自己評価書

（２００１年度改訂）

MULTI-S01

（株）日立製作所

目次

1. 自己評価書概要

本自己評価書（以下評価書）は、MULTI-S01（マルチエスゼロワン）暗号の評価についての研究成果についてまとめたものである。

MULTI-S01 は、内部に擬似乱数生成器 PANAMA を使用しており、この安全性に基づいて暗号全体の安全性を保証している。この評価書では、三つの部分に分割して議論し、その中で以下に示す話題について議論する。

1. 擬似乱数生成器の安全性を仮定した上での安全性評価
   - ✓ メッセージ秘匿
   - ✓ メッセージ認証
2. 擬似乱数生成器 PANAMA の安全性評価
   - ✓ 乱数性テスト
   - ✓ 暗号学的安全性の検討
3. 処理速度などの実装評価
   - ✓ ソフトウェア実装
   - ✓ ハードウェア実装

これらの評価結果に基づき MULTI-S01 暗号が安全かつ、効率的な共通鍵暗号処理であることの根拠とする。

なお、本暗号の最新情報については、下記のＵＲＬにて随時情報を公開する。

http://www.sdl.hitachi.co.jp/crypto/s01/index-j.html

2. メッセージ秘匿とメッセージ認証に関する安全性

2.1. はじめに

この章では、擬似乱数生成器 PANAMA の安全性を仮定した場合の、MULTI-S01 の安全性について考察する。MULTI-S01 は共通鍵暗号として、以下のふたつの安全性を供給する。

メッセージ秘匿

秘密鍵を知らない攻撃者は暗号文単独攻撃からは、平文の情報を知ることができない、という性質。

メッセージ認証

秘密鍵を知らない攻撃者は平文暗号文組からは、有意な改竄を行うことができないという性質。有意な改竄とは暗号処理で定められた（受信者による）改竄検査をパスするような（オリジナルとは異なる）暗号文の生成である。

従来から知られる技術によるアプローチ

ここでは簡単に従来技術による、メッセージ秘匿とメッセージ認証の解決手段について簡単にまとめる。暗号技術を使ったメッセージ秘匿、メッセージ認証の手法は大別して、共通鍵暗号技術によるもの、公開鍵暗号技術によるものに二分することができる。

表 2.1 秘匿と認証の暗号技術

	メッセージ秘匿	メッセージ認証
共通鍵暗号技術	ブロック暗号 ストリーム暗号	メッセージ認証子(MAC)
公開鍵暗号技術	公開鍵暗号	デジタル署名

本稿では、共通鍵暗号技術のみを扱い、以下では公開鍵暗号技術については触れない。

メッセージ秘匿を目的とする現実的な暗号技術のうち、現在一般に知られるものは、ブロック暗号とストリーム暗号に分割することができる。

ブロック暗号の場合、通常よく用いられる ECB, CBC, CFB, OFB, PCBC, カウンタモードなどはメッセージの秘匿の目的で用いられる。ストリーム暗号についても、同期式、自己同期式の両方ともにメッセージ秘匿の目的で用いられる。

これに対し、データの完全性はメッセージ認証の技術を用いて達成される。具体的にはメッセージ認証符号(Message Authentication Code:MAC)の生成による方法で実現される。MAC の生成には、DES-MAC[1]のようなブロック暗号を使ったものや、MMH[12]のような擬似乱数生成器によるもの、HMAC などのハッシュ関数を使ったものがある。

例えば秘匿性と完全性の両方を必要とする暗号処理の実装を考えた場合、上で挙げた両方の技術を実装することで達成することができるが、効率の面で次のような問題点がある。

まず、鍵の長さが余計に必要となる。一般に暗号処理のメカニズムとメッセージ認証のメカニズムとで同じ秘密鍵を用いることによる安全性は保証されておらず、場合によっては(例えば CBC モードで暗号処理、CBC-MAC によるメッセージ認証の組合せ)致命的な欠陥が発生する。

また、暗号処理、メッセージ認証それぞれのためにメッセージをスキャンする必要がある。多くのソフトウェア実装の場合では暗号処理、メッセージ認証それぞれで関数呼出を行い、暗号処理、メッセージ認証それぞれ独立に行う。よってメッセージが長い場合などは記憶領域へ記憶しておく必要がある場合がある。

また、必要な追加コストも無視することができない。例えば現在知られる最も高速な MAC である UMAC[6]では、ハッシュ関数と擬似乱数生成器の両方を実装する必要がある。

秘匿性と完全性を同時に保証する方式としていくつかが考えられる。BEAR と LION[2]は任意のブロック長のブロック暗号を擬似乱数生成器とハッシュ関数より生成する手法である。しかし、これらを同時にメッセージ認証に用いた場合、メッセージ全体と同じ容量の作業用メモリを必要とし、特に長いメッセージに対して効率的でない。

iaPCBC はメッセージ認証を可能とするブロック暗号操作モード[11]であるが、CBC に代表されるブロック暗号を用いたデータを連鎖するモードの一つであり、事前計算、並列処理の適用が一般に困難である。

並列処理性の優れたメッセージ認証を行う操作モードに unforgeable encryption[14]がある。この場合、並列処理性がある一方、暗号文の長さがメッセージに対して定比例的に増加してしまう。

本稿では、ストリーム暗号の高速処理、事前計算、並列計算の機能を持ったメッセージ認証可能な共通鍵暗号方式を提案し、その安全性、効率について議論する。

提案する新しい暗号方式の最も大きな特徴は、データの秘匿と完全性を同時に保証することである。この特徴を保ちつつ、効率的な暗号アルゴリズムを提案するためのアプローチを以下のように整理する：

完全性の保証について

ブロック暗号の CBC モードのように、理想的なブロックごとの攪拌を行い、次のブロックへの適当なフィードバックを構成する。これにより、改ざんを行ったとき、最終ブロックへの理想的な差分の伝播を構成することを考える。

処理の効率について

ストリーム暗号の構造のように、暗号処理のうち、もっとも計算量のかかるブロック暗号や擬似乱数生成器の処理を平文・暗号文に作用させないようにする。これにより、事前計算や並列計算を効率よく行うことを考える。

これらを達成する一手法として、本方式を提案する。以下に解決手段となる方式の紹介を行う。

2.2.暗号化モデル

この章で扱う暗号処理と、復号化のときのメッセージ認証について示す。暗号化処理のモデルは以下のとおり：

暗号化

入力：メッセージ M (n×64 ビット)
冗長符号 R (64 ビット)
秘密鍵 A ($A \neq 0$, 64 ビット), B_i ((n+2) × 64 ビット), S (64 ビット)
（ただし、乱数発生の順序から $S = B_{n+3}$）
出力：暗号文 C ((n +2)×64 ビット)

- **処理１：（メッセージのパディング）**
  暗号処理のためのデータ P を生成する（ここで $A||B$ はバイナリ文字列（A,B ）の連結である。
  $P = M||S||R$
  また、P を 64 ビットごとに区切ったデータを P_i（$1 \leq i \leq n+2$）とする。

- **処理２：（暗号化演算）**
  暗号文ブロック C_i を以下の式で生成する（ただし $F_0=0$ とし、⊕⊗は有限体 $\mathbf{F}2^{64}$ 上の加算、乗算を表すものとする）。
  $F_i = P_i \oplus B_i$ (1)
  $C_i = A \otimes F_i \oplus F_{i-1}$ (2)

復号化

入力：暗号文 C (64×n' ビット)
　　冗長符号 R (64 ビット)
　　秘密鍵 A (≠0,64 ビット), B (64×n' ビット), S (64 ビット)
出力：改竄検出信号、またはメッセージ M (64×(n'−2)ビット)

- **処理１：（復号化演算）**
  暗号文ブロック C_i を以下の式で生成する（暗号化の場合と同様に F_0=0 とする）。

$$F_i = A^{-1} \otimes (C_i \oplus F_{i-1}) \quad (3)$$

$$P_i = F_i \oplus B_i \quad (4)$$

- **処理２：（冗長性の切り出しと検査）**
  P' から埋め込み乱数 S'と冗長符号 Rを切り出す。処理１で得られた P の最終ブロック $P_{n'}$ を R'とし、最終第 2 ブロック $P_{n'-1}$ を S' とする。$R'=R$ かつ $S'=S$ ならば残りの P（すなわち $P_1,...,P_{n'-2}$）を出力し、そうでないなら改竄検出信号を出力する。

