(19)日本国特許庁(JP)　(12)特　許　公　報(B2)　(11)特許番号
特許第5338945号
(P5338945)
(45)発行日　平成25年11月13日(2013.11.13)　(24)登録日　平成25年8月16日(2013.8.16)

(51)Int.Cl.　F I
H04L 9/06 (2006.01)　H04L 9/00 611Z
G09C 1/00 (2006.01)　G09C 1/00 610A

請求項の数 18　(全 37 頁)

(21)出願番号　特願2012-116685(P2012-116685)
(22)出願日　平成24年5月22日(2012.5.22)
(62)分割の表示　特願2006-238225(P2006-238225)の分割
原出願日　平成18年9月1日(2006.9.1)
(65)公開番号　特開2012-155349(P2012-155349A)
(43)公開日　平成24年8月16日(2012.8.16)
審査請求日　平成24年5月22日(2012.5.22)

(73)特許権者　000002185
ソニー株式会社
東京都港区港南１丁目７番１号
(74)代理人　100093241
弁理士　宮田　正昭
(74)代理人　100101801
弁理士　山田　英治
(74)代理人　100086531
弁理士　澤田　俊夫
(74)代理人　100095496
弁理士　佐々木　榮二
(74)代理人　110000763
特許業務法人大同特許事務所

最終頁に続く

(54)【発明の名称】復号処理装置、情報処理装置、および復号処理方法、並びにコンピュータ・プログラム

(57)【特許請求の範囲】
【請求項１】
入力データを２以上の数で分割して得られる各データ系列に対し、Ｓボックスを含むＦ関数によるデータ変換処理をラウンド関数として各々実行する復号処理部を有し、
前記復号処理部は、入力データの系列と出力データの系列が同一であり、かつ隣り合う位置にあるＦ関数において対応するビット列に適用されるＳボックスが異なる構成であり、
前記復号処理部は、
ｓビットの入出力をもつＳボックスとして、
（１）タイプ１：拡大体ＧＦ（２^{s}）上の逆元写像：Ｙ＝Ｘ$^{-1}$、または、べき乗関数Ｙ＝Ｘ^{q}を利用したＳボックス、
（２）タイプ２：ｔ－ｂｉｔの小さなＳ－ｂｏｘを複数組み合わせて作り出されたＳボックス、ただしｔ＜ｓとする。
（３）タイプ３：ランダムに選択されたＳボックス、
上記（１）～（３）の３タイプのＳボックス中、少なくとも２つ以上の異なるタイプのＳボックスを利用した構成である復号処理装置。
【請求項２】
前記復号処理部は、前記Ｓボックスを、
（ａ）一部をタイプ１のＳボックスとし、その他をタイプ２のＳボックスとした構成、
（ｂ）一部をタイプ１のＳボックスとし、その他をタイプ３のＳボックスとした構成、

（ｃ）一部をタイプ２のＳボックスとし、その他をタイプ３のＳボックスとした構成、
（ｄ）一部をタイプ１のＳボックスとし、他の一部をタイプ２のＳボックスとし、その他をタイプ３のＳボックスとした構成、
上記（ａ）～（ｄ）のいずれかの構成を有することを特徴とする請求項１に記載の復号処理装置。
【請求項３】
前記復号処理部は、
ラウンド関数実行部内に、処理対象データを分割した分割データ各々の非線形変換処理を実行する複数のＳボックスを有し、
１つのラウンドでは１種類のタイプのＳボックスを利用し、ラウンド単位で異なるタイプのＳボックスを利用した処理を実行する構成であることを特徴とする請求項１に記載の復号処理装置。
【請求項４】
前記復号処理部は、
ラウンド関数実行部内に、処理対象データを分割した分割データ各々の非線形変換処理を実行する複数のＳボックスを有し、
１つのラウンドで、複数のタイプのＳボックスを利用した構成であることを特徴とする請求項１に記載の復号処理装置。
【請求項５】
前記ラウンド関数実行部の各々に含まれるＳボックスの種類および各Ｓボックスの数は、各Ｆ関数において同一の設定であることを特徴とする請求項４に記載の復号処理装置。
【請求項６】
前記復号処理部は、
共通鍵暗号方式に従った復号処理を実行する構成であることを特徴とする請求項１～５いずれかに記載の復号処理装置。
【請求項７】
前記復号処理部は、
共通鍵ブロック暗号方式に従った復号処理を実行する構成であることを特徴とする請求項１～５いずれかに記載の復号処理装置。
【請求項８】
入力データを２以上の数で分割して得られる各データ系列に対し、Ｓボックスを含むＦ関数によるデータ変換処理をラウンド関数として各々実行する復号処理部と、
前記復号処理部における復号処理の制御を行う制御部と、
前記制御部の実行する復号処理制御プログラムを格納したメモリを有し、
前記復号処理部は、入力データの系列と出力データの系列が同一であり、かつ隣り合う位置にあるＦ関数において対応するビット列に適用されるＳボックスが異なる構成であり、
前記復号処理部は、
ｓビットの入出力をもつＳボックスとして、
（１）タイプ１：拡大体ＧＦ（２^{s}）上の逆元写像：Ｙ＝Ｘ$^{-1}$、または、べき乗関数Ｙ＝Ｘ^{q}を利用したＳボックス、
（２）タイプ２：ｔ－ｂｉｔの小さなＳ－ｂｏｘを複数組み合わせて作り出されたＳボックス、ただしｔ＜ｓとする。
（３）タイプ３：ランダムに選択されたＳボックス、
上記（１）～（３）の３タイプのＳボックス中、少なくとも２つ以上の異なるタイプのＳボックスを利用した構成である情報処理装置。
【請求項９】
前記復号処理部は、前記Ｓボックスを、
（ａ）一部をタイプ１のＳボックスとし、その他をタイプ２のＳボックスとした構成、
（ｂ）一部をタイプ１のＳボックスとし、その他をタイプ３のＳボックスとした構成、

（ｃ）一部をタイプ２のＳボックスとし、その他をタイプ３のＳボックスとした構成、
（ｄ）一部をタイプ１のＳボックスとし、他の一部をタイプ２のＳボックスとし、その他をタイプ３のＳボックスとした構成、
上記（ａ）～（ｄ）のいずれかの構成を有することを特徴とする請求項８に記載の情報処理装置。
【請求項１０】
前記復号処理部は、
ラウンド関数実行部内に、処理対象データを分割した分割データ各々の非線形変換処理を実行する複数のＳボックスを有し、
１つのラウンドでは１種類のタイプのＳボックスを利用し、ラウンド単位で異なるタイプのＳボックスを利用した処理を実行する構成であることを特徴とする請求項８に記載の情報処理装置。
【請求項１１】
前記復号処理部は、
ラウンド関数実行部内に、処理対象データを分割した分割データ各々の非線形変換処理を実行する複数のＳボックスを有し、
１つのラウンドで、複数のタイプのＳボックスを利用した構成であることを特徴とする請求項８に記載の情報処理装置。
【請求項１２】
前記ラウンド関数実行部の各々に含まれるＳボックスの種類および各Ｓボックスの数は、各Ｆ関数において同一の設定であることを特徴とする請求項１１に記載の情報処理装置。
【請求項１３】
前記復号処理部は、
共通鍵暗号方式に従った復号処理を実行する構成であることを特徴とする請求項８～１２いずれかに記載の情報処理装置。
【請求項１４】
前記復号処理部は、
共通鍵ブロック暗号方式に従った復号処理を実行する構成であることを特徴とする請求項８～１２いずれかに記載の情報処理装置。
【請求項１５】
復号処理装置において、復号処理を実行する復号処理方法であり、
復号処理部において、入力データを２以上の数で分割して得られる各データ系列に対し、Ｓボックスを含むＦ関数によるデータ変換処理をラウンド関数として各々実行する復号処理ステップを実行し、
前記復号処理ステップにおいて、入力データの系列と出力データの系列が同一であり、かつ隣り合う位置にあるＦ関数において対応するビット列に適用するＳボックスを異なる設定のＳボックスを適用したデータ変換処理を実行し、
前記復号処理ステップにおいて、
ｓビットの入出力をもつＳボックスとして、
（１）タイプ１：拡大体ＧＦ（２^{s}）上の逆元写像：$Y=X^{-1}$、または、べき乗関数$Y=X^{q}$を利用したＳボックス、
（２）タイプ２：ｔ－ｂｉｔの小さなＳ－ｂｏｘを複数組み合わせて作り出されたＳボックス、ただしｔ＜ｓとする。
（３）タイプ３：ランダムに選択されたＳボックス、
上記（１）～（３）の３タイプのＳボックス中、少なくとも２つ以上の異なるタイプのＳボックスを利用したデータ変換処理を実行する復号処理方法。
【請求項１６】
前記復号処理ステップは、
共通鍵暗号方式または共通鍵ブロック暗号方式に従った復号処理を実行することを特徴

とする請求項１５に記載の復号処理方法。
【請求項１７】
復号処理装置において、復号処理を実行させるコンピュータ・プログラムであり、
復号処理部において、入力データを２以上の数で分割して得られる各データ系列に対し、Ｓボックスを含むＦ関数によるデータ変換処理をラウンド関数として各々実行する復号処理ステップを実行させ、
前記復号処理ステップにおいて、入力データの系列と出力データの系列が同一であり、かつ隣り合う位置にあるＦ関数において対応するビット列に適用するＳボックスを異なる設定のＳボックスを適用したデータ変換処理を行なわせ、
前記復号処理ステップにおいて、
ｓビットの入出力をもつＳボックスとして、
（１）タイプ１：拡大体ＧＦ（２^{s}）上の逆元写像：Ｙ＝Ｘ$^{-1}$、または、べき乗関数Ｙ＝Ｘ^{q}を利用したＳボックス、
（２）タイプ２：ｔ－ｂｉｔの小さなＳ－ｂｏｘを複数組み合わせて作り出されたＳボックス、ただしｔ＜ｓとする。
（３）タイプ３：ランダムに選択されたＳボックス、
上記（１）～（３）の３タイプのＳボックス中、少なくとも２つ以上の異なるタイプのＳボックスを利用したデータ変換処理を実行させるコンピュータ・プログラム。
【請求項１８】
前記復号処理ステップは、
共通鍵暗号方式または共通鍵ブロック暗号方式に従った復号処理を実行させるステップであることを特徴とする請求項１７に記載のコンピュータ・プログラム。
【発明の詳細な説明】
【技術分野】
【０００１】
本発明は、復号処理装置、情報処理装置、および復号処理方法、並びにコンピュータ・プログラムに関する。さらに詳細には、例えば、共通鍵ブロック暗号を適用した復号処理を実行する復号処理装置、情報処理装置、および復号処理方法、並びにコンピュータ・プログラムに関する。
【背景技術】
【０００２】
昨今、ネットワーク通信、電子商取引の発展に伴い、通信におけるセキュリティ確保が重要な問題となっている。セキュリティ確保の１つの方法が暗号技術であり、現在、様々な暗号化手法を用いた通信が実際に行なわれている。
【０００３】
例えばＩＣカード等の小型の装置中に暗号処理モジュールを埋め込み、ＩＣカードと、データ読み取り書き込み装置としてのリーダライタとの間でデータ送受信を行ない、認証処理、あるいは送受信データの暗号化、復号を行なうシステムが実用化されている。
【０００４】
暗号処理アルゴリズムには様々なものがあるが、大きく分類すると、暗号化鍵と復号鍵を異なる鍵、例えば公開鍵と秘密鍵として設定する公開鍵暗号方式と、暗号化鍵と復号鍵を共通の鍵として設定する共通鍵暗号方式とに分類される。
【０００５】
共通鍵暗号方式にも様々なアルゴリズムがあるが、その１つに共通鍵をベースとして複数の鍵を生成して、生成した複数の鍵を用いてブロック単位（６４ビット，１２８ビットなど）のデータ変換処理を繰り返し実行する方式がある。このような鍵生成方式とデータ変換処理を適用したアルゴリズムの代表的なものが共通鍵ブロック暗号方式である。
【０００６】
代表的な共通鍵ブロック暗号のアルゴリズムとしては，例えば過去に米国標準暗号であったＤＥＳ（Data Encryption Standard）アルゴリズム，現在の米国標準暗号であるＡＥ

