq i \leq 3$ について $p_i = k_{i+1} - 1$ を満たす。

この条件に従えば、λ 関数は ρ 関数への入力位置 $k_1, \ldots, k_4$ を決めれば一意に決定される。そこで、表 1 では一部のパラメータを省略して n_b, k_1, k_2, k_3, k_4 のみを記す。

表 1　*Enocoro* のバージョンとパラメータ

鍵長 (ビット)	バージョン	GF(2^8) の定義多項式	パラメータ $(n_b; k_1, k_2, k_3, k_4)$	初期化	
				ラウンド数	カウンタ
80	1.0 [30]	$x^8 + x^4 + x^3 + x + 1$	$20; 1, 4, 6, 16$	40	無
128	1.0 [30]	$x^8 + x^4 + x^3 + x + 1$	$32; 1, 5, 15, 29$	64	無
	1.1 [27]	$x^8 + x^4 + x^3 + x + 1$	$32; 2, 7, 16, 29$	64	無
	2.0	$x^8 + x^4 + x^3 + x^2 + 1$	$32; 2, 7, 16, 29$	96	有

3 安全性評価

3.1 要素関数の基本的な性質

Enocoro の 8 ビット Sbox $_8$ の暗号学的な性質は以下の通りである。

- 最大差分確率：$2^{-4.678}$
- 最大線形確率：2^{-4}
- 代数次数：6

また、線形変換 L の分岐数は 3 である。

3.2 Time-Memory-Trade-Off 攻撃

軽量なハードウェア実装を考えるならば、疑似乱数生成器の内部状態は小さい方が良い。しかし、安全性の面からすれば、内部状態はある程度大きいことが必要とされる。Babbage [1] と Golić [12] は独立に疑似乱数生成器に対する Time-Memory-Trade-Off 攻撃 (TMTO) を検討し、疑似乱数生成器の内部状態は少なくとも鍵長の 2 倍必要であることを明らかにした。したがって、鍵長が 128 ビットの疑似乱数生成器が TMTO に対する耐性を保証するためには、256 ビット以上の内部状態が必要である。*Enocoro*-128v2 の内部状態は 272 ビットであるので、単純な Babbage-Golić の TMTO を適用するために必要な計算量は $2^{272/2} = 2^{136}$ 程度であると見積もられる。

3.3 推測決定攻撃

推測決定攻撃 (Guess and Determine attack) は、Bleichenbacher と Patel が提案した攻撃法で [4]、特にバイト (もしくはワード) 単位処理を行うストリーム暗号に対して有効である。推測決定攻撃では、内部状態の全数探索において、出力情報を利用して推測する状態の冗長性を取り除くことで、探索空間をできるだけ少なくしている。

推測決定攻撃の一般的な攻撃シナリオでは、まず攻撃の起点としていくつかのワードの値を推測し、状態遷移と出力ワードとを通じて、推測が正しいかどうかを検証する。井手口と渡辺は、推測するワードをランダムに選択し、検証や推測ワードの追加を自動的に行うツールを開発し、*Enocoro*-128v1.1 を含む疑似乱数生成器を評価した [22]。彼らの評価では、*Enocoro*-128v1.1 に対する最良の結果は推定ビット数が 144 ビットのものであり、攻撃に必要な計算量は 2^{144} 程度となる。

Enocoro-128v2 の状態更新関数は *Enocoro*-128v1.1 と同じなので、[22] による評価結果はそのまま *Enocoro*-128v2 に適用できる。したがって、*Enocoro*-128v2 は推測決定攻撃に対して安全であると考えられる。

3.4 分割統治攻撃

分割統治攻撃は、内部状態 S と状態更新関数 $Next$ が以下のように分割できる場合に適用される攻撃である。

$$(a_1^{(t+1)}, \ldots, a_N^{(t+1)}) = S^{(t+1)} = Next(S^{(t)}) = (f_1(a_1^{(t+1)}), \ldots, f_N(a_N^{(t+1)})).$$

PKSG は、内部状態 S を 2 つのレジスタ a, b に分割しているが、その状態更新関数はいずれも 2 つのレジスタを入力としている。

$$a^{(t+1)} = \rho(a^{(t)}, b^{(t)}),$$
$$b^{(t+1)} = \lambda(a^{(t)}, b^{(t)}).$$