この暗号処理について、以下で示す二種類の安全性、すなわちメッセージ秘匿とメッセージ認証、について、以下の定理が証明される。

定理1　(MULTI-S01 の安全性)

擬似乱数生成器 PANAMA の安全性を仮定すると、MUTLI-S01 は完全秘守性を有し、改竄の成功確率はメッセージのブロック数（メッセージを 64 ビットのブロックへ区切った場合の）n に対し、高々$(n+1)/2^{64}$ である。

この証明については以下、完全秘守性、改竄成功確率の二面を独立に議論してゆく。

2.3. MULTI-S01 の完全秘守性

ある平文 M に決められた乱数パディング S、冗長性 R を付加したものを P とする。完全秘匿性のための必要十分条件は、暗号文を C として、$\Pr(P \mid C)=\Pr(P)$ が成り立つことである。ここで $\Pr(Y \mid X)$ は事象 X が起こったときの、事象 Y の起こる条件付き確率を表す。これを示すために補題を与える。

補題1　（ある既知平文組に対する等価鍵の個数）

P を平文列、C を暗号文列とする。固定した P,C を対応づける鍵ストリーム (A,B) の個数は $2^{64}-1$ 個である(ここで B は B_i の列を表す)。

証明：

式(1)(2)より、Bについての漸化式は以下のようになる。

$$B_1 = (C_1 \oplus F_0) \otimes A^{-1} \oplus P_1$$
$$B_i = (C_i \oplus P_{i-1} \oplus B_{i-1}) \otimes A^{-1} \oplus P_i$$

任意のAに対して、B列は一意に決まるので、PをCへ暗号化する(A, B)はAの数ぶん、すなわち$2^{64}-1$個以上は存在する。

次に、PをCに暗号化する鍵候補のうち、Aの値に対してB列は一意に決まることを示し、全体で$2^{64}-1$だけ存在することを示す。

二つの鍵ストリーム(A, B'), (A, B'')を考える。ここでAが共通であることに注意する。これらによる平文Pの暗号処理は以下のとおり：

$$F'_{i+1} = P_i \oplus B'_i$$
$$C'_i = (F'_{i+1} \otimes A) \oplus F'_i$$
$$F''_{i+1} = P_i \oplus B''_i$$
$$C''_i = (F''_{i+1} \otimes A) \oplus F''_i$$

jを、$j = \min_i(i : \text{such that } B'_i \neq B''_i)$とする。式(1)(2)より$F'_j = F''_j$、$B' \neq B''$かつ、$F'_{j+1} \neq F''_{j+1}$。よって$C'_j \neq C''_j$。ゆえに、同じ$A$をもちかつ異なる$B$列を持つ鍵ストリームは、いかなる$P$も同じ$C$に暗号化することはない。

以上により(P, C)の組を対応づける(A, B)の数は$2^{64}-1$個である。

次に秘匿性を証明する。

定理2　（MULTI-S01 の秘匿性）

式(1)(2)で表される暗号処理は完全秘匿性を保持する。

証明：

$\Pr(P)$を（パディングしたあとの）平文入力としてPが与えられる確率とする。この平文がCへ暗号化される確率を評価する。lをPの長さ（ビット）とする。この場合暗号文Cの長さもlビットとなる。鍵の場合の数はAが非ゼロかつ64ビット、Bがlビットなので$(2^{64}-1)2^l$である。このうち暗号文Cを対応づけるのは補題1より$2^{64}-1$個であるので、任意の平文Pについて、暗号文がCとなる（条件付き）確率は

$$\Pr(C \mid P) = (2^{64}-1) / \{(2^{64}-1)2^l\} = 1/2^l$$

となる。よって、既知平文(P, C)が現れる確率は

$$\Pr(P, C) = \Pr(P) / 2^l$$

であり、これから暗号文の出現確率$\Pr(C)$を導く。$\Pr(C)$は、すべての平文について$\Pr(P, C)$の和により表されるので

$$\Pr(C) = \Sigma_P \Pr(P,C) = \Sigma_P \Pr(P) / 2^l = 1/2^l$$

となる。よって $\Pr(P|C)=\Pr(P,C) / \Pr(C) = \Pr(P)$となり、完全秘匿性の条件が示される。

注意：（平文の部分情報が攻撃者に事前に知られる場合の秘匿性）

メッセージ自体に潜在する冗長性や、メッセージに付加する固定の冗長符号は事前に攻撃者に知られていると考えられる。しかし、これらはその他の情報の漏洩とは関係なく、攻撃者は（事前に知ることのできる情報以上の）平文情報を得る事ができないことも保証される。これは、上記証明で、平文 P の存在確率として、メッセージの冗長性やパディングなどを考慮した確率分布を考えることで、証明の仮定の中に含めることができる。よって、上記のような、平文の部分情報が攻撃者へ漏れるにも、その他の情報の秘匿性を保証することができる。

2.4. MULTI-S01 の改竄成功確率

まず、攻撃の前提を示す。

攻撃の前提条件

攻撃者には既知平文（メッセージ M と冗長性 R、それに暗号文 C）が与えられると仮定し、攻撃として、「改竄を行うことにより、別の暗号文であって、復号化したときの最後の二ブロックの値（すなわち、最終第 2 ブロック（乱数パディング S）と最終ブロック（冗長データ R））両方がもとの値と同じものを生成すること」を考える。

この前提条件のもとで改竄の成功確率を考察する。改竄の成功確率は(1)暗号文の長さが変化しない改竄、(2)暗号文が短くなる改竄、(3)暗号文が長くなる改竄、の 3 通りに分けて考察する。

2.4.1. 暗号文の長さが変化しない改竄の場合

まず、補題 1より、攻撃者は平文暗号文の情報から A の値について全く情報を得ることができない ($2^{64}-1$ 通り考えうる)ことに注意する。改竄の成功確率の評価は、改竄成功の必要条件となる A についての（改竄により特定される）A についての方程式を考える。

まず、改竄した暗号文とその復号化結果である(C'_i, P'_i)について以下が成り立つ。

$$F'_0 = F_0\ (=0)$$
$$F'_i = (C'_i \oplus F_{i-1}) \otimes A^{-1} \quad (5)$$
$$P'_i = F'_i \oplus B_i \quad (6)$$

さらに、もともとの平文と暗号文(P_i, C_i)について以下が成り立つ。

$$F_i = (C_i \oplus F_{i-1}) \otimes A^{-1}$$
$$P_i = F_i \oplus B_i$$

ただし、

$$1 \le i \le n'(=n+2)$$

よって、これらから、以下の式が得られる。

$$P_{n'-1} = B_{n'-1} \oplus (\oplus_{i=0\ldots n'-2}\ C_{n'-i-1}\ A^{-(i+1)})$$
$$P'_{n'-1} = B_{n'-1} \oplus (\oplus_{i=0\ldots n'-2} C'_{n'-i-1}\ A^{-(i+1)})$$

ここで、$\oplus_i\ X_i$ は項 X_i に対する各 i についての排他的論理和を表し、$\delta_i = C_i \oplus C'_i$ と定義する。このとき、任意の（暗号文への）改竄はδ_i で一意に表現できることに注意する。攻撃者の目的は、改竄したあとの乱数パディングと冗長性にあたる（復号文の）データの**両方**が変更されないような改竄である。よって関係式 $P_{n'-1} = P'_{n'-1}$ は、改竄が成功するための必要条件である[1]。このような暗号文へ改竄することは、以下の式を満たすδ_iを求めることと同値である。$P_{n'-1} = P'_{n'-1}$ より、改竄のパターンであるδ_iについて以下の式が導かれる。

$$0 = \oplus_{i=0\ldots n'-2}\ \delta_{n'-i-1}\ A^{-(i+1)}$$

補題 1より、攻撃者は A についてのいかなる情報も知らない。A を知らない状態で上記方程式を成り立たせる係数δを決定する。

上記の式を A についての方程式と考えると、$A \neq 0$ であることから $n'-2$ 次以下の方程式となる。よって、攻撃者にとってランダムな A が方程式を満たすような確率をなるべく大きくするような方法は、相異なる（方程式で取りうる最大の解の個数である）$n'-2$ 個の解を持つ方程式となるようにδ を選ぶことである。この場合、改竄が成功する確率は、乱数 A が $n'-2$ 個の解のひとつである確率なので $(n'-2)/(2^{64}-1) = n/(2^{64}-1)$となる。