Ｓ（Advanced Encryption Standard）アルゴリズムなどが知られている。
【０００７】
このような、共通鍵ブロック暗号のアルゴリズムは、主として、入力データの変換を繰り返し実行するラウンド関数実行部を有する暗号処理部と、ラウンド関数部の各ラウンドで適用するラウンド鍵を生成する鍵スケジューリング部とによって構成される。鍵スケジューリング部は、秘密鍵であるマスター鍵（主鍵）に基づいて、まずビット数を増加させた拡大鍵を生成し、生成した拡大鍵に基づいて、暗号処理部の各ラウンド関数部で適用するラウンド鍵（副鍵）を生成する。
【０００８】
このようなアルゴリズムを実行する具体的な構造として、線形変換部および非線形変換部を有するラウンド関数を繰り返し実行する構造が知られている。例えば代表的な構造にＦｅｉｓｔｅｌ構造がある。Ｆｅｉｓｔｅｌ構造は、データ変換関数としてのラウンド関数（Ｆ関数）の単純な繰り返しにより、平文を暗号文に変換する構造を持つ。ラウンド関数（Ｆ関数）においては、線形変換処理および非線形変換処理が実行される。なお、Ｆｅｉｓｔｅｌ構造を適用した暗号処理について記載した文献としては、例えば非特許文献１、非特許文献２がある。
【０００９】
しかし、この共通鍵ブロック暗号処理の問題点として、暗号解析による鍵の漏洩がある。暗号解析による鍵の解析が容易であるということは、その暗号処理の安全性が低いということになり、実用上、大きな問題となる。
【先行技術文献】
【非特許文献】
【００１０】
【非特許文献１】K. Nyberg, "Generalized Feistel networks", ASIACRYPT'96, SpringerVerlag, 1996, pp.91--104.
【非特許文献２】Yuliang Zheng, Tsutomu Matsumoto, Hideki Imai: On the Construction of Block Ciphers Provably Secure and Not Relying on Any Unproved Hypotheses. CRYPTO 1989: 461-480
【発明の開示】
【発明が解決しようとする課題】
【００１１】
本発明は、上記問題点に鑑みてなされたものであり、暗号解析の困難性を高め、安全性の高い共通鍵ブロック暗号アルゴリズムを適用した復号処理を実行する復号処理装置、情報処理装置、および復号処理方法、並びにコンピュータ・プログラムを提供することを目的とする。
【課題を解決するための手段】
【００１２】
本発明の第１の側面は、
入力データを２以上の数で分割して得られる各データ系列に対し、Ｓボックスを含むＦ関数によるデータ変換処理をラウンド関数として各々実行する復号処理部を有し、
前記復号処理部は、入力データの系列と出力データの系列が同一であり、かつ隣り合う位置にあるＦ関数において対応するビット列に適用されるＳボックスが異なる構成であり、
前記復号処理部は、
ｓビットの入出力をもつＳボックスとして、
（１）タイプ１：拡大体ＧＦ（２^{s}）上の逆元写像：Ｙ＝Ｘ$^{-1}$、または、べき乗関数Ｙ＝Ｘ^{q}を利用したＳボックス、
（２）タイプ２：ｔ－ｂｉｔの小さなＳ－ｂｏｘを複数組み合わせて作り出されたＳボックス、ただしｔ＜ｓとする。
（３）タイプ３：ランダムに選択されたＳボックス、

上記（１）～（３）の３タイプのＳボックス中、少なくとも２つ以上の異なるタイプのＳボックスを利用した構成である復号処理装置にある。

【００１３】

さらに、本開示の復号処理装置の一実施態様において、前記復号処理部は、前記Ｓボックスを、

（ａ）一部をタイプ１のＳボックスとし、その他をタイプ２のＳボックスとした構成、

（ｂ）一部をタイプ１のＳボックスとし、その他をタイプ３のＳボックスとした構成、

（ｃ）一部をタイプ２のＳボックスとし、その他をタイプ３のＳボックスとした構成、

（ｄ）一部をタイプ１のＳボックスとし、他の一部をタイプ２のＳボックスとし、その他をタイプ３のＳボックスとした構成、

上記（ａ）～（ｄ）のいずれかの構成を有することを特徴とする。

【００１４】

さらに、本開示の復号処理装置の一実施態様において、前記復号処理部は、ラウンド関数実行部内に、処理対象データを分割した分割データ各々の非線形変換処理を実行する複数のＳボックスを有し、１つのラウンドでは１種類のタイプのＳボックスを利用し、ラウンド単位で異なるタイプのＳボックスを利用した処理を実行する構成であることを特徴とする。

【００１５】

さらに、本開示の復号処理装置の一実施態様において、前記復号処理部は、ラウンド関数実行部内に、処理対象データを分割した分割データ各々の非線形変換処理を実行する複数のＳボックスを有し、１つのラウンドで、複数のタイプのＳボックスを利用した構成であることを特徴とする。

【００１６】

さらに、本開示の復号処理装置の一実施態様において、前記ラウンド関数実行部の各々に含まれるＳボックスの種類および各Ｓボックスの数は、各Ｆ関数において同一の設定であることを特徴とする。

【００１７】

さらに、本開示の復号処理装置の一実施態様において、前記復号処理部は、共通鍵暗号方式に従った復号処理を実行する構成であることを特徴とする。

【００１８】

さらに、本開示の復号処理装置の一実施態様において、前記復号処理部は、共通鍵ブロック暗号方式に従った復号処理を実行する構成であることを特徴とする。

【００１９】

さらに、本開示の第２の側面は、

入力データを２以上の数で分割して得られる各データ系列に対し、Ｓボックスを含むＦ関数によるデータ変換処理をラウンド関数として各々実行する復号処理部と、

前記復号処理部における復号処理の制御を行う制御部と、

前記制御部の実行する復号処理制御プログラムを格納したメモリを有し、

前記復号処理部は、入力データの系列と出力データの系列が同一であり、かつ隣り合う位置にあるＦ関数において対応するビット列に適用されるＳボックスが異なる構成であり、

前記復号処理部は、

ｓビットの入出力をもつＳボックスとして、

（１）タイプ１：拡大体ＧＦ（２^{s}）上の逆元写像：Ｙ＝Ｘ$^{-1}$、または、べき乗関数Ｙ＝Ｘ^{q}を利用したＳボックス、

（２）タイプ２：ｔ－ｂｉｔの小さなＳ－ｂｏｘを複数組み合わせて作り出されたＳボックス、ただしｔ＜ｓとする。

（３）タイプ３：ランダムに選択されたＳボックス、

上記（１）～（３）の３タイプのＳボックス中、少なくとも２つ以上の異なるタイプのＳボックスを利用した構成である情報処理装置にある。

【００２０】
さらに、本開示の情報処理装置の一実施態様において、前記復号処理部は、前記Ｓボックスを、
（ａ）一部をタイプ１のＳボックスとし、その他をタイプ２のＳボックスとした構成、
（ｂ）一部をタイプ１のＳボックスとし、その他をタイプ３のＳボックスとした構成、
（ｃ）一部をタイプ２のＳボックスとし、その他をタイプ３のＳボックスとした構成、
（ｄ）一部をタイプ１のＳボックスとし、他の一部をタイプ２のＳボックスとし、その他をタイプ３のＳボックスとした構成、
上記（ａ）～（ｄ）のいずれかの構成を有することを特徴とする。
【００２１】
さらに、本開示の情報処理装置の一実施態様において、前記復号処理部は、ラウンド関数実行部内に、処理対象データを分割した分割データ各々の非線形変換処理を実行する複数のＳボックスを有し、１つのラウンドでは１種類のタイプのＳボックスを利用し、ラウンド単位で異なるタイプのＳボックスを利用した処理を実行する構成であることを特徴とする。
【００２２】
さらに、本開示の情報処理装置の一実施態様において、前記復号処理部は、ラウンド関数実行部内に、処理対象データを分割した分割データ各々の非線形変換処理を実行する複数のＳボックスを有し、１つのラウンドで、複数のタイプのＳボックスを利用した構成であることを特徴とする。
【００２３】
さらに、本開示の情報処理装置の一実施態様において、前記ラウンド関数実行部の各々に含まれるＳボックスの種類および各Ｓボックスの数は、各Ｆ関数において同一の設定であることを特徴とする。
【００２４】
さらに、本開示の情報処理装置の一実施態様において、前記復号処理部は、共通鍵暗号方式に従った復号処理を実行する構成であることを特徴とする。
【００２５】
さらに、本開示の情報処理装置の一実施態様において、前記復号処理部は、共通鍵ブロック暗号方式に従った復号処理を実行する構成であることを特徴とする。
【００２６】
さらに、本開示の第３の側面は、
復号処理装置において、復号処理を実行する復号処理方法であり、
復号処理部において、入力データを２以上の数で分割して得られる各データ系列に対し、Ｓボックスを含むＦ関数によるデータ変換処理をラウンド関数として各々実行する復号処理ステップを実行し、
前記復号処理ステップにおいて、入力データの系列と出力データの系列が同一であり、かつ隣り合う位置にあるＦ関数において対応するビット列に適用するＳボックスを異なる設定のＳボックスを適用したデータ変換処理を実行し、
前記復号処理ステップにおいて、
ｓビットの入出力をもつＳボックスとして、
（１）タイプ１：拡大体ＧＦ（２^{s}）上の逆元写像：Ｙ＝Ｘ$^{-1}$、または、べき乗関数Ｙ＝Ｘ^{q}を利用したＳボックス、
（２）タイプ２：ｔ－ｂｉｔの小さなＳ－ｂｏｘを複数組み合わせて作り出されたＳボックス、ただしｔ＜ｓとする。
（３）タイプ３：ランダムに選択されたＳボックス、
上記（１）～（３）の３タイプのＳボックス中、少なくとも２つ以上の異なるタイプのＳボックスを利用したデータ変換処理を実行する復号処理方法にある。
【００２７】
さらに、本開示の復号処理方法の一実施態様において、前記復号処理ステップは、共通

鍵暗号方式または共通鍵ブロック暗号方式に従った復号処理を実行することを特徴とする。

【００２８】

さらに、本開示の第４の側面は、

復号処理装置において、復号処理を実行させるコンピュータ・プログラムであり、

復号処理部において、入力データを２以上の数で分割して得られる各データ系列に対し、Ｓボックスを含むＦ関数によるデータ変換処理をラウンド関数として各々実行する復号処理ステップを実行させ、