このような理由から、*Enocoro* に対して分割統治攻撃を適用することは困難であると考える。

3.5 代数的な攻撃

近年、XSL 攻撃 [6] や Gröbner 基底の計算による代数的攻撃の計算量削減手法が一部のストリーム暗号に対して有効な攻撃法であることがわかってきた。しかし、その主な適

用先は線形フィードバックシフトレジスタを使ったフィルタ型生成器やコンバイナ型生成器であり、非線形な状態更新関数を持つようなストリーム暗号への適用はあまり知られていない。*Enocoro* の状態更新関数は非線形関数であることから、XSL 攻撃などの適用は困難であると考える。

3.6 出力列の線形相関を利用した乱数識別攻撃

出力列の線形相関を利用した乱数識別攻撃 (Linear Distinguishing Attack: LDA) は、線形攻撃 [15] の応用であり、疑似乱数生成器の出力の乱数性を理論的に評価することができる方法である。LDA では、出力列の任意の有限線形和を評価対象とし、その値の 0/1 の分布を調べる。値が 1 となる確率を p とすれば、$2|p-1/2|^2$ ビット程度の出力列を観測することで、評価対象の出力列と真の乱数を区別することができる。*Enocoro* のようにバイト (もしくはワード) 単位の処理で構成される疑似乱数生成器の場合には、LDA の攻撃モデルにおける最大線形特性確率の上界を truncation 評価で求めることができる。評価方法の詳細については、[17] を参照されたい。

LDA 耐性という観点では、*Enocoro*-128v2 と *Enocoro*-128v1.1 のアルゴリズムは等価である。また、*Enocoro*-128v1.1 については、武藤らによる評価結果 [27] が知られており、2^{144} 以上の計算量が必要である。したがって、*Enocoro*-128v2 は LDA に対して安全であると考えられる。

3.7 初期化の安全性

Daemen らは、IV を用いた同期の仕組みを持つ疑似乱数生成器について、出力の乱数性だけではなく、初期化関数の評価が重要であることを指摘した [8]。この節では、差分攻撃と線形攻撃を用いて *Enocoro*-128 を評価した結果について紹介する。

Enocoro の初期化に関する最初の研究は武藤らによって行われた [27]。彼らは *Enocoro*-80 の線形攻撃に対する耐性を概評価手法を用いて評価した。彼らの報告によれば、*Enocoro*-80 の初期化を線形攻撃でランダム関数と識別するためには 2^{144} の既知 IV が必要である。鴻巣らは、武藤らと同様の手法を *Enocoro*-128v1.1 の初期化処理に適用し、差分攻撃、線形攻撃に対する耐性を評価した [24]。鴻巣らによる報告では、(差分／線形攻撃で) 初期化関数の非乱数性を観測するためには、それぞれ $2^{177.8}$ および 2^{216} 程度の既知 IV が必要とされた。岡本らは、ブロック暗号における鍵推定の手法を適用することで、攻撃に必要となる既知 IV 数を削減できることを指摘し、*Enocoro*-80 について $2^{70.2}$ 程度の選択 IV を使えば、差分攻撃を用いて秘密鍵を求めることができる可能性を指摘した [28]。

岡本らと同じ手法を *Enocoro*-128v1.1 に適用すると、攻撃に必要な既知 IV 数の下限は $2^{102.9}$ 程度となる [31]。ただし、岡本らの手法では、攻撃の際に多くのビットについて鍵推定を行う必要があり、この手法を用いて秘密鍵を復元するために必要な総合的な攻撃計算量は、今のところ 2^{128} を上回っている。また、*Enocoro*-128v1.1 の IV 長は 64 ビットであり、実際に $2^{102.9}$ の既知 IV を得ることはできない。さらに、[31] では、初期化が 80 段と 96 段に変更した場合についても同様の評価を行い、攻撃に必要な既知 IV 数がそれぞれ $2^{140.3}$、$2^{177.8}$ 程度となることを確認している。*Enocoro*-128v2 の初期化段数は 96 段なので、岡本らの攻撃に対しても十分な安全性を持っていると考えられる。