2.4.2. 暗号文が短くなる改竄の場合

改竄後の暗号文の長さを n'' ブロック（$n'' < n'$ かつ $n'' > 2$）とする。改竄前の平文を P、改竄後を P' とする。この場合の改竄が成功するための必要十分条件は以下のとおり。

[1] この条件にさらに $C_{n'} = C'_{n'}$ を条件を加えることで、改竄が成功する($P_{n'} = P'_{n'}$ かつ $P_{n'-1} = P'_{n'-1}$)ための必要十分条件となるが、ここでは単に $P_{n'-1} = P'_{n'-1}$ のための条件を考える。

$$P'_{n''} = P_{n'} (= R)\ \text{かつ} \quad P'_{n''-1} = B_{n''+1}\ (\text{秘密パディング } S)$$

必要条件として $P'_{n''-1} = B_{n''+1}$ を考える。復号化の式(3)(4)より

$$P'_{n''-1} = B_{n''-1} \oplus (\bigoplus_{i=0\ldots n''-2} C'_{n''-i-1} A^{-(i+1)})$$
$$B_{n''+1} = P_{n''+1} \oplus (\bigoplus_{i=0\ldots n''} C_{n''-i+1} A^{-(i+1)})$$
$$B_{n''-1} = P_{n''-1} \oplus (\bigoplus_{i=0\ldots n''-2} C_{n''-i-1} A^{-(i+1)})$$

これらより

$$\begin{aligned} P'_{n''-1} \oplus B_{n''+1} &= B_{n''-1} \oplus P_{n''+1} \\ &\quad \oplus (\bigoplus_{i=0\ldots n''-2} C'_{n''-i-1} A^{-(i+1)}) \\ &\quad \oplus (\bigoplus_{i=0\ldots n''} C_{n''-i+1} A^{-(i+1)}) \\ &= P_{n''+1} \oplus P_{n''-1} \\ &\quad \oplus (\bigoplus_{i=0\ldots n''-2} C'_{n''-i-1} A^{-(i+1)}) \\ &\quad \oplus (\bigoplus_{i=0\ldots n''} C_{n''-i+1} A^{-(i+1)}) \\ &\quad \oplus (\bigoplus_{i=0\ldots n''-2} C_{n''-i-1} A^{-(i+1)}) \end{aligned}$$

よって「次の A についての方程式を満たす C' を決定すること」は、「改竄が成功すること」の必要条件なる。ここで、$\delta_i = C_i \oplus C'_i$ とする。

$$0 = P_{n''+1} \oplus P_{n''-1} \oplus (\bigoplus_{i=0\ldots n''-2} \delta_{n''-i-1} A^{-(i+1)}) \oplus (\bigoplus_{i=0\ldots n''} C_{n''-i+1} A^{-(i+1)})$$

これが（攻撃者にはわからない未知数 A について）成り立つようなδ を決めたとき、改竄が成功する。この式は（A は 0 でないので、$A^{n''+1}$ を両辺に乗じることで）A についての $n''+1$ 次以下の方程式と等価となる。この方程式の係数は C, δ で決定されるが、この方程式の解の個数が最大 $n''+1$ であることは自明である。よって、攻撃者が改竄を成功させるための最も効率的な手段はこの方程式が最大の解の個数を与えるようなδを決定するときであるので、攻撃の成功確率は最大でも、$(n''+1)/(2^{64}-1) \le n'/(2^{64}-1) = (n+2)/(2^{64}-1)$となる。

2.4.3. 暗号文が長くなる改竄の場合

改竄後の暗号文の長さを n'' ブロック（$n' < n''$）とする。改竄前の平文を P、改竄後を P' とする。この場合の改竄が成功するための必要十分条件は以下のとおり。

$$P'_{n''} = P_{n'} (= R)\ \text{かつ} \quad P'_{n''-1} = B_{n''+1}\ (\text{秘密パディング } S)$$

必要条件として $P'_{n''-1} = B_{n''+1}$ を考える。復号化の式(3)(4)より

$$P'_{n''-1} = B_{n''-1} \oplus (\bigoplus_{i=0\ldots n''-2} C'_{n''-i-1} A^{-(i+1)})$$
$$B_{n''+1} = P_{n''+1} \oplus (\bigoplus_{i=0\ldots n''} C_{n''-i+1} A^{-(i+1)})$$

これらより

$$P'_{n''-1} \oplus B_{n''+1} = B_{n''-1} \oplus P_{n''+1} \oplus (\bigoplus_{i=0\ldots n''-2} C'_{n''-i-1} \mathrm{A}^{-(i+1)}) \oplus (\bigoplus_{i=0\ldots n''} C_{n''-i+1} \mathrm{A}^{-(i+1)})$$

よって「次の A についての方程式を満たす C' を決定すること」と改竄が成功することが等価となる。

$$0 = B_{n''-1} \oplus P_{n''+1} \oplus (\bigoplus_{i=0\ldots n''-2} C'_{n''-i-1} \mathrm{A}^{-(i+1)}) \oplus (\bigoplus_{i=0\ldots n''} C_{n''-i+1} \mathrm{A}^{-(i+1)})$$

ここで、$B_{n''-1}$ は暗号化処理では生成されていない乱数であり、攻撃者は $B_{n''-1}$ について値をいかなる情報もわからない。よって、攻撃者が改竄を成功させるための最も効率的な手段は、A, $B_{n''-1}$ をランダムに推定し、それを満たすような C' を決定することなので、攻撃の成功確率は最大でも、$1/2^{64}$ となる。

3. 擬似乱数生成器 PANAMA の安全性

PANAMA は、J. Daemen と C. Clapp が 1998 年に FSE'98 で提案した[8]暗号モジュールであり、モードの選択により、ストリーム暗号、ハッシュ関数として用いることができる。PANAMA は、著名な暗号研究者が設計した暗号モジュールとして注目を集めた。その後、現在にいたるまで、疑似乱数生成モードに関する目立った解析の報告はないが、ハッシュモードに関してはハッシュ値が衝突するメッセージの生成方法が発表された[31]。PANAMA の安全性に関するもっとも詳細な記述は、発表に先立つ設計者の一連の研究に見られる。これらの研究結果は、[7]にまとめられている。

擬似乱数生成器の出力は、真の乱数と区別することができてはならない。しかし、これはひじょうに曖昧な要求である。そこで、擬似乱数生成器の安全性を議論する上で、次のような指標が存在する。

(1) 長周期性
(2) 乱数性(いくつかの乱数性テストに合格すること)
(3) その他、理論的な欠陥がないこと

暗号学的に安全な擬似乱数は、鍵長に相応しい長い周期を持つことが求められる。PANAMA の出力する乱数列の周期を評価することは容易ではないが、特に非線形変換 γ の複雑な挙動から、擬似乱数として十分長い周期を持つものと考えられる。

(2)の条件は、ここでは 01 の出現頻度など、いくつかの統計的性質に関する χ^2 検定に合格することとする。これは、擬似乱数が満たすべきと思われる性質のうち、もっとも基本的なものである。(2)に関する評価結果は3.1で述べる。

(3)は(1)(2)に含まれない雑多な統計的性質とする。本稿では、(3)に属する統計的性質として、差分伝播性と相関性、特に線形相関性を取り上げる。(3)に関する結果は3.2で述べる。

3.1.乱数性テスト

本節では、PANAMA の出力に対して、いくつかの乱数性をチェックするテストを行った結果について述べる。

暗号学的安全性が要求される擬似乱数列の乱数性をチェックする方法は、例えば FIPS140-1([9])などに記載されている。しかし、[9]に記載されているテストでは、テストする擬似乱数列の長さが短いため、ストリーム暗号に用いる擬似乱数列の安全性を保証することはできない。我々は、FIPS 140-1 に準拠した乱数性テストの他に、十分に長い平文長に対して頻度検定を行った。

3.1.1. FIPS 140-1 に準拠した乱数性テスト

[9]では、20000 ビットの乱数列に対する乱数性テストとして、以下の 3 つが記載されている。

(1) 1 ビットの頻度検定
(2) 4 ビットの頻度検定
(3) 連の検定(長すぎる連の検出を含む)