前記復号処理ステップにおいて、入力データの系列と出力データの系列が同一であり、かつ隣り合う位置にあるＦ関数において対応するビット列に適用するＳボックスを異なる設定のＳボックスを適用したデータ変換処理を行なわせ、

前記復号処理ステップにおいて、

ｓビットの入出力をもつＳボックスとして、

（１）タイプ１：拡大体ＧＦ（２^{s}）上の逆元写像：Ｙ＝Ｘ$^{-1}$、または、べき乗関数Ｙ＝Ｘ^{q}を利用したＳボックス、

（２）タイプ２：ｔ－ｂｉｔの小さなＳ－ｂｏｘを複数組み合わせて作り出されたＳボックス、ただしｔ＜ｓとする。

（３）タイプ３：ランダムに選択されたＳボックス、

上記（１）～（３）の３タイプのＳボックス中、少なくとも２つ以上の異なるタイプのＳボックスを利用したデータ変換処理を実行させるコンピュータ・プログラムにある。

【００２９】

さらに、本開示のコンピュータ・プログラムの一実施態様において、前記復号処理ステップは、共通鍵暗号方式または共通鍵ブロック暗号方式に従った復号処理を実行させるステップであることを特徴とする。

【００３０】

なお、本発明のコンピュータ・プログラムは、例えば、様々なプログラム・コードを実行可能なコンピュータ・システムに対して、コンピュータ可読な形式で提供する記憶媒体、通信媒体、例えば、ＣＤやＦＤ、ＭＯなどの記録媒体、あるいは、ネットワークなどの通信媒体によって提供可能なコンピュータ・プログラムである。このようなプログラムをコンピュータ可読な形式で提供することにより、コンピュータ・システム上でプログラムに応じた処理が実現される。

【００３１】

本発明のさらに他の目的、特徴や利点は、後述する本発明の実施例や添付する図面に基づくより詳細な説明によって明らかになるであろう。なお、本明細書においてシステムとは、複数の装置の論理的集合構成であり、各構成の装置が同一筐体内にあるものには限らない。

【発明の効果】

【００３２】

本発明の一実施例の構成によれば、共通鍵ブロック暗号を適用した復号処理において、ラウンド関数実行部に設定される非線形変換処理部としてのＳボックスに、少なくとも２種類以上の複数の異なるＳボックスを利用した構成とした。本構成により、飽和攻撃に対する耐性を高めることが可能となる。また、Ｓボックスのタイプとして、異なるタイプのものを混在させる本発明の一実施例の構成によれば、代数的攻撃（ＸＳＬ攻撃）に対する耐性を高めることが可能となり、安全性の高い復号処理装置が実現される。

【図面の簡単な説明】

【００３３】

【図１】共通鍵ブロック暗号アルゴリズムの基本構成を示す図である。

【図２】図１に示す共通鍵ブロック暗号処理部Ｅ１０の内部構成について説明する図である。

【図３】図２に示す暗号処理部１２の詳細構成について説明する図である。

【図４】ラウンド関数実行部の一構成例としてのＳＰＮ構造ラウンド関数について説明する図である。
【図５】ラウンド関数実行部の一構成例としてのＦｅｉｓｔｅｌ（フェイステル）構造について説明する図である。
【図６】ラウンド関数実行部の一構成例としての拡張Ｆｅｉｓｔｅｌ構造について説明する図である。
【図７】非線形変換処理部の具体例について説明する図である。
【図８】線形変換処理部の具体例について説明する図である。
【図９】Ｆｅｉｓｔｅｌ構造もしくは拡張Ｆｅｉｓｔｅｌ構造の一般的構成例について説明する図である。
【図１０】Ｆｅｉｓｔｅｌ構造もしくは拡張Ｆｅｉｓｔｅｌ構造において、異なるＳ－ｂｏｘを配置した構成例について説明する図である。
【図１１】異なるＳ－ｂｏｘを配置することにより飽和攻撃に対する耐性の向上を実現した構成例について説明する図である。
【図１２】異なるＳ－ｂｏｘを配置することにより飽和攻撃に対する耐性の向上を実現した構成例について説明する図である。
【図１３】異なるＳ－ｂｏｘを配置することにより飽和攻撃に対する耐性の向上を実現した構成例について説明する図である。
【図１４】異なるタイプのＳ－ｂｏｘを配置することにより代数的攻撃（ＸＳＬ攻撃）に対する耐性の向上を実現した構成例について説明する図である。
【図１５】異なるタイプのＳ－ｂｏｘを配置することにより代数的攻撃（ＸＳＬ攻撃）に対する耐性の向上を実現した構成例について説明する図である。
【図１６】異なるタイプのＳ－ｂｏｘを配置することにより代数的攻撃（ＸＳＬ攻撃）に対する耐性の向上を実現した構成例について説明する図である。
【図１７】異なるタイプのＳ－ｂｏｘを配置することにより代数的攻撃（ＸＳＬ攻撃）に対する耐性の向上を実現した構成例について説明する図である。
【図１８】異なるタイプのＳ－ｂｏｘを配置することにより代数的攻撃（ＸＳＬ攻撃）に対する耐性の向上を実現した構成例について説明する図である。
【図１９】本発明に係る暗号処理を実行する暗号処理装置としてのＩＣモジュールの構成例を示す図である。
【発明を実施するための形態】
【００３４】
以下、本発明の復号処理装置、および復号処理方法、並びにコンピュータ・プログラムの詳細について説明する。説明は、以下の項目に従って行なう。
１．共通鍵ブロック暗号の概要
２．複数の異なるＳボックスの配置により耐性を向上させた構成
（２Ａ）Ｓ－ｂｏｘを用いたＦｅｉｓｔｅｌもしくは拡張Ｆｅｉｓｔｅｌ型暗号において２種類以上の異なるＳ－ｂｏｘを配置することで飽和攻撃に対する耐性を向上させた構成
（２Ｂ）Ｓ－ｂｏｘを用いたブロック暗号において、２種類以上の異なるＳ－ｂｏｘを混在させることで代数的攻撃（ＸＳＬ攻撃）に対する耐性を向上させた構成
（２Ｃ）Ｓ－ｂｏｘを用いたＦｅｉｓｔｅｌ暗号もしくは拡張Ｆｅｉｓｔｅｌ型暗号において上記（２Ａ），（２Ｂ）を同時に実現する構成
３．暗号処理装置の構成例
【００３５】
［１．共通鍵ブロック暗号の概要］
まず、本発明の適用可能な共通鍵ブロック暗号の概要について説明する。本明細書において、共通鍵ブロック暗号（以下ではブロック暗号）は、以下に定義するものを指すものとする。
【００３６】

ブロック暗号は平文Ｐと鍵Ｋを入力し、暗号文Ｃを出力する。平文と暗号文のビット長をブロックサイズと呼びここではｎで示す。ｎは任意の整数値を取りうるが、通常、ブロック暗号アルゴリズムごとに、予め１つに決められている値である。ブロック長がｎのブロック暗号のことをｎビットブロック暗号と呼ぶこともある。

【００３７】

鍵のビット長は、ｋで表す。鍵は任意の整数値を取りうる。共通鍵ブロック暗号アルゴリズムは１つまたは複数の鍵サイズに対応することになる。例えば、あるブロック暗号アルゴリズムＡはブロックサイズｎ＝１２８であり、ビット長ｋ＝１２８、またはｋ＝１９２またはｋ＝２５６の各種の鍵サイズに対応するという構成もありうる。

平文［Ｐ］、暗号文［Ｃ］、鍵［Ｋ］の各ビットサイズは、以下のように示される。

平文Ｐ：ｎビット

暗号文Ｃ：ｎビット

鍵Ｋ：ｋビット

【００３８】

図１にｋビットの鍵長に対応したｎビット共通鍵ブロック暗号アルゴリズムＥの図を示す。図１に示すように、共通鍵ブロック暗号処理部Ｅ１０は、ｎビットの平文Ｐと、ｋビットの鍵Ｋを入力して、予め定められた暗号アルゴリズムを実行して、ｎビットの暗号文Ｃを出力する。なお、図１には平文から暗号文を生成する暗号化処理のみを示している．暗号文から平文を生成する復号処理は，一般的にはＥ１０の逆関数が用いられる．ただし，暗号化処理部Ｅ１０の構造によっては、復号処理においても、同様の共通鍵ブロック暗号処理部Ｅ１０が適用でき，鍵の入力順などのシーケンスの変更によって復号処理が可能となる。

【００３９】

図１に示す共通鍵ブロック暗号処理部Ｅ１０の内部構成について、図２を参照して説明する。ブロック暗号は２つの部分に分けて考えることができる。ひとつは鍵Ｋを入力とし、ある定められたステップにより入力鍵Ｋのビット長を拡大して拡大鍵Ｋ'（ビット長ｋ'）を出力する鍵スケジューリング部１１と、平文Ｐと鍵スケジューリング部１１から入力する拡大鍵Ｋ'を受け取り、平文Ｐを入力して、拡大鍵Ｋ'を適用した暗号処理を実行して、暗号文Ｃを生成するためのデータの変換を実行する暗号処理部１２である。なお，先に説明したように、暗号処理部１２の構造によっては，暗号文を平文に戻すデータ復号処理にも暗号処理部１２が適用可能な場合もある。

【００４０】

次に、図２に示す暗号処理部１２の詳細構成について図３を参照して説明する。暗号処理部１２は、図３に示すように、ラウンド関数実行部２０を適用したデータ変換を繰り返し実行する構成を持つ。すなわち、暗号処理部１２は、ラウンド関数実行部２０という処理単位に分割できる。ラウンド関数実行部２０は入力として、前段のラウンド関数実行部の出力X_iと、拡大鍵に基づいて生成されるラウンド鍵PK_iの２つのデータを受け取り、内部でデータ変換処理を実行して出力データX_{i+1}を次のラウンド関数実行部に出力する。なお、第１ラウンドでは、入力は、平文または平文に対する初期化処理データである。また最終ラウンドの出力は暗号文となる。

【００４１】

図３示す例では、暗号処理部１２は、ｒ個のラウンド関数実行部２０を有し、ｒ回のラウンド関数実行部におけるデータ変換を繰り返して暗号文を生成する構成となっている。ラウンド関数の繰り返し回数をラウンド数と呼ぶ。図に示す例では、ラウンド数はｒとなる。

【００４２】

各ラウンド関数実行部の入力データX_iは暗号化途中のｎビットデータであり、あるラウンドにおけるラウンド関数の出力X_{i+1}が次のラウンドの入力として供給される。各ラウンド関数実行部のもう一つの入力データは鍵スケジュールから出力された拡大鍵のＫ'に基づくデータが用いられる。この各ラウンド関数実行部に入力され、ラウンド関数の

実行に適用される鍵をラウンド鍵と呼ぶ。図で、ｉラウンドに適用するラウンド鍵をＲＫ$_i$として示している。拡大鍵Ｋ'は、例えば、ｒラウンド分のラウンド鍵ＲＫ$_1$～ＲＫ$_r$の連結データとして構成される。