3.8 関連鍵攻撃モデルにおける弱鍵の探索

この節では、[31] で提案された *Enocoro*-128v1.1 に対する関連鍵攻撃を利用した弱鍵の探索方法を紹介する。また、*Enocoro*-128v2 がこの攻撃に対して施した対策について述べる。なお、関連鍵攻撃の定義は Bellare の定式化 [3] に拠っており、通常の鍵差分攻撃に比べると非現実的な鍵ペアを必要とすることに留意されたい。

3.8.1 位相ずれした出力列

鍵 K, IV I に対応する時刻 t の出力を $z(K,I)^{(t)}$ と表す。渡辺らは、*Enocoro*-80 の初期化関数の特性として、$z(K,I)^{(t)} = z(K',I')^{(t+T)}$ を満たすような鍵、IV と時刻のペア (以下 $T-$slid pair と呼ぶ) が存在することを指摘し、発見に必要な計算量が 2^{32} 回の状態更新関数呼び出しに相当すると見積もった [18]。

Enocoro-128v1.1 も *Enocoro*-80 と同様の性質を持ち、K, I を初期値として T 回更新したときに、$b_{24}, \ldots, b_{31},\ a_0, a_1$ が初期定数と一致していれば、$K' = (b_0, \ldots, b_{15})$, $I' = (b_{16}, \ldots, b_{23})$ が (K, I) と T-slid pair になる。したがって、K, I をランダムに選んだ場合には、任意の T について 2^{-80} 程度の確率で T-slid pair が見つかる。

3.8.2 *Enocoro*-128v1.1 の弱鍵

以下、8-slid pair を用いた関連鍵攻撃で *Enocoro*-128v1.1 の弱鍵を探索する方法について述べる。初期化の状態更新を 8 段繰り返したとき、バッファの下位 8 バイトには IV の値がほぼそのまま格納される。したがって、$I_0, I_1, \ldots, I_7$ として初期定数 $C_0 \oplus C_5, C_1, \ldots, C_7$ を取れば、8-slid pair が見つかる確率は 2^{-16} である。

K の関連鍵 K' を $(K', I', C') = \mathit{Next}^8(K, I, C)$ で定義する。$C = C'$ となる鍵 K が弱鍵であり、このとき (K, I) と (K', I') は 8-slid pair となる。アルゴリズム 1 は弱鍵を検出するための方法である。

Algorithm 1 *Enocoro*-128v1.1 の弱鍵の検出アルゴリズム

ステップ 1：攻撃者は I' の値 X を推定する。

ステップ 2：攻撃者は (K, I) と (K', X) で生成される鍵ストリームを比較する。8-slid pair になっていないならば、ステップ 1 に戻る。

ステップ 3: 8-slid pair になっていれば $X = I'$ と判断する。関係式 $K_8 = I'_0 \oplus C_0, \ldots, K_{15} = I'_7 \oplus C_7$ を用いて、I' から 8 バイトの鍵を復元する。

ステップ 4: $K_0, \ldots, K_7$ を総当りで求める。

アルゴリズム 1 では、ステップ 1 が X の総当りなので 64 ビットの推定、ステップ 4 が $K_0, \ldots, K_7$ の総当りなので同じく 64 ビットの推定を行う。したがって、弱鍵の検出に必要な計算量は 2^{65} 程度である。

3.8.3 *Enocoro*-128v2 の場合

Enocoro-128v1.1 で slid pair が容易に見つかるのは、初期化関数が乱数生成時の状態更新関数の繰り返しで構成されていることが要因である。対策としては、[5] に述べられているようにカウンタ値を用いれば良い。*Enocoro*-128v2 では、初期化の状態更新では 8 ビットのカウンタを用いている。したがって、内部状態のうち、最初に定数がセットされる 80 ビットの値が一致することがあっても、カウンタ値により差分が挿入されることになるので、slid pair とはならないと考えられる。

4 実装評価

Enocoro は、各種実装プラットフォームのうち、特に、専用ハードウェアでの軽量実装性の向上に主眼を置いた設計となっている。本節では、Enocoro のハードウェア、及び、ソフトウェア実装評価の結果についてそれぞれ述べる。