しかし、ストリーム暗号に用いる擬似乱数生成器の出力を検査する場合、20000 ビットはあまりに短い(短めのテキスト文書相当)。ここでは、乱数列の長さを 2^{21} ビット(256K バイト)の出力に対して検定を行った。PANAMA の初期値(鍵、乱数列番号)は、C 言語標準ライブラリの乱数生成関数を用いて 2^{12} 組生成した。頻度検定の方法は[15]に、連の検定は[17]にもとづく。

テストの結果を表 3.1にまとめた。表の値は、それぞれのテスト内容を、ある棄却率で検定した場合に、「乱数ではない」と判定された初期値の組の数である。例えば、1 ビットの頻度検定では、4096 個中 39 個の初期値が出力する乱数列が 99%の確率で真の乱数ではない、と言える。カッコの中は、検定した初期値の組の数に対する棄却された初期値の組の数の比率である。

表 3.1 乱数性テスト結果(FIPS 140-1)

棄却率 / テスト内容	0.05	0.01	0.001
1 ビットの頻度検定(/4096)	152(0.037)	39(0.0095)	3(0.0007)
4 ビットの頻度検定(/4096)	172(0.042)	33(0.0081)	3(0.0007)
連の検定(/4096)	196(0.048)	43(0.0105)	4(0.001)

表 3.1から、いずれのテストについても、初期値の組が、真の乱数と区別できる乱数列を出力する割合は、棄却率にほぼ等しい。また、同様の条件で長い連の検出も行った結果、最も長い連は 32 であった。乱数列の長さが 2^{21} ビットのとき、長さ 32 の連が存在する頻度の期待値は 2^{-12} である。

3.1.2. 長い乱数列に対する頻度検定

2.4の議論から、MULTI-S01 は、平文長が 2^{32}(64 ビット)以上ならば、擬似乱数 A を取り換えて使用する必要がある。そこで、乱数列の長さは、最大で 2^{38} ビット($2^{32}\times 64$ ビット)として頻度検定を行い、MULTI-S01 で使用する乱数列として必要とされる乱数性を確認した。PANAMA は、32 ビット幅の処理を行う擬似乱数生成器なので、本来ならば、頻度検定は 32 ビットに対して行うべきである。しかし、2^{32} のメモリ空間を用意するのは事実上不可能であるため、今回は、1 ビット、2 ビット、8 ビットに対して検定を行った。検定の方法は[15]にもとづく。

さらに、平文長の違いによる乱数性の変化を観察するために、平文長が 2^{22},2^{26},2^{30},2^{34} ビットの場合にも同様のテストを行った。鍵、乱数列番号は、C 言語標準の乱数生成関数を用いて 512 組生成し、これらを固定して使用した。

次に、テストの結果について述べる。それぞれのテスト結果を表 3.2、表 3.3、表 3.4にまとめた。表の値は、それぞれのテストを、ある乱数列の長さに対して行い、ある棄却率で検定した場合に、「乱数ではない」と判定された初期値の組の数である。例えば、2^{33} ビットの乱数列に対して、1 ビットの頻度検定を行った結果を見ると、512 個中 5 個の初期値が出力する乱数列が 99%の確率で真の乱数ではない、と言えることがわかる。

表 3.2 乱数列テストの結果(1 ビットの頻度検定)

棄却率 / 乱数列の長さ(ビット)	0.05	0.01
2^{17}	17(/512)	3(/512)
2^{21}	24(/512)	8(/512)
2^{25}	27(/512)	4(/512)
2^{29}	21(/512)	5(/512)
2^{33}	26(/512)	5(/512)

表 3.3 乱数列テストの結果(2 ビットの頻度検定)

棄却率 / 乱数列の長さ(ビット)	0.05	0.01
2^{17}	20(/512)	4(/512)
2^{21}	22(/512)	3(/512)
2^{25}	29(/512)	4(/512)
2^{29}	21(/512)	6(/512)
2^{33}	22(/512)	4(/512)

表 3.4 乱数列テストの結果(8 ビットの頻度検定)

棄却率 / 乱数列の長さ(ビット)	0.05	0.01
2^{17}	29(/512)	10(/512)
2^{21}	21(/512)	3(/512)
2^{25}	17(/512)	7(/512)
2^{29}	18(/512)	5(/512)
2^{33}	26(/512)	4(/512)

いずれのテストについても、初期値の組が、真の乱数と区別できる乱数列を出力する割合は、棄却率にほぼ等しい。また、すべての乱数列の長さに対して乱数と区別できるような初期値の組は存在しなかった。

以上のテストの結果、PANAMA の出力は真の乱数と区別できるとは言えない。

3.2.暗号学的な安全性

擬似乱数生成器は、ブロック暗号に比べて構造が豊富であり、一般に適用できる攻撃法は出力される乱数列の情報を用いたものに限定される(擬似乱数生成器の構造によっては、既知の内部情報(カウンタなど)を利用することができる場合もある)。擬似乱数生成器に対する攻撃とは、次の 2 つを指す。

(i) 入手した乱数列から、固定の秘密情報(シード)を求める。

(ii) 入手した乱数列から、その後生成される乱数列を推定する。

擬似乱数生成器の安全性に関する指標のひとつに線形複雑度がある。線形複雑度は、与えられた(有限な)乱数列を生成する最小の線形フィードバックシフトレジスタのレジスタ長で定義される。与えられた乱数列の線形複雑度を求める Massey-Berlekamp のアルゴリズムはよく知られているが、十分に長い周期を持つことが予想される擬似乱数生成器について、線形複雑度を計算するのは現実的ではない。例えば、鍵長が 256 ビットの擬似乱数生成器が安全であるためには、線形複雑度が 2^{255} 以上である必要がある。この理由から、本稿では、PANAMA の線形複雑度について論じることはしない。

PANAMA は、初期化の段階の攪拌で、シードを推定することが非常に困難となるため、攻撃対象として(i)を想定するのは得策ではない。本稿では、PANAMA に対する攻撃を(ii)に絞り、想定する攻撃手法として、差分伝播性と線形相関性を考える。非線形関数 ρ に対するこれらの性質に関する考察は、[7]で十分になされており、ここでは、[7]の結果を紹介するにとどめる。

3.2.1. 差分伝播性と相関性の定義

差分伝播性

$\mathbf{F}2^n$ の元 a, a^*に対して、その差分 a' を 2 つの元の排他的論理和 $a+a^*$で定義する。$\mathbf{F}2^n$ 上の関数 f に対して、$b'=f(a)+f(a^*)$とするとき、関数 f は差分 a'を差分 b'に伝播する、と言う。

擬似乱数生成器の出力する乱数列から求める差分を持つブロックを抽出するのは困難であるため、差分伝播性は、そのまま解読の容易性には結びつかない[5]。

相関性

n 変数ブール多項式 f, g に対して、その相関係数を

$$C(f,g) := 2 \cdot \Pr(f(a) = g(a)) - 1$$

で定義する。$C(f \circ \phi, g) \neq 0$ のとき、f と g は相関性を持つ、と言う。特に、非線形関数 φ に対して、2 つの 1 次式 f, g をとり、相関係数 $C(f \circ \phi, g)$ を考えることは、通常の意味での線形近似を考えることに相当する。

3.2.2. 非線形変換ρに関する結果

γの非線形性

γについては、次の 2 つの結果がある。
(1) γの線形確率は、出力マスクのハミング重みに比例して、指数的に減少する。
(2) γの差分確率は、入力差分のハミング重みに比例して、指数的に減少する。

θの伝播性

θは、ビット位置の分散を行う線形変換であり、ワード単位の分散を行う前半部と、巡回シフトを用いてワード内の位置の変更を行う後半部とに分けられる。θの前半部は、17 ワードの変換として可逆であり、さらに、前期の条件を満たすものの中で、攪拌性の高いものを選んでいる。表 3.5は、θの前半部をビット単位の関数と見た場合に、入力のハミング重みと出力ビットのハミング重みの相関を表したものである。表 3.5において、縦軸は入力(17 ビット)のハミング重み、横軸は出力(17 ビット)のハミング重み、表内の数値は頻度を表す。例えば、入力のハミング重みが 4 である場合(3360 通り)に、出力のハミング重みが 4 となるのは 119 通りである。