【００４３】

図３に示す構成は、暗号処理部１２の入力側から見て１ラウンド目の入力データをＸ$_0$とし、ｉ番目のラウンド関数から出力されるデータをＸ$_i$、ラウンド鍵をＲＫ$_i$として示した暗号処理部１２の構成である。なお，この暗号処理部１２の構造によっては，例えば，適用するラウンド鍵の適用シーケンスを，暗号化処理と逆に設定し，暗号文を暗号処理部１２に入力することで平文を出力する構成にできる。

【００４４】

図３に示す暗号処理部１２のラウンド関数実行部２０は、さまざまな形態をとりうる。ラウンド関数はその暗号アルゴリズムが採用する構造（ｓｔｒｕｃｔｕｒｅ）によって分類できる。代表的な構造として、

（ア）ＳＰＮ（Substitution Permutation Network）構造、

（イ）Ｆｅｉｓｔｅｌ構造、

（ウ）拡張Ｆｅｉｓｔｅｌ構造、

がある。以下、これらの具体的構成について、図４～図６を参照して説明する。

【００４５】

（ア）ＳＰＮ構造ラウンド関数

まず、図４を参照して、ラウンド関数実行部２０の一構成例としてのＳＰＮ構造ラウンド関数について説明する。ＳＰＮ構造ラウンド関数実行部２０ａは、非線形変換層（Ｓ層）と線形変換層（Ｐ層）を接続したいわゆるＳＰ型の構成を有する。図４に示すように、ｎビットの入力データすべてに対して、ラウンド鍵との排他的論理和（ＥＸＯＲ）演算を実行する排他的論理和演算部２１、排他的論理和演算部２１の演算結果を入力し、入力データの非線形変換を実行する非線形変換処理部２２、非線形変換処理部２２における非線形変換処理結果を入力し、入力データに対する線形変換処理を実行する線形変換処理部２３などによって構成される。線形変換処理部２３の線形変換処理結果が、次のラウンドに出力される。最終ラウンドでは暗号文となる。なお、図４に示す例では、排他的論理和演算部２１、非線形変換処理部２２、線形変換処理部２３の処理順を示しているが、これらの処理部の順番は、限定されるものではなく、他のシーケンスで処理を行なう構成としてもよい。

【００４６】

（イ）Ｆｅｉｓｔｅｌ構造

次に、図５を参照してラウンド関数実行部２０の一構成例としてのＦｅｉｓｔｅｌ（フェイステル）構造について説明する。Ｆｅｉｓｔｅｌ構造は、図５に示すように、前ラウンドからの入力（第１ラウンドでは入力文）であるｎビットの入力データをｎ／２ビットの２つのデータに分割して、各ラウンドにおいて入れ替えながら処理を実行する。

【００４７】

Ｆｅｉｓｔｅｌ構造を持つラウンド関数実行部２０ｂを適用した処理においては、図に示すように、一方のｎ／２ビットデータとラウンド鍵とがＦ関数部３０に入力される。Ｆ関数部３０は、上述したＳＰＮ構造と同様、非線形変換層（Ｓ層）と線形変換層（Ｐ層）を接続したいわゆるＳＰ型の構成を有する。

【００４８】

前ラウンドからのｎ／２ビットデータとラウンド鍵とがＦ関数部３０の排他的論理和演算部３１に入力され排他的論理和（ＥＸＯＲ）処理がなされる。さらに、この結果データを非線形変換処理部３２に入力して非線形変換を実行し、さらに、この非線形変換結果が線形変換処理部３３に入力され線形変換が実行される。この線形変換結果が、Ｆ関数処理結果データとして出力する。

【００４９】

さらに、このＦ関数出力と、前ラウンドから入力するもう１つのｎ／２ビット入力とを

、排他的論理和演算部３４に入力し、排他的論理和演算（ＥＸＯＲ）を実行して、実行結果を、次のラウンドにおけるＦ関数の入力として設定される。なお、図に示す第ｉラウンドにおけるＦ関数入力に設定されたｎ／２ビットは次のラウンドのＦ関数出力との排他的論理和演算に適用される。このように、Ｆｅｉｓｔｅｌ構造は、各ラウンドにおいて入力を交互に入れ替えながらＦ関数を適用したデータ変換処理を実行する。

【００５０】

（ウ）拡張Ｆｅｉｓｔｅｌ構造

次に、図６を参照してラウンド関数実行部２０の一構成例としての拡張Ｆｅｉｓｔｅｌ構造について説明する。先に、図５を参照して説明したＦｅｉｓｔｅｌ構造は、ｎビットの平文を２つに分割してｎ／２ビットずつに区分して処理を実行していた。すなわち、分割数：ｄ＝２とした処理である。なお、この分割数は、データ系列数とも呼ばれる。

【００５１】

拡張Ｆｅｉｓｔｅｌ構造では、このデータ系列数（分割数）ｄを２以上の任意の整数とした設定である。データ系列数（分割数）ｄの値に応じたさまざまな拡張Ｆｅｉｓｔｅｌ構造を定義することができる。図６に示す例では、データ系列数（分割数）ｄ＝４であり、各系列に対してｎ／４ビットのデータが入力される。各ラウンドでは、１つ以上のラウンド関数としてのＦ関数が実行される。図に示す例は、１ラウンドにおいて２つのＦ関数部によるラウンド演算を行なう構成例である。

【００５２】

Ｆ関数部４１，４２の構成は、先に図５を参照して説明したＦ関数部３０の構成と同様であり、ラウンド鍵と入力値との排他的論理和演算と、非線形変換処理、線形変換処理を実行する構成となっている。なお、各Ｆ関数部に入力するラウンド鍵は、入力ビットとビット数が一致するように調整される。図に示す例では、各Ｆ関数部４１，４２に入力するラウンド鍵は、ｎ／４ビットとなる。これらの鍵は、拡大鍵を構成するラウント鍵をさらにビット分割して生成する。なお、データ系列数（分割数）をｄとしたとき、各系列に入力されるデータはｎ／ｄビットであり、各Ｆ関数に入力する鍵のビット数もｎ／ｄビットに調整される。

【００５３】

なお、図６に示す拡張Ｆｅｉｓｔｅｌ構造では、データ系列数（分割数）をｄとして、各ラウンドにおいてｄ／２のＦ関数を並列に実行する構成例であるが、拡張Ｆｅｉｓｔｅｌ構造は、各ラウンドにおいて、１つ以上ｄ／２以下のＦ関数を実行する構成が可能である。

【００５４】

図４～図６を参照して説明したように、共通鍵ブロック暗号における暗号処理部１２のラウンド関数実行部２０は、

（ア）ＳＰＮ（Substitution Permutation Network）構造、

（イ）Ｆｅｉｓｔｅｌ構造、

（ウ）拡張Ｆｅｉｓｔｅｌ構造、

これらの構造をとり得る。これらのラウンド関数実行部は、いずれも非線形変換層（Ｓ層）と線形変換層（Ｐ層）を接続したいわゆるＳＰ型の構成を有する。すなわち、非線形変換処理を実行する非線形変換処理部と、線形変換処理を実行する線形変換処理部とを有する。以下、これらの変換処理構成について説明する。

【００５５】

（非線形変換処理部）

非線形変換処理部の具体例について、図７を参照して説明する。図７に示すように、非線形変換処理部５０は、具体的には、Ｓボックス（Ｓ－ｂｏｘ）５１と呼ばれるｓビット入力ｓビット出力の非線形変換テーブルがｍ個並んだものであり、ｍｓビットの入力データがｓビットずつ分割されてそれぞれ対応するＳボックス（Ｓ－ｂｏｘ）５１に入力されデータが変換される。各Ｓボックス５１では、例えば変換テーブルを適用した非線形変換処理が実行される。

【００５６】
入力されるデータのサイズが大きくなると実装上のコストが高くなる傾向がある。それを回避するために、図７に示すように、処理対象データＸを複数の単位に分割し、それぞれに対して、非線形変換を施す構成がとられることが多い。例えば入力サイズをｍｓビットとしたとき、ｓビットずつのｍ個のデータに分割して、ｍ個のＳボックス（Ｓ－ｂｏｘ）５１それぞれに対してｓビットを入力し、例えば変換テーブルを適用した非線形変換処理を実行して、これらの各Ｓビット出力ｍ個を合成してｍｓビットの非線形変換結果を得る。
【００５７】
（線形変換処理部）
線形変換処理部の具体例について、図８を参照して説明する。線形変換処理部は、入力値、例えば、Ｓボックスからの出力データであるｍｓビットの出力値を入力値Ｘとして入力し、この入力に対して線形変換を施しｍｓビットの結果を出力する。線形変換処理は、例えば、入力ビット位置の入れ替え処理などの線形変換処理を実行して、ｍｓビットの出力値Ｙを出力する。線形変換処理は、例えば、入力に対して、線形変換行列を適用して入力ビット位置の入れ替え処理を行なう。この行列の一例が図８に示す線形変換行列である。
【００５８】
線形変換処理部において適用する線形変換行列の要素は拡大体：ＧＦ（2^8）の体の要素やＧＦ（２）の要素など、一般的にはさまざまな表現を適用した行列として構成できる。図８は、ｍｓビット入出力をもち、ＧＦ（2^s）の上で定義されるｍ×ｍの行列により定義される線形変換処理部の１つの構成例を示すものである。
【００５９】
［２．複数の異なるＳボックスの配置により耐性を向上させた構成］
上述したように、共通鍵ブロック暗号は、ラウンド関数の繰り返しによる暗号処理を行なう構成である。この共通鍵ブロック暗号処理の問題点として、暗号解析による鍵の漏洩がある。暗号解析による鍵の解析が容易であるということは、その暗号処理の安全性が低いということになり、実用上、大きな問題となる。以下では、複数の異なるＳ－ｂｏｘ（Ｓボックス）を配置することで耐性を向上させた暗号処理構成について説明する。
【００６０】
図７を参照して説明したように、ラウンド関数実行部に含まれる非線形変換処理部は、非線形変換処理を実行する複数のＳ－ｂｏｘ（Ｓボックス）によって構成される。従来、これらのＳ－ｂｏｘはすべて共通の非線形変換処理用のテーブルを適用し、各Ｓボックスにおいて共通の非線形変換処理を行なう構成となっていた。
【００６１】
本発明では、このＳ－ｂｏｘの共通性に起因する脆弱性、すなわち、鍵解析などの暗号解析である攻撃に対する弱さに着目し、複数の異なるＳ－ｂｏｘの配置により耐性を向上させた構成を提案する。
【００６２】
以下、本発明の実施例として、以下の３つの実施例について、順次、説明する。
（２Ａ）Ｓ－ｂｏｘを用いたＦｅｉｓｔｅｌもしくは拡張Ｆｅｉｓｔｅｌ型暗号において２種類以上の異なるＳ－ｂｏｘを配置することで飽和攻撃に対する耐性を向上させた構成
（２Ｂ）Ｓ－ｂｏｘを用いたブロック暗号において、２種類以上の異なるＳ－ｂｏｘを混在させることで代数的攻撃（ＸＳＬ攻撃）に対する耐性を向上させた構成
（２Ｃ）Ｓ－ｂｏｘを用いたＦｅｉｓｔｅｌ暗号もしくは拡張Ｆｅｉｓｔｅｌ型暗号において上記（２Ａ），（２Ｂ）を同時に実現する構成
【００６３】
（２Ａ）Ｓ－ｂｏｘを用いたＦｅｉｓｔｅｌもしくは拡張Ｆｅｉｓｔｅｌ型暗号において２種類以上の異なるＳ－ｂｏｘを配置することで飽和攻撃に対する耐性を向上させた構