4.1 ハードウェア実装

本節では、*Enocoro* のハードウェア実装評価の結果について述べる。*Enocoro* ハードウェアは、アルゴリズム通りの構成、即ち、ステート a、及びバッファ b を各々、2 バイト、32 バイトのレジスタとして、λ 関数、ρ 関数各々 1 個を組み合わせ論理として実装し、1 クロックで 1 サイクル分の処理を行なう構成が基本形 (以下、この構成を基本形実装と称する) である。又、我々は、基本形実装が最も高効率、好バランスであると考えている。

8 ビット Sbox s_8 は、複数の 4 ビット Sbox s_4 と線形変換 L 等の組合せで構成されて

おり、s_8 のハードウェア化には、8 ビットテーブルを用いる表参照方式と、数学的定義に従い組み合わせ論理を用いて演算を行なう演算方式の、二つの構成法を採ることが可能である。前者は、論理規模は大きくなるが処理遅延は小さく、後者は、処理遅延は大きくなるが、小論理規模での実装が可能であるという特徴を持つ。

基本形実装に対し、省論理化を図る場合には、ρ 関数内の Sbox の共有化が有効である。但し、共有した分、処理に要するクロック数が増加するので、処理性能は共有度合いに応じて減じることになる。例えば、Sbox 数を本来の 4 から、共有により 2、1 と減じた場合、処理スループットは基本形実装に較べて、1/2、1/4 となる。

逆に、基本形実装に対し、処理性能向上を図る場合には、λ 関数、ρ 関数を複数段分直列に具備し、1 クロックで複数サイクル分処理する構成が有効である。例えば、λ 関数、ρ 関数を 4 段分持つことで、1 クロックで 4 サイクル分 32bit の乱数出力を得るような高速実装が可能である。但し、処理遅延が大きくなることで動作周波数が低下するため、性能向上は限定的である。

以下、*Enocoro*-128v2 のハードウェア実装評価数値を示す。実装アーキテクチャは基本形実装とし、Sbox 実装は、表参照方式と演算方式の 2 種類を対象とした。ASIC 実装評価として、90nm スタンダードライブラリを用いた際の論理合成結果を示す。論理規模は、2 入力 NAND ゲート換算である。又、FPGA 実装評価として、ALTERA 社 StratixII シリーズ上での実装評価結果を示す。論理規模は、ALTERA 社 Stratix II の単位ロジックモジュールである Adaptive Logic Module(ALM) 数を、従来の FPGA で一般的な Logic Element(LE) 数に換算した値で示している。尚、両者とも、合成条件は実装方式に応じて、その特徴を活かせるよう適切に設定している。

以上に示したように、*Enocoro* は小さな論理規模でのハードウェア実装が可能という特徴を持つ。一般に、Sbox 等の非線形変換部の実装には大きな論理量が必要であるが、*Enocoro* では、Sbox の必要数が 4 と小さく、更に、演算方式によりその Sbox の回路規模を抑えることも可能であることから、小論理規模での実装が可能となっている。

又、実装論理規模が小さいことから、ハードウェア実装モジュールの消費電力、単位暗号処理ビット当たりの消費電力量が少ないという特徴がある。FPGA 上での消費電力評価として、ALTERA 社 Stratix EP1S80 での消費電力測定結果は以下である [23]。暗号処理中の消費電力は、6.77mW、又、ビット処理電力量は、0.103nW 秒である。上記数値は、AES、MUGI を対象とした同様の測定値と比較して、それぞれ 1/10 以下、1/2 以下である。尚、本結果は、*Enocoro*-128v1.0 での測定値であり、*Enocoro*-128v2 では、初期化サイクルが延長している分、消費電力が若干増加する可能性があるが、大差はないと考えている。

表 2 *Enocoro*-128v2 の ASIC 実装結果

Sbox 実装	合成条件	論理規模 (Kgates)	動作周波数 (MHz)	処理スループット (Gbps)
表参照方式	速度優先	8.7	1250	10.0
演算方式	速度優先	4.7	1052	8.4
	規模優先	4.1	440	3.5

表 3 *Enocoro*-128v2 の FPGA 実装結果

Sbox 実装	合成条件	論理規模 (LE)	動作周波数 (MHz)	処理スループット (Gbps)
表参照方式	速度優先	795	149	1.2
演算方式	速度優先	530	140	1.1
	規模優先	525	118	0.9