表 3.5 θのハミング重み分布

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
0	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
1	–	–	–	17	–	–	–	–	–	–	–	–	–	–	–	–	–	–
2	–	–	–	–	51	–	85	–	–	–	–	–	–	–	–	–	–	–
3	–	–	–	17	–	153	–	374	–	136	–	–	–	–	–	–	–	–
4	–	–	–	–	119	–	561	–	1071	–	578	–	51	–	–	–	–	–
5	–	–	–	34	–	561	–	1802	–	2465	–	1173	–	153	–	–	–	–
6	–	–	–	–	374	–	1870	–	4624	–	3910	–	1445	–	153	–	–	–
7	–	–	–	170	–	1394	–	5440	–	7208	–	4233	–	935	–	68	–	–
8	–	–	68	–	748	–	4454	–	8806	–	7344	–	2584	–	289	–	17	–
9	–	17	–	289	–	2584	–	7344	–	8806	–	4454	–	748	–	68	–	–
10	–	–	68	–	935	–	4233	–	7208	–	5440	–	1394	–	170	–	–	–
11	–	–	–	153	–	1445	–	3910	–	4624	–	1870	–	374	–	–	–	–
12	–	–	–	–	153	–	1173	–	2465	–	1802	–	561	–	34	–	–	–
13	–	–	–	–	–	51	–	578	–	1071	–	561	–	119	–	–	–	–
14	–	–	–	–	–	–	–	–	136	–	374	–	153	–	17	–	–	–
15	–	–	–	–	–	–	–	–	–	–	–	85	–	51	–	–	–	–
16	–	–	–	–	–	–	–	–	–	–	–	–	–	–	17	–	–	–
17	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	1

3.2.3. 線形攻撃に対する耐性

以下では、ρを状態 a のみからなる変換ρ'と、バッファの挿入部に分割して考える。

$$\rho' = \theta \circ \pi \circ \gamma,$$
$$\rho = \rho' + K$$

また、t ラウンド目のρの出力を $a^{(t)}$、そのうち第 1〜8 ワードを $a^{(t)}_{\mathrm{H}}$、第 9〜16 ワードを $a^{(t)}_{\mathrm{L}}$ と表す。

各ワードの 1 ビットのみを用いた線形近似を総当たりで探索した結果、ρ'は、$a^{(t)}_{\mathrm{L}}$ のみからなる偏差 2^{-3} の線形近似を持つことがわかった。この線形近似を与える入力マスク、出力マスクをそれぞれ Γ_1、Γ_2 とすると、

$$\Gamma_1 \bullet a_L^{(t)} = \Gamma_2 \bullet \rho'(a^{(t)})_L$$

前節の結果と合わせれば、これがρ'の最良近似を与える事がわかる。各ラウンドのρに対して、バッファの 1 ビットを含む線形近似式

$$\Gamma_1 \bullet a_L^{(t)} + \Gamma_2 \bullet a_L^{(t+1)} = \Gamma_2 \bullet K^{(t)} \quad (\mathrm{prob}=2^{-3})$$

が成り立つ。この近似式を利用して、$K^{(t)}$を線形フィードバックシフトレジスタで表現するには、バッファの大きさから、$K^{(t)}$が 63 個必要である。この表現の成立する確率は、

$$2^{63-1}(2^{-3})^{63} = 2^{-127}$$

したがって、解読に必要な乱数列は

$$2^{2^{\times}127} \times 256 = 2^{262}$$

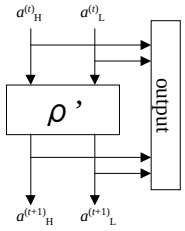
となる。

3.2.4. 簡略化した PANAMA に対する攻撃

PANAMA では、非線形変換ρに挿入されるバッファの値がすべて異なる。このため、バッファから挿入される値を統計的手法により特定することは難しい。したがって、ブロック暗号への攻撃法をそのまま適用することはできない。しかし、すべての鍵が独立と考えるのは、擬似乱数生成器の性質上無意味である。そこで、この節では、PANAMA の構造を簡略化したもの(以下、TOY と総称)を 3 つ考察し、これらを攻撃することを試みる。TOY1〜3 に対しては、いずれも容易に攻撃が成立する。

TOY1：非線形部ρの出力をすべて擬似乱数として出力する

PANAMA は非線形変換ρの出力のうち半分を乱数列とする。



上の定義より、任意の時間 t において、

$$a^{(t+1)} = \rho'(a^{(t)}) + K_t$$

と書けるから、攻撃者は K_t の値を常に知ることができる。すなわち、攻撃者は、1 ラウンドにつきバッファの状態を 512 ビット入手できる。この結果、TOY1 の 21 ラウンドの出力を得ることで、バッファの状態を完全に決定することができる。

TOY2：ρ=ρ'と考える(バッファの挿入を無視する)

(1) TOY2 では、

$$a^{(t+1)}{}_L = \rho'(a^{(t)})_L$$

なる関係式が成り立つ。この式をブール多項式で表現すると、

$$c_{9,j} = a_{12,j+13} + (a_{13,j+13}+1)a_{14,j+13} + a_{2,j+23} + (a_{3,j+23}+1)a_{4,j+23} + a_{6,j+27} + (a_{7,j+27}+1)a_{8,j+27} + 1$$

$$c_{10,j} = a_{2,j+23} + (a_{3,j+23}+1)a_{4,j+23} + a_{9,j+2} + (a_{10,j+2}+1)a_{11,j+2} + a_{13,j+9} + (a_{14,j+9}+1)a_{15,j+9} + 1$$

$$c_{11,j} = a_{9,j+2} + (a_{10,j+2}+1)a_{11,j+2} + a_{16,j+14} + (a_{0,j+14}+1)a_{1,j+14} + a_{3,j+24} + (a_{4,j+24}+1)a_{5,j+24} + 1$$

$$c_{12,j} = a_{16,j+14} + (a_{0,j+14}+1)a_{1,j+14} + a_{6,j+27} + (a_{7,j+27}+1)a_{8,j+27} + a_{10,j+8} + (a_{11,j+8}+1)a_{12,j+8} + 1$$

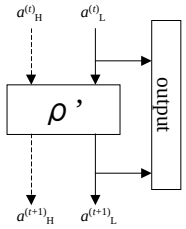
$$c_{13,j} = a_{6,j+27} + (a_{7,j+27}+1)a_{8,j+27} + a_{13,j+9} + (a_{14,j+9}+1)a_{15,j+9} + a_{0,j} + (a_{1,j}+1)a_{2,j} + 1$$

$$c_{14,j} = a_{13,j+9} + (a_{14,j+9}+1)a_{15,j+9} + a_{3,j+24} + (a_{4,j+24}+1)a_{5,j+24} + a_{7,j+1} + (a_{8,j+1}+1)a_{9,j+1} + 1$$

$$c_{15,j} = a_{3,j+24} + (a_{4,j+24}+1)a_{5,j+24} + a_{10,j+8} + (a_{11,j+8}+1)a_{12,j+8} + a_{14,j+3} + (a_{15,j+3}+1)a_{16,j+3} + 1$$

$$c_{16,j} = a_{10,j+8} + (a_{11,j+8}+1)a_{12,j+8} + a_{0,j} + (a_{1,j}+1)a_{2,j} + a_{4,j+6} + (a_{5,j+6}+1)a_{6,j+6} + 1$$

となる。ただし、上の式で a_{ij} は第 i ワード、第 j ビットを意味する。この式で、特に出力の第 11 ワード、第 15 ワードに注目すると、

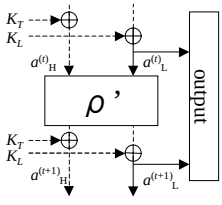


$$c_{15,j} = c_{11,j} + (t \text{ ラウンド目の出力から得られる値}) + (a_{0,j+14}+1)a_{1,j+14}$$

なる関係式が得られる。$(a_{0,j+18}+1)a_{1,j+18}$ の値は(他の値と独立であると仮定すれば)確率 1/4 で 1 である。したがって、非線形変換 ρ は(攻撃者が入手可能な情報からなる)非線形な相関性を持つことがわかる。このことから、TOY2 の出力する擬似乱数列を $c_{11,j}$ まで得た場合に、$c_{15,j}$ の値(1 ビット)を確率 3/4 で推定することができる。