成

【００６４】

まず、Ｓ－ｂｏｘを用いたＦｅｉｓｔｅｌもしくは拡張Ｆｅｉｓｔｅｌ型暗号において２種類以上の異なるＳ－ｂｏｘを配置することで飽和攻撃に対する耐性を向上させた構成について説明する。

【００６５】

（２Ａ－１．飽和攻撃の概要）

まず、最初にブロック暗号に対する攻撃として知られる飽和攻撃について説明する。飽和攻撃には複数のタイプがある。第１のタイプは、平文の特定のバイト位置について、２５６種類の値を変化させながら入力した場合、数ラウンド分のラウンド変換処理後に、出力値の特定のバイト位置に２５６種類の値がすべて現れるという性質を利用した攻撃法である。

また、飽和攻撃の別の形態の攻撃としては、数ラウンド分のラウンド変換後の特定のバイト位置に表れる値の総和を計算すると必ず０になるという性質を利用した攻撃手法である。

【００６６】

例えば、ラウンド関数を実行する共通鍵ブロック暗号処理装置に対して入力する２５６種類の平文P_0～P_{255}として、

P_0＝（０，０，０，０，０，０，０，０）

P_1＝（０，０，０，０，０，０，０，１）

：　　：

P_{255}＝（０，０，０，０，０，０，０，２５５）

これらの平文P_0～P_{255}を、順次入力する。なお、上記表記において、各［０］は１バイトデータの０を示す。

【００６７】

これらの平文P_0～P_{255}を、順次入力した場合に、ある特定のラウンド分のデータ変換処理が終了した後の出力値を以下のようにC_0～C_{255}とする。

C_0＝（c_0，？，？，？，？，？，？，？）

C_1＝（c_1，？，？，？，？，？，？，？）

：　　：

C_{255}＝（c_{255}，？，？，？，？，？，？，？）

上記出力中［？］は、いかなるビット値でも構わない。

【００６８】

これらの出力C_0～C_{255}には、上記のように、特定のバイト位置（上記例では最初のバイト位置）に２５６種類の値c_0～c_{255}がすべて現れるという性質がある。このように、０から２５５までの値が、順番は問わずに、それぞれ一回ずつ必ず現れることが予め分かっている場合、その性質を利用した攻撃をすることが可能である。入力値を、順次変更して出力値を解析することで、ラウンド鍵の推定を行なうことが可能であることが知られている。

【００６９】

さらに出力C_0～C_{255}に含まれる特定のバイト位置の値：c_0～c_{255}を総和（ＥＸＯＲ）した場合に０となる場合、この性質を利用した攻撃（暗号解析）が可能となる。このように、２５６種類の平文P_0～P_{255}を順次、入力して、特定バイト位置の出力を解析することで、鍵の推定が可能となる。

【００７０】

飽和攻撃は、このようにラウンド関数部の変換結果に、上記のような特定の規則をもたらす出力、すなわち、

２５６種類の値c_0～c_{255}がすべて出現する。あるいは、

特定のバイト位置の値：c_0～c_{255}を総和（ＥＸＯＲ）した場合に０となる。

このような規則性のある出力が発生する場合に、これらの規則性に基づいて実行する攻

撃（解析）手法である。

【００７１】

従って、暗号の設計の段階で、このような特異な出力をラウンド関数部の出力として発生させない構成とすることが、飽和攻撃に対して強い暗号とするために有効である。なお、この飽和攻撃は、バイト単位（８ビット）のみの解析に限らず、任意のビット長に対して同様の性質を利用した攻撃が可能となる。

【００７２】

（２Ａ－２）Ｆｅｉｓｔｅｌ構造、または拡張Ｆｅｉｓｔｅｌ構造を適用した暗号処理における問題点

次に、Ｆｅｉｓｔｅｌ構造、または拡張Ｆｅｉｓｔｅｌ構造を適用した暗号処理における問題点について検討する。

【００７３】

Ｆｅｉｓｔｅｌ構造、または拡張Ｆｅｉｓｔｅｌ構造については、先に、図５、図６を参照して説明したように、いずれもＳＰ型の非線形変換処理部と、線形変換処理部を有するＦ関数部を適用したラウンド演算を繰り返し実行する構成である。Ｆｅｉｓｔｅｌ構造は、データ系列数（分割数）が２に限定されるが、拡張Ｆｅｉｓｔｅｌ構造では、データ系列数（分割数）が２以上の任意の数に設定される点が異なる。

【００７４】

以下では、Ｆｅｉｓｔｅｌ構造、または拡張Ｆｅｉｓｔｅｌ構造を適用した暗号処理におけるラウンド関数の実行部であるＦ関数内の非線形変換処理部にＳ－ｂｏｘを利用した構成を想定する。Ｓ－ｂｏｘは、先に図７を参照して説明したように、非線形変換処理部に入力されるｍｓビットデータをｍ分割したｓビットデータそれぞれについて、例えば非線形変換用テーブルを適用した非線形変換処理を実行する。

【００７５】

先に、説明したように、旧来のブロック暗号におけるラウンド関数の実行に適用するＦ関数は、各ラウンドにおいて繰り返し同じものが利用される。このような同じＦ関数を各ラウンドに設定したＦｅｉｓｔｅｌ構造、または拡張Ｆｅｉｓｔｅｌ構造では、前述した飽和攻撃にさらされ易くなる。この理由について、図９を参照して説明する。

【００７６】

図９は、Ｆｅｉｓｔｅｌ構造もしくは拡張Ｆｅｉｓｔｅｌ構造の一部分を切り出した構成を示す図である。すなわち、図９には、Ｆｅｉｓｔｅｌ構造もしくは拡張Ｆｅｉｓｔｅｌ構造を持つ暗号構成に含まれる２つのラウンド関数実行部、すなわち、Ｆ関数１０１，１０２を示している。この２つのＦ関数１０１，１０２は入力データの系列（ｘ）と出力データの系列（ｙ）が同一であり、かつ上下に隣り合う位置にあるＦ関数である。

【００７７】

２つのＦ関数１０１，１０２は、ラウンド鍵との排他的論理和演算部、非線形変換処理部、線形変換処理部によって構成されている。本処理例では、Ｆ関数１０１，１０２は、３２ビット入出力処理を行う構成であり、非線形変換処理部は、４つのＳ－ｂｏｘによって構成され、それぞれのＳ－ｂｏｘは、８ビット入出力を行なう。

【００７８】

図９に示すＡ～Ｊは、様々なデータを示す。すなわち、

Ａ：先行Ｆ関数１０１に対する入力

Ｂ：先行Ｆ関数１０１の出力

Ｃ：後続Ｆ関数１０２の入力

Ｄ：後続Ｆ関数１０２の出力

Ｅ：先行Ｆ関数１０１の出力Ｂに対する排他的論理和演算データ

Ｆ：データＡに対する排他的論理和演算データ

Ｇ：データＢとデータＥとの排他的論理和演算結果

Ｈ：データＤとデータＧとの排他的論理和演算結果

Ｉ：先行Ｆ関数１０１に対して入力するラウンド鍵

Ｊ：後続Ｆ関数１０２に対して入力するラウンド鍵
これらの各データを示している。
【００７９】
以降の説明で、各Ｆ関数１０１，１０２において処理される３２ビットデータをバイト単位（８ビット）で分割して示す場合、例えばデータＡが３２ビットデータである場合、各１バイト（８ビット）データＡ［０］，Ａ［１］，Ａ［２］，Ａ［３］の連結データとして、
Ａ＝Ａ［０］｜Ａ［１］｜Ａ［２］｜Ａ［３］
上記のように表現する。
【００８０】
ここで、図９に示す暗号化処理構成に対して入力する平文として２５６種類のデータ、例えば、
P_0＝（０，０，０，０）
P_1＝（１，０，０，０）
：　　：
P_{255}＝（２５５，０，０，０）
これらの平文P_0～P_{255}を、順次入力すると仮定する。なお、上記表記において、各［０］，［１］～［２５５］は１バイトデータを示す。
【００８１】
この入力値を、図９に示す先行Ｆ関数１０１に対する入力データＡとする。データＡは上記のように、２５６種類のデータを観察したときに第１番目のバイトＡ［０］に０から２５５までの２５６種類の全ての値が現れ、それ以外のバイト位置はつねに同じ値で固定されていたものとする。（これは飽和攻撃を試みる攻撃者が平文入力を制御してこのような状況を作り出すことがあるために想定する。）
【００８２】
さらにデータＡに対する排他的論理和演算データＦの値が、上記２５６種類のデータＡの順次入力処理の間、常に固定であったと仮定すると、後続Ｆ関数１０２の入力データＣの第一番目のバイトＣ［０］もまた０から２５５までの２５６種類の全ての値が現れ、それ以外のバイト位置はつねに同じ値で固定されることが保証される。
【００８３】
このとき、
Ｉ：先行Ｆ関数１０１に対して入力するラウンド鍵
Ｊ：後続Ｆ関数１０２に対して入力するラウンド鍵
および、
Ｆ：データＡに対する排他的論理和演算データ
これらの各データの値の組み合わせによっては、常に以下の式が成立することが発生し得る。すなわち、
Ａ［０］（ＥＸＯＲ）Ｉ［０］＝Ｃ［０］（ＥＸＯＲ）Ｊ［０］
上記式が成立することがありうる。
なお、（ＥＸＯＲ）は排他的論理和演算を示し、
Ａ［０］（ＥＸＯＲ）Ｉ［０］は、データＡ［０］とデータＩ［０］との排他的論理和演算、
Ｃ［０］（ＥＸＯＲ）Ｊ［０］は、データＣ［０］とデータＪ［０］との排他的論理和演算、
を示している。
【００８４】
式：Ａ［０］（ＥＸＯＲ）Ｉ［０］＝Ｃ［０］（ＥＸＯＲ）Ｊ［０］
この式が意味することは２つのＦ関数１０１，１０２における２つのＳ－ｂｏｘに入力される値がつねに同じ値となることに他ならない。これらＳ－ｂｏｘはすべて同じ非線形変換処理を実行しており、同じ入力値に対しては同じ出力値を出力する。従って、２つの

Ｆ関数１０１，１０２における２つのＳ－ｂｏｘの出力が常に同じになる。各Ｆ関数１０１，１０２では、この同じＳ－ｂｏｘ出力が、線形変換処理部の行列により線形変換されて右側のデータ系列（ｙ）の排他的論理和演算部に出力される。図に示す排他的論理和演算部１１１，１１２である。