4.2 ソフトウェア実装

本節では、*Enocoro* のソフトウェア性能について述べる。ソフトウェアの性能測定では、現時点での標準的な環境として、表 4 に掲げる環境を利用した。

表 4 ソフトウェア性能測定環境

CPU	メモリ	OS	コンパイラ
Intel Core2Duo E6600 (2.4GHz)	2GB	Ubuntu Linux 8.04 32-bit distribution	gcc 4.2.4

我々は *Enocoro*-128v2 のアルゴリズムを C 言語で記述し、gcc のコンパイルオプションは`-O3` を使用した。このプログラムのコードサイズは 974 バイトであり、ワークエリアは 256 バイトであった。

表 5 に複数サイズのデータを処理した際の処理性能 (処理対象 1 バイト当たりのサイクル数) を示す。なお、各々の性能数値の処理サイクル数には、初期化に要する時間 (4870 サイクル) を含んでいる。

表 5 の結果は、上記の処理を 10000 回行い、最後の 100 回の測定結果について平均を

表 5 ソフトウェア性能測定結果

	データサイズ		
	16	256	1M
処理性能 (cycle/Byte)	318.6	64.9	46.3

取ったものである。処理速度の測定には CPU の TSC レジスタを用いた。以下に、処理速度の測定を行う C コードの例を示す。

```
#define RDTSC(X) asm volatile("rdtsc" : "=A" (X))

RDTSC(X);
init(&state, key, iv);
keystream(&state, out, ENOCORO_OUT_SIZE);
RDTSC(Y);
```

4.3 サイドチャネル攻撃に対する耐性

Enocoro のハードウェア実装モジュールは、同 AES モジュールと較べて、消費電力が小さいという特徴がある。DPA や CPA などの電力解析攻撃は、暗号化処理時の電力差分で秘密鍵を推定するものであることから、鍵推定に必要な電力差分を得るための波形計測の難易度が AES と比較した場合に大きい可能性がある。

AES など既存のブロック暗号への電力解析攻撃では、出力暗号文と出力段の 1 つ手前の処理中間値の間のバイト単位のレジスタ遷移を推定することで、容易に鍵を推定できることが知られている。これに対し、*Enocoro* の基本形実装では、乱数出力は毎サイクル 1 バイトのみとなり、AES と同様の手法で並列的に鍵推定をすることは困難である。

共通鍵暗号の電力解析に対する耐タンパ実装としては、Sbox への対策が有効であることが知られている。*Enocoro* の耐タンパ実装についても同様の手法の適用が考えられるが、*Enocoro* に必要な Sbox 数は 4 であり、AES の 16 と比較して小さいことから、対策実装コストが、小さい可能性がある。

5 既存の暗号技術との比較

この節では、*Enocoro* と他の暗号技術、特に CRYPTREC 推奨暗号を比較する。

5.1 安全性

暗号技術の安全性は、個々のアルゴリズムに特化した攻撃や、各々の評価の成熟度など多くの尺度が関係している。このため、複数の異なるアルゴリズムを安全性の観点から比較評価するのは現実的ではない。しかし、*Enocoro* は最初の提案から 2 年以上を経ており、深刻な脆弱性は報告されていない。また、*Enocoro*-128v2 は *Enocoro*-128v1.1 において概評価レベルで指摘された初期化の潜在的な脆弱性についても対策を施している。以上の理由から、*Enocoro*-128v2 は他の 128 ビット鍵暗号と比肩しうる十分な安全性を持っていると考える。

5.2 ソフトウェア実装性能

Enocoro はハードウェア実装における軽量性を目標として設計された疑似乱数生成器であり、ソフトウェア実装性能は CRYPTREC 推奨のブロック暗号やストリーム暗号と比べて明確な優位性は持っていない。

5.3 ハードウェア実装性能

表 6 は、*Enocoro*-128v2 と、CRYPTREC 推奨暗号である AES [11]、MUGI [19]、および Estream [9] portfolio に掲載されているハードウェア向けストリーム暗号のうち鍵長が 128 ビットである Grain-128 [14] と MICKEY 128 [2] のハードウェア実装評価結果を比較したものである。

表 6 から、*Enocoro*-128v2 は CRYPTREC 推奨暗号に比べて実装規模が小さく、また AES に比べて高速な処理を行うことができることがわかる。Estream portfolio に掲載されたアルゴリズム (Grain, MICKEY) との比較では、実装規模、処理速度ともに Grain と MICKEY の中間に位置している。