(2) 前節の議論から、ρ'は 2^{-3} の確率で成立する(攻撃者が入手可能な情報からなる)線形相関式を持つことがわかる。したがって、(1)の方法と同様にして、2 ラウンドの出力から、乱数列の 1 ビットを確率 5/8 で推定することができる。

TOY3：バッファからの挿入を固定値 K とする。



TOY2(2)で述べた偏差を利用することで、ブロック暗号に対する線形解読法と同様に、K の 1 ビットを推定することができる。

4. 実装評価

4.1. ソフトウェア実装

MULTI-S01 暗号は、$\mathbf{F}2^{64}$ 上の乗算を用いるため、64 ビットプロセッサ上で効率の良い実装が可能である。ここではまず、MULTI-S01 を Alpha プロセッサ 600MHz 上で実装した結果について述べる。実装は C 言語で行った。評価環境の詳細は表 4.1を参照のこと。

表 4.1 ソフトウェア評価環境（DEC Alpha 21164A）

ハードウェア	CPU	Alpha 21164A 600 MHz*
	RAM	512 Mbyte
ソフトウェア	OS	DIGITAL UNIX 4.0E
	コンパイラ	DEC cc
	言語	C
	最適化オプション	-tune-ev56 -arch -ev56 O6

*:3 次キャッシュ 4 MB

MULTI-S01 暗号をこの環境で実装した場合、実装コストは表 4.2のとおり。

表 4.2 メモリ使用量

コード量		1167 行
ワークエリア	初期化	2.4Kbyte
	暗号化	3.6Kbyte
	復号化	3.7Kbyte

ここで、コード量はソースファイルから空白行、コメント行を取り除いた行数として測定した。また、ワークエリアの測定では、平文、暗号文を格納するメモリはワークエリアから除外した。

この環境において、MULTI-S01 暗号は、暗号化 270.7Mbps、復号化 267.3Mbps の処理を行うことができる。この結果を clock/byte に換算したものが表 4.3である。参考までに、PANAMA による擬似乱数生成の速度も記載した。評価結果は、4096byte の平文を繰り返し暗号化(復号化)することで求めた。また、初期化には 31737clock を要する。初期化は、PANAMA の初期化、乱数 A およびその逆元 A^{-1} の計算、乗算の表(2KB)の領域確保、 A についての乗算の表の作成に要する時間である。

表 4.3 ソフトウェア上での速度（DEC Alpha 21164A）

測定内容	速度 (clock/byte)
暗号化	17.7
復号化	18.0
擬似乱数の生成	6.7

次に、Intel 社製、Pentium(R) III プロセッサによる実装の結果をまとめる。前述のとおり、64 ビット単位の処理が効率的であるため、ここでは MMX 命令を利用したアセンブラ実装によるプログラミングを行った。この結果、表 4.4 に示す測定結果が得られた。測定に用いられた装置は Pentium(R) III 600 MHz のプロセッサで 1 次キャッシュは 256 KB である。この測定は 32 KB の平文に対する処理を測定したものである。なお、同じ環境で平文長、1MB の場合、暗号化の処理速度は、21.75 clock/byte であった。

表 4.4 ソフトウェア上での速度（Pentium(R) III：平文長 32KB）

測定内容	速度 (clock/byte)
暗号化	17.62
復号化	18.52

最後に Hitachi、メディアプロセッサ MAP1000 での実装結果についてまとめる。MAP1000 200MHz は 2 クラスタを内蔵する VLIW アーキテクチャによるプロセッサである。今回はアセンブリ言語による特別命令を用いながらの実装により、64 ビットの乗算を効率的に実装し、表 4.5に示すソフトウェア処理速度を達成することができた。

表 4.5 ソフトウェア上での速度（MAP1000：平文長 32KB）

測定内容	速度 (clock/byte)
暗号化	13.69
復号化	14.18

4.2.ハードウェア実装

MULTI-S01 暗号は、擬似乱数生成部を除けば、排他的論理和および $\mathbf{F}2^{64}$ 上の乗算からなる簡単な構成である。このような特長から、MULTI-S01 をハードウェア実装した場合、ひじょうに高速な処理が可能である。実際、ハードウェア実装をシミュレートした結果、回路規模 140k ゲートで処理速度 9.1Gbps を得る。本評価では、0.35μm の CMOS 技術を想定して評価を行った。

4.2.1. 論理回路の設計

本評価では、(1)処理速度優先、(2)論理規模優先の 2 つの観点からそれぞれ設計を行った。実装では、セルライブラリとして日立 HG73C(0.35μm ルール)を用い、論理規模見積もりを行なった。動作速度は、ゲート遅延の最小値を用いて見積もっており、実際にはこれ以下の数値となる可能性もある。

擬似乱数生成部(PANAMA)

仕様:
内部レジスタ: K 値レジスタ(256bit)、Q 値レジスタ(256bit)
入力: クロック clock
内部処理: clock 同期で動作。但し、a レジスタ、b レジスタ、λ、ρは 4 分周 clock で動作し、4 分周 clock ごとに 256bit の擬似乱数を生成する。
出力: 終段で P/S 処理を施し、clock ごとに 64bit の擬似乱数を出力する B_t(B_1～B_n)バス(64bit)。B_1出力時に A を出力する A バス(64bit)。B_{n-1}出力時に S を出力する S バス(64bit)。

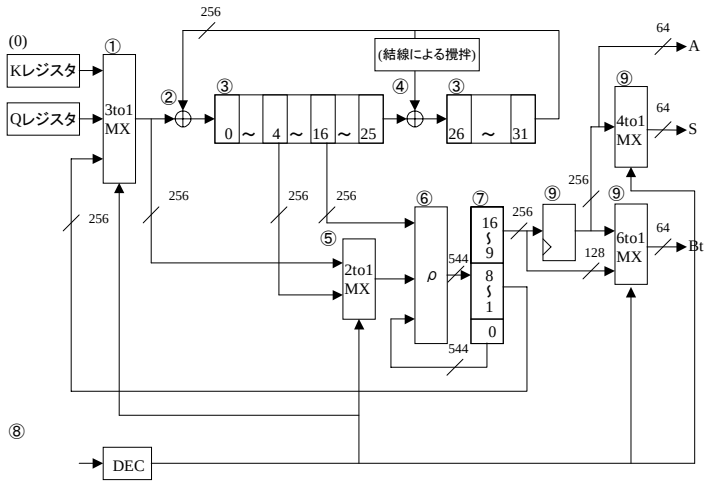


図 4.1 **擬似乱数生成部のブロック図**

処理速度優先実装結果を図 3 に示す。λはλ_0(2)、λ_1(4)の二つに分けて実装する。又、(0)～(9)の各部位に要するコアマクロを表 4.6に示す。

表 4.6 擬似乱数生成器に必要なコアマクロ

	内容		コアマクロ	個数
(0)	K**値レジスタ、**Q**値レジスタ**		DFF	512
(1)	push/pull**モードで**b0**に足し込む値を選択する**		3to1MX	256
(2)	λ_0	b31(+)①**出力**	2in-EOR	256
(3)	b**レジスタ**		DFF	8192
(4)	λ_1	b31**の攪拌**	**結線のみ**	-
		攪拌済b31(+)b25	2inEOR	256
(5)	push/pull**モードで** ρ **に入力値を選択する**		3to1MX	256
(6)	ρ	γ	2in-inv	544
			2in-OR	544
			2in-EOR	544
		π	**結線のみ**	-
		θ	3in-EOR	544
		δ	2in-EOR	544
(7)	a**レジスタ**		DFF	544
(8)	**初期化シーケンスカウンタ**		DFF	6
	MX**選択信号**		+α	+α
(9)	**ラストパディング用一時レジスタ**		DFF	256
	B_t**値セレクタ**		6to1MX	64
	S**値セレクタ**		4to1MX	64

逆元演算

$\alpha^{-1}=\alpha^{(2\wedge 64-2)}$は以下の手順で計算する。

表 4.7 α^{-1}の演算

値	内容
S_0	α
S_1	S_0^2・α
S_2	S_1^2・α
S_3	S_2^2・α
…	…
S_{62}	S_{61}^2・α
S_{63}	S_{62}^2・1