【００８５】

この２つのＦ関数１０１，１０２から排他的論理和演算部１１１，１１２に出力される値Ｂ，Ｄは特定の差分値Δをもつ。すなわち、

Ｂ（ＥＸＯＲ）Δ＝Ｄ

となる。

このとき、排他的論理和演算部１１１では、

Ｇ＝Ｂ（ＥＸＯＲ）Ｅ

の演算によりデータＧが算出され、

排他的論理和演算部１１２では、

Ｈ＝Ｇ（ＥＸＯＲ）Ｄ

が実行される。

上記式、Ｈ＝Ｇ（ＥＸＯＲ）Ｄは、Ｇ＝Ｂ（ＥＸＯＲ）Ｅ、Ｂ（ＥＸＯＲ）Δ＝Ｄであるので、

Ｈ＝Ｂ（ＥＸＯＲ）Ｅ（ＥＸＯＲ）Ｂ（ＥＸＯＲ）Δ

＝Ｅ（ＥＸＯＲ）Δとなる。

【００８６】

すなわち、固定の差分値をもつ値同士の排他的論理和演算結果は、固定値Δとなるので、結果として、

Ｈ＝Ｂ（ＥＸＯＲ）Ｅ（ＥＸＯＲ）Ｂ（ＥＸＯＲ）Δ

＝Δ（ＥＸＯＲ）Ｅ

＝Ｅ（ＥＸＯＲ）Δ

となる。

【００８７】

すなわち、排他的論理和演算部１１２の出力Ｈがデータ Ｅに固定された値Δが排他的論理和されただけとなり、ラウンド関数（Ｆ関数）を２段分実行したのにもかかわらず、データの攪拌がされないという結果となる。この性質を利用すると、後続のラウンドのラウンド鍵を推定することが容易となる。すなわち、後続のラウンドが存在しても、そのラウンドを仮に設定した鍵を用いて復号してＨのデータまで復号したときに、この性質が現れているか否かをチェックすることにより、仮に用いた鍵が正しいものかそうでないものかを確率的に判断することが可能となる。つまりラウンド鍵の推定が可能となり、飽和攻撃による解析を可能にしてしまうことになる。

【００８８】

この対策として、各Ｆ関数の位置によって線形変換処理に適用する行列部分を変更することは可能であるが、各Ｆ関数のＳ－ｂｏｘが同一のものである場合には、上記と同様の条件が発生すれば、線形変換行列の要素の関係性によっては、後続のＦ関数１０２の出力Ｄがデータ Ｇと排他的論理和された時点で、一部のバイトで打ち消しあいが起こってしまい、攻撃者に有利な状況が発生することがある。

【００８９】

このように、少なくとも同一系列に対する出力を行なう複数のＦ関数において適用する非線形変換処理を同一構成とすることは、飽和攻撃による鍵推定を可能にする可能性がある。さらに、Ｓ－ｂｏｘによっては、その演算（ＥＸＯＲ）結果、すなわち、

Ｓ（Ａ［０］（ＥＸＯＲ）Ｉ［０］）（ＥＸＯＲ）Ｓ（Ｃ［０］（ＥＸＯＲ）Ｊ［０］）

これらの結果として、０から２５５までの２５６種類の全ての値が現れるという場合も望ましいとは言えない。本来ならばＡ［０］とＣ［０］の両者が２５６種類の異なる値をそれぞれ出力していてもそれらの演算（ＥＸＯＲ）結果が、かならずしも２５６種類の出

力値をすべて取ることは保証されないが、Ｓ－ｂｏｘによってはそのような状況が発生することはありえる。この予期しないことがらが発生すると攻撃に利用できる情報（値がすべて異なるという情報）が次の段まで保存されてしまうことになり、攻撃者にとって有利な状況を生み出してしまう。

【００９０】

（２Ａ－３）複数種類のＳ－ｂｏｘを利用することによる耐性向上方法

以下、飽和攻撃による鍵推定を困難化するための構成例について説明する。すなわち、上記のような条件が万が一揃ってしまっても、データの打消しによるラウンド関数の実行前のデータと実行後のデータとが等しいデータとならないように、各Ｆ関数の非線形変換処理部、すなわちＳ－ｂｏｘを構成する。

【００９１】

この具体的例について、図１０を参照して説明する。図１０に示す構成も、図９と同様、Ｆｅｉｓｔｅｌ構造もしくは拡張Ｆｅｉｓｔｅｌ構造の一部分を切り出した構成を示し、入力データの系列（ｘ）と出力データの系列（ｙ）が同一であり、かつ上下に隣り合う位置にあるＦ関数２０１，２０２を示している。

【００９２】

２つのＦ関数２０１，２０２は、ラウンド鍵との排他的論理和演算部、非線形変換処理部、線形変換処理部によって構成されている。Ｆ関数２０１，２０２は、３２ビット入出力処理を行う構成であり、非線形変換処理部は、４つのＳ－ｂｏｘによって構成され、それぞれのＳ－ｂｏｘは、８ビット入出力を行なう。

【００９３】

図１０に示すＡ～Ｊは、図９と同様、

Ａ：先行Ｆ関数２０１に対する入力

Ｂ：先行Ｆ関数２０１の出力

Ｃ：後続Ｆ関数２０２の入力

Ｄ：後続Ｆ関数２０２の出力

Ｅ：先行Ｆ関数２０１の出力Ｂに対する排他的論理和演算データ

Ｆ：データＡに対する排他的論理和演算データ

Ｇ：データＢとデータＥとの排他的論理和演算結果

Ｈ：データＤとデータＧとの排他的論理和演算結果

Ｉ：先行Ｆ関数２０１に対して入力するラウンド鍵

Ｊ：後続Ｆ関数２０２に対して入力するラウンド鍵

これらの各データを示している。

【００９４】

図１０に示す構成において、先行Ｆ関数２０１と後続Ｆ関数２０２の各々に設定されたる非線形変換処理部のＳ－ｂｏｘは、異なるＳ－ｂｏｘ［Ｓ１］，［Ｓ２］を利用した構成としてある。

【００９５】

すなわち、先行Ｆ関数２０１において非線形変換処理を実行するＳ－ｂｏｘ［Ｓ１］と、後続Ｆ関数２０２において非線形変換処理を実行するＳ－ｂｏｘ［Ｓ２］とは異なる非線形変換処理を実行する。具体的には、例えば異なる非線形変換テーブルを適用した非線形変換処理を実行し、同じ入力に対して同じ出力がなされるとは限らない。

【００９６】

ここで，各Ｓ－ｂｏｘ：Ｓ１，Ｓ２は以下の条件を満たす２つの異なるＳ－ｂｏｘであるとする。

各Ｓ－ｂｏｘ：Ｓ１，Ｓ２がｎビット入出力の非線形変換処理を実行するＳ－ｂｏｘであるとき、

（条件１）

任意のｓビットデータｃに対して、すべてのｓビットデータである2^{s}個のｘを、順次、入力した場合、

入力データ［ｘ］に対応する第１のＳ－ｂｏｘ［Ｓ１］の出力Ｓ１（ｘ）と、
入力データ［ｘ（ＥＸＯＲ）ｃ］に対応するＳ－ｂｏｘ［Ｓ２］の出力Ｓ２（ｘ（ＥＸＯＲ）ｃ）、
は、最低でもひとつは異なる値を持つ。すなわち、
Ｓ１（ｘ）（ＥＸＯＲ）Ｓ２（ｘ（ＥＸＯＲ）ｃ）
上記式は固定値にならない。
さらに、
（条件２）
任意のｓビットデータｃに対して、すべてのｓビットデータである２^{s}個のｘを、順次、入力した場合、
入力データ［ｘ］に対応する第１のＳ－ｂｏｘ［Ｓ１］の出力Ｓ１（ｘ）と、
入力データ［ｘ（ＥＸＯＲ）ｃ］に対応するＳ－ｂｏｘ［Ｓ２］の出力Ｓ２（ｘ（ＥＸＯＲ）ｃ）、
は、最低でもひとつは重複値を持つ。すなわち、
Ｓ１（ｘ）（ＥＸＯＲ）Ｓ２（ｘ（ＥＸＯＲ）ｃ）
上記式は、２^{s}のすべてが一度ずつ表れる形にならない。

【００９７】

これは、図１０において、
データＡを［ｘ］
データＦを［ｃ］
と仮定した場合、
先行Ｆ関数２０１のＳ－ｂｏｘ［Ｓ１］の出力Ｓ１（ｘ）と、
後続Ｆ関数２０２のＳ－ｂｏｘ［Ｓ２］の出力Ｓ２（ｘ（ＥＸＯＲ）ｃ）と、
これらの２つの出力が全く同一であったり、排他的論理和した結果がすべてことなる値にはならないという条件を示すものである。

【００９８】

この条件を満たす２つのＳ－ｂｏｘ［Ｓ１］，［Ｓ２］を、図１０に示すように設定する。
すなわち、あるＦ関数ではＳ－ｂｏｘ［Ｓ１］のみを用いた非線形変換処理部として、次のＦ関数ではＳ－ｂｏｘ［Ｓ２］のみを用いた非線形変換処理部とする。それ以降のラウンドがある場合は、同様に、各Ｆ関数の非線形変換処理部にＳ－ｂｏｘ［Ｓ１］，［Ｓ２］の順に設定する。

【００９９】

このように、入力データの系列と出力データの系列が同一であり、かつ上下に隣り合う位置にあるＦ関数における非線形変換処理を異なる非線形変換処理を実行する構成、すなわち、異なるＳ－ｂｏｘの配置を行なうことで、出力系列に表れるデータが、ラウンド関数実行前の同一出力系列に出現するデータと強い相関をもつ可能性を著しく低下させることができる。

【０１００】

すなわち、上述した（条件１）を満たすＳ－ｂｏｘを利用することにより、たとえ２つのＳ－ｂｏｘの入力の関係が、固定値の差を持つ場合でも、その出力の排他的論理和は最低でも一回は異なる値を持つため、完全に打ち消されることはないことが保証できる。
また、上述した（条件２）を満たすＳ－ｂｏｘを利用することにより、たとえ２つのＳ－ｂｏｘの入力の関係が、固定値の差を持つ場合でも、その出力の排他的論理和は最低でも一回は重複する値を持つため、攻撃に利用可能な性質を損なわれるといえる。従って、２つのＳ－ｂｏｘを以上のように配置することで飽和攻撃の攻撃者に対する有利な条件が減少するため、攻撃に対する耐性が向上できることが期待できる。

【０１０１】

すなわち、図１０において、各Ｆ関数２０１，２０２におけるＳ－ｂｏｘの入力値が等しい場合、

Ａ［０］（ＥＸＯＲ）Ｉ［０］＝Ｃ［０］（ＥＸＯＲ）Ｊ［０］
となったとしても、
各Ｆ関数のＳ－ｂｏｘの出力値、すなわち、
Ｓ１（Ａ［０］（ＥＸＯＲ）Ｉ［０］）と、
Ｓ２（Ｃ［０］（ＥＸＯＲ）Ｊ［０］）と、
これらの出力値が、すべての場合において完全に一致することはなく、結果として、各Ｆ関数２０１，２０２のＦ関数出力Ｂ，Ｄは完全に一致することがなく、先に図９を参照して説明したような事象、すなわち、
Ｅ＝Ｈ（ＥＸＯＲ）Δ
といった、１つのデータ系列において、ラウンド関数（Ｆ関数）の実行前後のデータが固定値のみの差分をもつという可能性をなくすことができる。