登録商標

- Altera と Stratix は Altera Corporation の米国およびその他の国における登録商標です。
- *Enocoro* は株式会社日立製作所の登録商標です。
- Intel は Intel Corporations の米国およびその他の国における登録商標です。また、Core は Intel Corporation の製品の名称です。

表 6　ハードウェア実装性能比較

アルゴリズム	動作周波数 (MHz)	回路規模 (KGE)	処理速度 (Mbps)	プロセス (μm)
AES [16]	131.2	5.4	311	0.11
MUGI [29]	51.1	22.7	1,600	0.18
	186.2	46.0	11,900	
Grain128 [13]	925.9	1.9	926	0.13
	581.4	2.5	4,651	
MICKEY128 [13]	413.2	5.0	413	0.13
Enocoro-128v2	440.0	4.1	3,520	0.09

- Linux は Linus Torvalds の米国およびその他の国における登録商標です。
- Ubuntu は Canonical Ltd. の米国およびその他の国における登録商標です。

謝辞

Enocoro は、高度通信・放送研究開発に係る委託研究制度の一環として独立行政法人情報通信研究機構から委託を受け実施している「大容量データの安全な流通・保存技術に関する研究開発」の成果の一部である。

また、*Enocoro* の安全性評価については、東京理科大学の金子敏信教授をはじめ、研究室の多くの方にご協力いただいた。この場を借りて謝意を申し上げる。

参考文献

[1] S. H. Babbage, “Improved exhaustive search attacks on stream ciphers,” *European Convention on Security and Detection*, IEE Conference publication No. 408, pp. 161–166, 1995.

[2] S. Babbage and M. Dodd, “The stream cipher MICKEY-128 2.0,” available at `http://www.ecrypt.eu.org/stream/p3ciphers/mickey/mickey128_p3.pdf`

[3] M. Bellare and T. Kohno, “A Theoretical Treatment of Related-Key Attacks: RKA-PRPs, RKA-PRFs, and Applications,” *Advances in Cryptology–Eurocrypt 2003*, Lecture Notes in Computer Science, Vol. 2656, pp. 491–506,

Springer-Verlag, 2003.

[4] D. Bleichenbacher and S. Patel, "SOBER cryptanalysis," *Fast Software Encryption, FSE'99*, Springer-Verlag, LNCS 1636, pp. 305–316, 1999.

[5] C. De Cannière, Ö K̈uçük, and B. Preneel, "Analysis of Grain ' s Initialization Algorithm," *The State of the Art of Stream Ciphers, SASC 2008*, pp. 43–56, 2008.

[6] N. Courtois and J. Pieprzyk, "Cryptanalysis of Block Ciphers with Overdefined Systems of Equations," `Advances in Cryptology, Asiacrypt'02`, Springer-Verlag, LNCS 2501, pp. 267–287, 2002.

[7] J. Daemen and C. Clapp, "Fast Hashing and Stream Encryption with PANAMA," *Fast Software Encryption, FSE'98*, Springer-Verlag, LNCS1372, pp. 60–74, 1998.

[8] J. Daemen, R. Govaerts, J. Vandewalle, "Resynchronization weaknesses in synchronous stream ciphers," *Advances in Cryptology, Proceedings Eurocrypt'93*, Springer-Verlag, LNCS 765, pp. 159–169, 1994.

[9] eSTREAM, –The ECRYPT Stream Cipher Project–, `http://www.ecrypt.eu.org/stream/`.

[10] eSTREAM PHASE 3 Performance Figures Intel Pentium 4 revision 206, `http://www.ecrypt.eu.org/stream/phase3perf/2007a/pentium-4-a/`.

[11] FIPS 197, "Advanced Encryption Standard," National Institute of Standards and Technology, 2001. Available at `http://www.itl.nist.gov/fipspubs/`.

[12] I. Golić, "Cryptanalysis of alleged A5 stream cipher," *Advances in Cryptology, Eurotcrypt'97*, Springer-Verlag, LNCS 1233, pp. 239–255, 1997.