$(2^{64}-2)$d=FFFF FFFEh であるので、
$S_n=(S_{n-1})^2 \times \alpha$ $(1 \leqq n<63)$
$S_n=(S_{n-1})^2$ $(n=63)$
$S_{63}=\alpha^{-1}$である。二乗とα乗算を合わせた乗算回数は 125 回となる。
乗算器をデータ攪拌部と共有することとした逆元演算器の構成を図 4.1に示す。

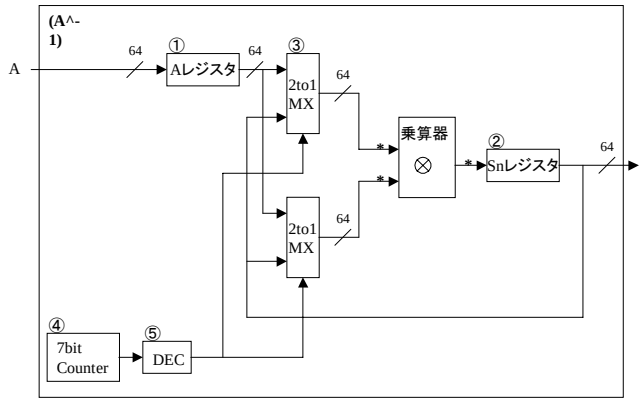


図 4.2 **逆元演算器のブロック図**

仕様:
内部レジスタ: 鍵データ A 値レジスタ(64bit)、S_n 一時保管レジスタ(64bit)。
入力: A 値(64bit)、クロック clock。
内部処理: 処理シーケンサから出力するパス選択信号で各演算のステージを管理する。
出力: A^{-1} 値(64bit)。

演算回路に必要なコアマクロを表 5 にまとめる。データ攪拌部と乗算器を共有する場合、表 3.1中の乗算器部分は不要となる。独立して具備する場合、⑤のセレクタは不要である。

表 4.8 α^{-1} の演算に必要なコアマクロ

	内容		コアマクロ	個数
①	a**レジスタ**		DFF	64
②	S_n**レジスタ**		DFF	64
③	**演算パス選択**		2to1MX	128
④	**演算シーケンスカウンタ**		DFF	7
	MX**選択信号**		+**α**	+**α**
⑤	**乗算器共用のためのパス選択**		2to1MX	256
	乗算器(速度優先)	**本体**	-	(36k)
		+**α**	-	(1.3k)
	乗算器(論理優先)	**本体**	-	(0.9k)
		+**α**	-	(1.3k)

データ攪拌部

仕様:
内部レジスタ: 鍵データ A 値レジスタ(64bit)、鍵データ B_t 値レジスタ(64bit)、冗長データ R 値レジスタ(64bit)、平文/暗号文データ P_t/C_t 値レジスタ(64bit)、暗号文/平文データ C_t/P_t 値レジスタ(64bit)。
入力: A 値(64bit)…擬似乱数生成部と直結、B_t 値(64bit) …擬似乱数生成部と直結、外部入力 M(64bit)…外部 P_t/C_t 値、S 値(64bit)…擬似乱数生成部と直結、クロック clock。
内部処理: P_t/C_t、S、R、A、B_t から clock ごとに P_t/C_t を生成する。
出力: C_t/P_t 値(64bit)。

データ攪拌部の構成を図 4.3に示す。データ攪拌部本体は一次評価の結果をそのまま用いており、P_t/C_t 入力選択、逆元演算器と加えている。又、IV として 1 を与える。(2)に示した逆元演算器と乗算器を共有する場合、一次評価で示したデータ攪拌部のデータパス上に、データセレクタを挿入する。挿入するデータセレクタ(2to1MX)は、表 4.8⑤としてある。又、図 4.2、図 4.3上のデータセレクタ挿入位置を*で示した。

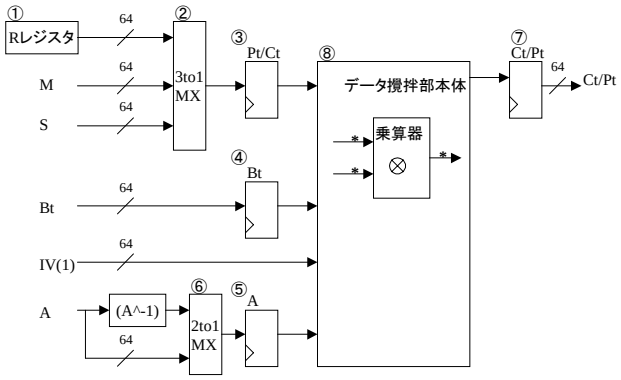


図 4.3 データ攪拌部のブロック図(全体図)

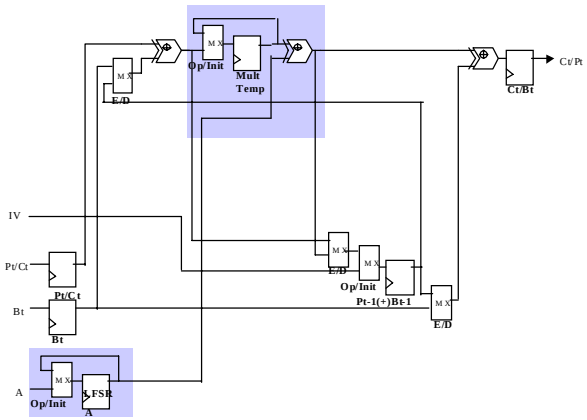


図 4.4 **データ攪拌部のブロック図(処理速度優先)**

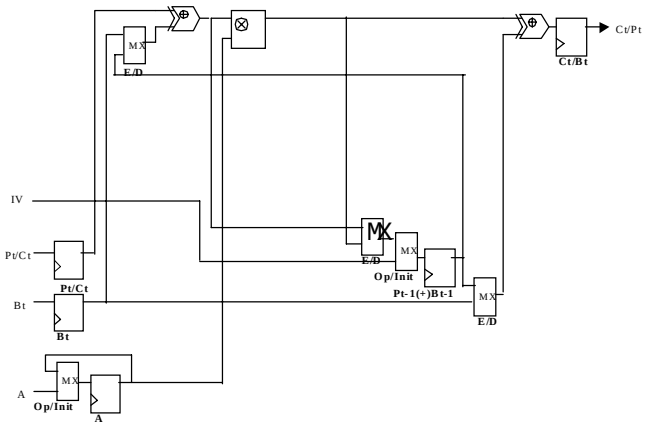


図 4.5 データ攪拌部のブロック図(論理規模優先)

表 4.9 論理規模見積もり結果

実装方式	処理速度優先	論理規模優先
論理規模(ゲート数)	38.5k	3k
動作速度(Hz)	150M	800M
64bit処理に必要なクロック数	1clock	65clock
処理スループット(bps)	9.6G	800M

表 4.10 データ攪拌部に必要なコアマクロ(処理速度優先)

	内容	コアマクロ	個数
①	Rレジスタ	DFF	64
②	P_t/C_t選択	3to1MX	64
③	P_t/C_tレジスタ	DFF	64
④	B_tレジスタ	DFF	64

⑤	Aレジスタ		DFF	64
⑥	A/A^{-1}選択		2to1MX	64
⑦	C_t/P_tレジスタ		DFF	64
⑧	乗算器	本体	-	(36k)
		+α	-	(1.3k)

表 4.11 データ攪拌部に必要なコアマクロ(論理規模優先)

	内容		コアマクロ	個数
①	Rレジスタ		DFF	64
②	P_t/C_t選択		3to1MX	64
③	P_t/C_tレジスタ		DFF	64
④	B_tレジスタ		DFF	64
⑤	Aレジスタ		DFF	64
⑥	A/A^{-1}選択		2to1MX	64
⑦	C_t/P_tレジスタ		DFF	64
⑧	乗算器	本体	-	(0.9k)
		+α	-	(1.3k)

4.2.2. 論理規模および動作速度の見積もり

前節の回路について、セルライブラリとして日立 HG73C(0.35μm ルール)を用いた論理規模見積もりを行なった。論理規模の概算結果を表 4.12に、動作速度の概算結果を表 4.13に示す。