【０１０２】
このように、入力データの系列と出力データの系列が同一であり、かつ上下に隣り合う位置にあるＦ関数における非線形変換処理を異なる非線形変換処理を実行する異なるＳ－ｂｏｘとすることで、飽和攻撃の難易度を大きく向上させ、攻撃に対する耐性が高めることが可能となる。

【０１０３】
（発展系－１）
図１０を参照して説明した上記構成は、２つのＦ関数の関係にのみ着目し、この２つのＦ関数に設定するＳ－ｂｏｘを異なるものとするという条件を導いたが、３つ以上のＦ関数においても同様のことがいえる。例えば、図１１に示すようにＦ関数ごとに異なるＳ－ｂｏｘを配置することにより飽和攻撃に対する耐性の向上が期待できる。

【０１０４】
図１１は、Ｆｅｉｓｔｅｌ構造もしくは拡張Ｆｅｉｓｔｅｌ構造の一部分を切り出した構成を示し、入力データの系列（ｘ）と出力データの系列（ｙ）が同一であり、かつ上下に隣り合う位置にある３つのＦ関数２１１～２１３を示している。
Ｆ関数２１１の非線形変換処理部にはＳ－ｂｏｘ［Ｓ１］が設定され、
Ｆ関数２１２の非線形変換処理部にはＳ－ｂｏｘ［Ｓ２］が設定され、
Ｆ関数２１３の非線形変換処理部にはＳ－ｂｏｘ［Ｓ３］が設定されており、
Ｓ１≠Ｓ２≠Ｓ３
である。

【０１０５】
このように、複数のＳ－ｂｏｘに求められる条件としては、
（条件１）
ｋ個（ｋ＞２）のＳ－ｂｏｘからなる集合Ｓ１，Ｓ２，．．，Ｓｋに対して、互いに相異なる２つのＳ－ｂｏｘの組ＳｉとＳｊ（ｉ≠ｊ）に対して、任意のｃに対して、すべての可能な2^{s}個のｘを入力として与えたとき、
Ｓｉ（ｘ）と、
Ｓｊ（ｘ（ＥＸＯＲ）ｃ）、
これらのＳボックス出力は、完全同一にならず、最低でもひとつは異なる値を出力する。すなわち、
Ｓｉ（ｘ）と、Ｓｊ（ｘ（ＥＸＯＲ）ｃ）の排他的論理和の結果は固定値にならない。
さらに、
（条件２）
ｋ個（ｋ＞２）のＳ－ｂｏｘからなる集合Ｓ１，Ｓ２，．．，Ｓｋに対して、互いに相異なる２つのＳ－ｂｏｘの組ＳｉとＳｊ（ｉ≠ｊ）に対して、任意のｃに対して、すべての可能な2^{n}個のｘを入力として与えたとき、
Ｓｉ（ｘ）と、
Ｓｊ（ｘ（ＥＸＯＲ）ｃ）、
これらのＳボックス出力は、2^{n}個のすべての値が一度ずつ現れる形にならない。すな

わち、最低でも一つの重複値が現れる。

【０１０６】

この条件を満足するＳ－ｂｏｘからなる集合Ｓ１，Ｓ２，．．，Ｓｋを設定し、入力データの系列（ｘ）と出力データの系列（ｙ）が同一であり、かつ上下にシーケンシャルに並ぶ位置にある複数のＦ関数に、これらのＦ関数を配列することで、出力系列に表れるデータが、ラウンド関数実行前の同一出力系列に出現するデータと一致する可能性を著しく低下させることができ、結果として、飽和攻撃の難易度を大きく向上させ、攻撃に対する耐性が高めることが可能となる。

【０１０７】

（発展系－２）

現実的な実装を考慮すると、各Ｆ関数に含まれるＳ－ｂｏｘの種類が複数になっても、各Ｆ関数に含まれるＳ－ｂｏｘの組み合わせは、同じになっている方が望ましい場合がある。

【０１０８】

すなわち、例えばハードウェアやソフトウェアによってＦ関数に相当するデータ変換を行う場合、各Ｆ関数に含まれるＳ－ｂｏｘの組み合わせが同じになっていれば、Ｆ関数としてのハードウェアやソフトウェアは同一のものとして、各ラウンドにおいて、適宜、入出力を変更するのみで、各ラウンドにおけるＦ関数のデータ変換が実現できる。

【０１０９】

具体的な例を図１２を参照して説明する。図１２も、図１０と同様、Ｆｅｉｓｔｅｌ構造もしくは拡張Ｆｅｉｓｔｅｌ構造の一部分を切り出した構成を示し、入力データの系列（ｘ）と出力データの系列（ｙ）が同一であり、かつ上下に隣り合う位置にあるＦ関数２２１，２２２を示している。

【０１１０】

先行Ｆ関数２２１に含まれる４つのＳ－ｂｏｘは、上からＳ１，Ｓ２，Ｓ１，Ｓ２の順に配置し、次のラウンドの後続Ｆ関数２２２では、上からＳ２，Ｓ１，Ｓ２，Ｓ１の順にする。

なお、Ｓ１≠Ｓ２

である。

【０１１１】

このような設定とすれば、Ｓ１を２つ、Ｓ２を２つ、並列実行可能な構成とした実装を行なえば、その構成を適用して、Ｆ関数２２１，２２２を実行することが可能となり、実装上のコストを低下させ、また、装置の小型化も可能となる。

【０１１２】

図１２に示す構成でも、各Ｆ関数２２１，２２２において、対応するビット列に適用される非線形変換処理は、

Ｓ１→Ｓ２、または、

Ｓ２→Ｓ１、

となり、対応するビットデータ（例えば各バイト単位）の処理としては、図１０を参照して説明したと同様の処理となり、結果として、同様の効果、すなわち、出力系列に表れるデータが、ラウンド関数実行前の同一出力系列に出現するデータと一致する可能性を著しく低下させることができ、結果として、飽和攻撃の難易度を大きく向上させ、攻撃に対する耐性が高めることが可能となる。

【０１１３】

もう１つの具体的な例を図１３に示す。図１３は、図１１と同様、Ｆｅｉｓｔｅｌ構造もしくは拡張Ｆｅｉｓｔｅｌ構造の一部分を切り出した構成を示し、入力データの系列（ｘ）と出力データの系列（ｙ）が同一であり、かつ上下に隣り合う位置にある３つのＦ関数２３１～２３３を示している。

【０１１４】

先行Ｆ関数２３１に含まれる４つのＳ－ｂｏｘは、上からＳ１，Ｓ２，Ｓ３，Ｓ４の順

に配置し、次のラウンドの中間Ｆ関数２３２では、上からＳ２，Ｓ３，Ｓ４，Ｓ１の順、さらに次のラウンドの中間Ｆ関数２３３では、上からＳ３，Ｓ４，Ｓ１，Ｓ２の順に設定する。

なお、Ｓ１≠Ｓ２≠Ｓ３≠Ｓ４

である。

【０１１５】

このような設定とすれば、Ｓ１～Ｓ４を各々１つずつ並列実行可能な構成とした実装を行なえば、その構成を適用して、すべてのＦ関数２３１～２３３を実行することが可能となり、実装上のコストを低下させ、また、装置の小型化も可能となる。

【０１１６】

図１３に示す構成でも、各Ｆ関数２３１～２３３において、対応するビット列に適用される非線形変換処理は、

Ｓ１→Ｓ２→Ｓ３→Ｓ４→Ｓ１→Ｓ２・・・、

の順番となり、対応するビットデータ（例えば各バイト単位）の処理としては、図１０や図１１を参照して説明したと同様の処理となり、結果として、同様の効果、すなわち、出力系列に表れるデータが、ラウンド関数実行前の同一出力系列に出現するデータと一致する可能性を著しく低下させることができ、結果として、飽和攻撃の難易度を大きく向上させ、攻撃に対する耐性を高めることが可能となる。

【０１１７】

（２Ｂ）Ｓ－ｂｏｘを用いたブロック暗号において、２種類以上の異なるＳ－ｂｏｘを混在させることで代数的攻撃（ＸＳＬ攻撃）に対する耐性を向上させた構成

次に、Ｓ－ｂｏｘを用いたブロック暗号において、種類の異なるＳ－ｂｏｘを混ぜることで代数的攻撃（ＸＳＬ攻撃）に対する耐性を向上させる構成について説明する。

【０１１８】

（２Ｂ－１）代数的攻撃（ＸＳＬ攻撃）の概要

まず、ブロック暗号に対する攻撃として知られる代数的攻撃（ＸＳＬ攻撃）について説明する。ブロック暗号に対する代数的攻撃（ＸＳＬ攻撃）はＳ－ｂｏｘの代数表現を利用した攻撃である。Ｓ－ｂｏｘの入出力に関して代数式として表現すると複数の式を導き出すことができるが、その式の最大次数や含まれる項の数によって、攻撃に必要な計算量が変化する。

【０１１９】

代数的攻撃（ＸＳＬ攻撃）のひとつの実施例としてブール式を用いた方式がある。たとえばある８－ｂｉｔ入出力を持つＳ－ｂｏｘを複数組み込んだブロック暗号が存在するとする。このとき、８ビット入出力Ｓ－ｂｏｘにおいて、入力側と出力側のビットをそれぞれ、

入力Ｘ：（ｘ１，ｘ２，ｘ３，ｘ４，ｘ５，ｘ６，ｘ７，ｘ８）、

出力Ｙ：（ｙ１，ｙ２，ｙ３，ｙ４，ｙ５，ｙ６，ｙ７，ｙ８）、

とすると２次以下のブール式で書き表すことの式の数を評価する。

【０１２０】

より具体的に言うと、上記入出力Ｘ，Ｙをブール式で表した結果に含まれる項が、

（１，ｘｉ，ｙｉ，ｘｉｘｊ，ｙｉｙｊ，ｘｉｙｊ）

このような２次以下のいずれかの形で表すことのできる多項式の数を評価する。

【０１２１】

このように表された全ブール式のうち、最大次数を２次などの低次数に限定して取り出した際に独立な式の数がより多く、項の数が少ないと攻撃に有利だとされている。すなわち、このように最大次数を２次などに限定した際に独立な式の数がより多く、項の数が少ないと攻撃に有利だとされ、攻撃に対する耐性が劣ることになる。

また、ブール式のみではならず、拡大体ＧＦ（2^8）などの定義体の上で低い次数での代数表現ができる場合も同様な手法を使って代数的攻撃（ＸＳＬ攻撃）をすることが容易となり、攻撃耐性が弱いとされる。