[13] T. Good and M. Benaissa, "Hardware results for selected stream cipher candidates," available at `http://www.ecrypt.eu.org/stream/papersdir/2007/023.pdf`

[14] M. Hell, T. Johansson and W. Meier, "A Stream Cipher Proposal: Grain-128," available at `http://www.ecrypt.eu.org/stream/p3ciphers/grain/Grain128_p3.pdf`

[15] M. Matsui, "Linear cryptanalysis method for DES cipher," *Advances in Cryptology, Eurocrypt'93*, Springer-Verlag, LNCS 765, pp. 159–169, 1994.

[16] A. Satoh, S. Morioka, K. Takano, and S. Munetoh, "A Compact Rijndael Hardware Architecture with S-Box Optimization," *Advances in Cryptology, ASIACRYPT 2001*, Springer-Verlag, LNCS 2249, pp. 230–254, 2001.

[17] H. Sekine, T. Nosaka, Y. Hatano, M. Takeda, and T. Kaneko, "A strength evaluation of a pseudoramdom number generator MUGI against linear cryptanalysis," *IEICE Trans.* Vol. E88-A, No. 1, pp. 16-24, January 2005.

[18] D. Watanabe, K. Ideguchi, J. Kitahara, K. Muto, H. Furuichi, T. Kaneko, "Enocoro-80: A Hardware Oriented Stream Cipher," *Second International Workshop on Advances in Information Security*, 2008.

[19] 株式会社日立製作所、「疑似乱数生成器 MUGI 仕様書 Ver. 1.3」、`http://www.sdl.hitachi.co.jp/crypto/mugi/` より入手可能.

[20] 株式会社日立製作所、「疑似乱数生成器 *Enocoro* 仕様書 Ver. 2.0」、`http://www.sdl.hitachi.co.jp/crypto/enocoro/` より入手可能.

[21] 古市洋希, 武藤健一郎, 渡辺大, 金子敏信, "擬似乱数生成器 *Enocoro*-80 の再同期攻撃（差分攻撃）に対する耐性評価," 暗号と情報セキュリティシンポジウム, SCIS2008, 4A1-3 January 2008.

[22] 井手口恒太, 渡辺大, "推測決定攻撃に対する安全性評価の一手法," 暗号と情報セキュリティシンポジウム, SCIS2008, 3A1-4 January 2008.

[23] 北原潤, 渡辺大 "暗号アルゴリズム Enocoro のハードウェア実装及び消費電力評価," 暗号と情報セキュリティシンポジウム, SCIS2008, 2C2-3 January 2008.

[24] 鴻巣慧, 武藤健一郎, 古市洋希, 渡辺大, 金子敏信, "Enocoro-128 ver.1.1 の再同期攻撃耐性評価," 信学技報, ISEC2007-147. 2008.

[25] 武藤健一郎, 渡辺大, 金子敏信, " 疑似乱数生成器 Enocoro-80 の Linear Distinguish Attack 耐性評価," 情報理論とその応用シンポジウム, SITA2007 2.4, 2007.

[26] 武藤健一郎, 渡辺大, 金子敏信, "Enocoro-80 の再同期攻撃（線形攻撃）耐性評価," 暗号と情報セキュリティシンポジウム, SCIS2008, 4A1-2 January 2008.

[27] 武藤健一郎, 渡辺大, 金子敏信, "Enocoro-128 の LDA 耐性評価と改良," 暗号と情報セキュリティシンポジウム, SCIS2008, 4A1-1, January 2008.

[28] 岡本和人, 武藤健一郎, 金子敏信, "疑似乱数生成器 Enocoro-80 の差分／線形攻撃耐性評価（II）," 暗号と情報セキュリティシンポジウム, SCIS2009, 4B2-3, 2009.

[29] 大和田 徹, 平 重喜, 五十嵐 悠一, 北原 潤, "MUGI のハードウェア実装及び評価," 暗号と情報セキュリティシンポジウム, SCIS2005, 1A3-5, 2005.

[30] 渡辺大, 金子敏信, "軽量な PANAMA 型疑似乱数生成器の構成に関する検討," 信学技報, ISEC2007-78. 2007.

[31] 渡辺大, 岡本和人, 金子敏信, "軽量ハードウェア向け疑似乱数生成器 *Enocoro*-128v2," 暗号と情報セキュリティシンポジウム, SCIS2010, 3D1-3, 2010.