表 4.12 論理規模見積もり結果

実装方式	処理速度優先		論理規模優先	
	乗算器共有	乗算器独立	乗算器共有	乗算器独立
擬似乱数生成部	61.5k			
逆元計算部	1.8k	38.4k	1.8k	3.3k
データ攪拌部	39.6k	39.6k	4.5k	4.5k
合計	102.9k	139.5k	67.8k	69.3k

表 4.13 最大遅延見積もり結果

実装方式	最大遅延			
	処理速度優先		論理規模優先	
	乗算器共有	乗算器独立	乗算器共有	乗算器独立
擬似乱数生成部	1.36ns			
逆元計算部	6.88ns	6.16ns	1.48ns	1.12ns
データ攪拌部	7.71ns	6.99ns	1.59ns	1.23ns
動作周波数	130MHz	140MHz	620MHz (200MHz)	730MHz (200MHz)
A^{-1}計算時間	1μs	0.8μs	13μs (40μs)	11μs (40μs)
処理スループット	8.3Gbps	9.1Gbps	620Mbps (200Mbps)	730Mbps (200Mbps)

擬似乱数生成器
クリティカルパス: (0)-(1)-(5)-(6)-(7)
構成 3to1MX 一段、2to1MX 一段、EOR 一段、DFF 一段
遅延時間 1.36ns
⑨を除く最大動作周波数 730MHz
データ攪拌部よりも高速に動作し、且つ 1 クロックで生成される擬似乱数が、データ攪拌部で処理されるデータの 4 倍(64bit に対して 256bit)であるため、データ攪拌部よりも確実に処理スループットが高い。このため、擬似乱数生成部が処理のボトルネックとなることはなく、全体の処理速度はデータ攪拌部に依存する。

逆元演算器
(乗算器共有、速度優先乗算器)
クリティカルパス: (1)-(3)-(2)
構成 2to1MX 三段、乗算器一段、DFF 一段
遅延時間 6.88ns
(乗算器共有、論理規模優先乗算器)
クリティカルパス: (1)-(3)-(2)
構成 2to1MX 三段、DFF 一段
遅延時間 1.48ns
(乗算器独立、速度優先乗算器)
クリティカルパス: (1)-(3)-(2)
構成 2to1MX 一段、乗算器一段、DFF 一段
遅延時間 6.16ns
(乗算器独立、論理規模優先乗算器)
クリティカルパス: (1)-(3)-(2)
構成 2to1MX 二段、DFF 一段
遅延時間 1.12ns

データ攪拌部
(乗算器共有、速度優先乗算器)
構成 2to1MX 五段、EOR 一段、乗算器一段、DFF 一段
遅延時間 7.71ns
(乗算器共有、論理規模優先乗算器)
構成 2to1MX 三段、EOR 一段、DFF 一段
遅延時間 1.59ns
(乗算器独立、速度優先乗算器)
構成 2to1MX 三段、EOR 一段、乗算器一段、DFF 一段
遅延時間 6.99ns
(乗算器独立、論理規模優先乗算器)
構成 2to1MX 二段、EOR 一段、DFF 一段
遅延時間 1.23ns

結論

処理速度優先実装時には、140k ゲート、動作周波数 140MHz により、9.1Gbps の処理スループットを得る見通しを得た。
論理規模優先実装には、68k ゲート、動作周波数 620(200)MHz により、620(200)Mbps の処理スループットを得る見通しを得た。

5. 参考文献

[1] ANSI X9.9, ``American National Standard for Financial Institution Message Authentication (Wholesale)," *American Bankers Association*, 1981. Revised 1986.

[2] Anderson, R., Biham, E., ``Two Practical and Provably Secure Block Ciphers: BEAR and LION," *Fast Software Encryption, Third International Workshop, Cambridge, UK, February, LNCS Vol.1039, Springer-Verlag*, 1996.

[3] 青木和麻呂, 太田和夫, ``ソフトウェアによる $\mathbf{F}2^n$ 上の演算,"暗号と情報セキュリティシンポジウム講演予稿集 SCIS'97-14A, 1997.

[4] Bellare, M., Kilian, J., Rogaway, P., ``On the security of cipher block chaining," *Advances in Cryptology, -CRYPTO'94, LNCS Vol. 839, Springer-Verlag*, 1994.

[5] Biryukov, A. and Kushilevitz, E., ``From Differencial Cryptanalysis to Ciphertext-Only Attacks," Advances in Cryptology -CRYPTO'98, Proceedings, LNCS1462, Springer-Verlag, 1998.

[6] Black, J., Halevi, S., Krawczyk, H., Krovets, T., Rogaway, P., ``UMAC: Fast and Secure Message Authentication," *Advances in Cryptology, -CRYPTO'99, LNCS Vol. 1666, Springer-Verlag*, 1999.

[7] Daemen, J., ``Cipher and hash function design strategies based on linear and differential cryptanalysis," *Doctoral Dissertation*, March 1995, K. U. Leuven.

[8] Daemen, J., Clapp, C., ``Fast Hashing and Stream Encryption with PANAMA," Fast Software Encryption, 5th International Workshop, FSE'98, Proceedings, LNCS1372, Springer-Verlag, 1998.

[9] FIPS 140-1, *Security Requirements for Cryptographic Modules*, Federal Information Processing Statndard(FIPS), Publication 140-1, National Institute of Standards adn Technology, US Department of Commerce, Washington D.C., January 1994,
`http://www.itl.nist.gov/fipspubs/index.htm.`

[10] 古屋聡一, 佐藤尚宜, 高橋昌史, 宮崎邦彦, 宝木和夫, 佐々木良一 ``メッセージ認証可能なストリーム暗号操作モード," 暗号と情報セキュリティシンポジウム講演予稿集 SCIS2000-A17, 2000.

[11] Gligor, V., Donescu, P., ``Integrity-Aware PCBC Encryption Schemes," *The 1999 Security Protocols Workshop Pre-proceedings*, Cambridge, 1999.

[12] Halevi, S., Krawczyk, H., ``MMH: Software message authentication in the Gbit/second rates," *Fast Software Encryption, 4th International Workshop, FSE'97, LNCS Vol. 1267, Springer-Verlag*, 1997.

[13] 日立 CMOS セルベース IC HG73C/M シリーズセルライブラリ コア 3.3V 用

[14] Katz, J., Yung, M., ``Unforgeable Encryption and Chosen Cipher Secure Modes of Operation,'' *Fast Software Encryption, 7th International Workshop, FSE2000, Pre-proceedings*, 2000.

[15] Knuth, Donald E., ``*The Art of Computer Programming Volume II (2nd ed.)*,'' Addison-Wesley, 1981.

[16] Jakubowski, M. H., Venkatesan, R., ``The Chain & Sum Primitive and Its Applications to MACs and Stream Ciphers,'' *Advances in Cryptology-EUROCRYPT'98, LNCS Vol. 1403, Springer-Verlag*}, 1998.

[17] Menezes, Alfred J., van Oorschot, Paul C., Vanstone, Scott A., ``*HANDBOOK of APPLIED CRYPTOGRAPHY*,'' CRC Press, 1997.

[18] Rijmen, V., B. Rompay, B. V., Preneel, B., Vandewalle, J., ``Producing Collisions for PANAMA,'' Preproceedings of FSE2001, 8th Fast Software Encryption Worksop, Yokohama Japan, 2001.

[19] Roe, M., ``Cryptography and Evidence,'' *Doctoral Dissertation with the University of Cambridge*, 1997, available at **http://www.ccsr.cam.ac.uk/techreports/index.html.**

[20] Schneier, B., ``Block and Stream Ciphers,'' Crypto-Gram, January 15, 2000, available at **http://www.counterpane.com/crypto-gram-0001. html.**

[21] Shannon, C. E., ``A Mathematical Theory of Communication,'' *Bell System Technical Journal, Vol. 27, No. 4*, 1948.

[22] Shoup, V., ``On Fast and Provably Secure Message Authentication Based on Universal Hashing,'' preliminary version appears in *Advances in Cryptology-CRYPTO'96, LNCS Vol. 1109, Springer-Verlag*, 1996.

[23] Taylor, R., ``An Integrity Check Value Algorithm for Stream Ciphers,'' *Advances in Cryptology - CRYPTO'93, LNCS Vol. 773, Springer-Verlag*, 1993.

[24] Wegman, R., Carter, L., ``New hash functions and their use in authentication and set equality,'' *Journal of Computer and System Sciences, 22:265-279*, 1981.