【０１２２】
（２Ｂ－２）１種類のＳ－ｂｏｘを用いたときの問題点
次に、Ｓ－ｂｏｘを用いたブロック暗号において、１種類のみのＳ－ｂｏｘを利用した構成での、問題点、すなわち、代数的攻撃（ＸＳＬ攻撃）が行ないやすくなるという問題点について説明する。
【０１２３】
ｎ－ｂｉｔの入出力の非線形変換を実行するｓビットＳ－ｂｏｘには、以下の３つの代表的なタイプがある。
タイプ１：拡大体ＧＦ（２^{s}）上の逆元写像：Ｙ＝Ｘ$^{-1}$や、べき乗関数Ｙ＝Ｘ^{p}を利用したＳ－ｂｏｘ
タイプ２：例えば４－ｂｉｔ入出力等のｓビットより小さなＳ－ｂｏｘを複数組み合わせて作り出されたＳ－ｂｏｘ
タイプ３：ランダムに選択されたＳ－ｂｏｘ
これらの３タイプが代表的である。
特にタイプ１とタイプ２に関しては、ハードウェア（Ｈ／Ｗ）実装時に低コストで実装できるためしばしば利用されるＳ－ｂｏｘである。
以下、上記タイプ１～３、各々について、１種類のみのＳ－ｂｏｘを利用した構成での、問題点、すなわち、代数的攻撃（ＸＳＬ攻撃）が行ないやすくなるという問題点について説明する。
【０１２４】
〈タイプ１の問題点〉
タイプ１、すなわち、ＧＦ（２^{s}）上の逆元写像：Ｙ＝Ｘ$^{-1}$や、べき乗関数Ｙ＝Ｘ^{p}を利用したＳ－ｂｏｘの場合の問題点について説明する。
【０１２５】
たとえば、ＧＦ（２^{8}）上の逆元写像を用いたＳ－ｂｏｘの場合、ブール式で表すと２０数個の独立な２次式でかつ項数が８０数個となる表現があることが知られている。べき乗関数に関しても同様な単純な関係がある。またＧＦ（２^{8}）上だけではなくＧＦ（２^{s}）上で定義されるＳ－ｂｏｘでも同様の関係が見込まれる。
【０１２６】
これらの多項式表現を用いると、代数的攻撃（ＸＳＬ攻撃）に対する計算量を見積もることができ、暗号の設計の段階では安全性を確保するのに十分な計算量が得られるように十分な数のＳ－ｂｏｘを利用する必要がある。さらにＧＦ（２^{s}）上の逆元写像を用いたＳ－ｂｏｘの場合ＧＦ（２^{s}）でＸＹ＝１というような代数表現が可能であり次数の低い多項式を導出することも可能である。これらの性質を利用した攻撃方法も存在することが知られている。べき乗関数に関しても同様の結果が適用されうる。
【０１２７】
このように、ＧＦ（２^{s}）上の逆元写像やべき乗写像を用いたＳ－ｂｏｘだけを用いた暗号では、２種類の代数的性質を利用されるためその両者に対して配慮した設計をしなくてはならない。
【０１２８】
なお、逆元写像やべき乗関数の前後にアフィン変換を追加した作成されたＳ－ｂｏｘに関しても上記と同様のことが言える。
【０１２９】
〈タイプ２の問題点〉
次に、タイプ２、すなわち、より小さな（例えば４－ｂｉｔ）Ｓ－ｂｏｘを複数組み合わせて作り出されたＳ－ｂｏｘの場合の問題点について説明する。
【０１３０】
入出力が例えば４－ｂｉｔの小さなＳ－ｂｏｘを複数組み合わせて作り出した８ビットＳ－ｂｏｘを用いた場合を考える。４ビットＳ－ｂｏｘを３～５個利用して８ビットＳ－ｂｏｘを作り上げることができることが知られている。代数的攻撃（ＸＳＬ攻撃）のため

には４ビットＳ－ｂｏｘの入出力のビットに対して２次以下のブール式多項式を導出することになるが、そもそも入出力ビットの合計が８つしかないため、そのような低い次数で表現される独立な式が少なくとも２０数個程度存在することが知られている。よって、これを利用した攻撃を構成することができる。この傾向は大きな入出力サイズのＳ－ｂｏｘを作るためにより小さなサイズのＳ－ｂｏｘを用いて構成された場合に言えることである。

【０１３１】

しかしこの方式のメリットとしては、ＧＦ（2^8）上の逆元写像を用いたＳ－ｂｏｘを使った場合のようなＧＦ（2^s）の体の上で単純な代数的関係が存在する確率は極端に低くなることから、攻撃に必要な計算量が多くなることが分かっているため、上記ことがらより前出のＳ－ｂｏｘにくらべて代数的攻撃（ＸＳＬ攻撃）の観点からは良い点と悪い点が両者混在しているといえる。

【０１３２】

〈タイプ３の問題点〉

次に、タイプ３、すなわち、ランダムに選択されたＳ－ｂｏｘの場合の問題点について説明する。ランダムに選択されたＳ－ｂｏｘは上述のような代数的に見て弱い性質があることが望めず、代数的攻撃（ＸＳＬ攻撃）に対しての高い安全性が期待できるが、Ｈ／Ｗ実装コストが極めて高いため、すべてのＳ－ｂｏｘをランダムに選択されたＳ－ｂｏｘにすることは望ましくはないという問題がある。

【０１３３】

（２Ｂ－３）代数的性質の異なる複数種類のＳ－ｂｏｘを利用することにより、耐性を向上させた構成

上記の問題に鑑み、代数的性質の異なる２種類以上のＳ－ｂｏｘを利用することでブール多項式を用いた代数的攻撃（ＸＳＬ攻撃）、およびＧＦ（2^s）の体を利用した代数的攻撃（ＸＳＬ攻撃）の両者に対して耐性を向上さら、さらにすべてのＳ－ｂｏｘをランダムに選択されたＳ－ｂｏｘにする場合よりハードウェア（Ｈ／Ｗ）実装効率を高めた構成について、以下説明する。

【０１３４】

先に、説明したように、ｓ－ｂｉｔの入出力の非線形変換を実行するｓビットＳ－ｂｏｘには、以下の３つの代表的なタイプがある。

タイプ１：拡大体ＧＦ（2^s）上の逆元写像：$Y=X^{-1}$や、べき乗関数$Y=X^p$を利用したＳ－ｂｏｘ

タイプ２：ｔ－ｂｉｔの小さなＳ－ｂｏｘを複数組み合わせて作り出されたＳ－ｂｏｘ（ただしｔ＜ｓとする）

タイプ３：ランダムに選択されたＳ－ｂｏｘ

これらの３タイプが代表的である。

【０１３５】

本実施例では、これらの異なるタイプのＳ－ｂｏｘを組み合わせて利用することで、代数的攻撃（ＸＳＬ攻撃）に対する耐性を向上させ、またハードウェア（Ｈ／Ｗ）実装効率を高めた構成を実現するものである。すなわち、Ｓ－ｂｏｘを用いたブロック暗号において、２種類以上の異なるＳ－ｂｏｘを混在させることで代数的攻撃（ＸＳＬ攻撃）に対する耐性を向上させた構成である。なお、本実施例の適用可能な暗号処理構成は、非線形変換処理を実行するＳ－ｂｏｘを有する暗号処理構成であればよく、例えば、先に説明した以下の各種の暗号処理構成、すなわち、

（ア）ＳＰＮ（Substitution Permutation Network）構造、

（イ）Ｆｅｉｓｔｅｌ構造、

（ウ）拡張Ｆｅｉｓｔｅｌ構造、

これらのいずれにも適用可能である。

【０１３６】

本処理例においては、データ変換処理を実行するラウンド関数中に含まれる非線形変換

処理部としてのＳ－ｂｏｘを、以下の（ａ）～（ｄ）のいずれかの設定とする。
（ａ）一部をタイプ１のＳ－ｂｏｘとし、その他をタイプ２のＳ－ｂｏｘとする構成
（ｂ）一部をタイプ１のＳ－ｂｏｘとし、その他をタイプ３のＳ－ｂｏｘとする構成
（ｃ）一部をタイプ２のＳ－ｂｏｘとし、その他をタイプ３のＳ－ｂｏｘとする構成
（ｄ）一部をタイプ１のＳ－ｂｏｘとし、他の一部をタイプ２のＳ－ｂｏｘとし、その他をタイプ３のＳ－ｂｏｘと構成

【０１３７】

例えば、上記（ａ）の設定とする場合、
データ変換処理を実行するラウンド関数中に含まれる非線形変換処理部としてのＳ－ｂｏｘ中、半分のＳ－ｂｏｘをタイプ１、すなわち、ＧＦ（2^8）上の逆元写像を用いたＳ－ｂｏｘとし、それ以外のＳ－ｂｏｘを取り除いた架空の暗号を考え、そのもとでブール式を利用した代数的攻撃（ＸＳＬ攻撃）の計算量の見積もりを行い、十分な計算量が見積もられていれば、残りの半分のＳ－ｂｏｘを、タイプ２、すなわち、４－ｂｉｔの小さなＳ－ｂｏｘを複数組み合わせて作り出した８ビットＳ－ｂｏｘとする構成にする。

【０１３８】

このように、上記（ａ）のタイプ１とタイプ２の混在設定とした暗号処理構成にすることでＧＦ（2^8）上での計算量の見積もりをしたときに十分な耐性を持つことができれば、それぞれのＳ－ｂｏｘを単独に利用したときに比べて、総合的な耐性が向上したブロック暗号を作ることが可能である。

【０１３９】

上記設定に限らず、上記（ａ）～（ｄ）いずれの場合でも同様に、一部のＳ－ｂｏｘだけに限定しても代数的攻撃（ＸＳＬ攻撃）に対する耐性が十分高くなるように設定し、残りのＳ－ｂｏｘは実装効率などを配慮して決定すればよい。

【０１４０】

上記（ａ）～（ｄ）のように異なるタイプのＳ－ｂｏｘを配置した具体的な暗号処理構成例について、図１４～図１８を参照して説明する。図１４～図１８に示す例は、いずれも６ラウンドのラウンド関数実行部を持つ暗号処理構成を示し、各ラウンド関数実行部には、複数のＳ－ｂｏｘからなる非線形変換処理部と、線形変換処理部を有する。

【０１４１】

図１４は、６ラウンドからなり１ラウンドあたり１０個のＳ－ｂｏｘが含まれるＳＰＮブロック暗号の例を示している。ＳＰＮブロック暗号は、非線形変換層（Ｓ層）、線形変換層（Ｐ層）を有したデータ変換を各ラウンドにおいて実行する。各ラウンドの１０個のＳ－ｂｏｘは、例えば入力データを１０分割した分割データを入力して、非線形変換処理を実行して、非線形変換結果データを線形変換層（Ｐ層）に出力し、線形変換処理結果が次のラウンド関数実行部に出力される。最終段のラウンド関数実行部の出力が暗号文となる。

【０１４２】

図に示す各ラウンド関数実行部３０１～３０６の［S_1］，［S_2］は、それぞれタイプ１のＳ－ｂｏｘ、タイプ２のＳ－ｂｏｘを示し、上述した異なるタイプの非線形変換処理部としてのＳ－ｂｏｘである。

【０１４３】

図１４に示す例は、先行する３ラウンドのラウンド関数実行部３０１～３０３に、
タイプ１：拡大体ＧＦ（2^s）上の逆元写像：$Y=X^{-1}$や、べき乗関数$Y=X^p$を利用したＳ－ｂｏｘ
このタイプ１のＳ－ｂｏｘ［S_1］を配置し、
後続する３ラウンドのラウンド関数実行部３０１～３０３に、
タイプ２：４－ｂｉｔのような小さなＳ－ｂｏｘを複数組み合わせて作り出されたＳ－ｂｏｘ
このタイプ２のＳ－ｂｏｘ［S_2］を配置した構成例である。

【０１４４】

図１４の構成では、非線形変換処理は、前半のラウンドでは、タイプ１のＳ－ｂｏｘを適用した処理として実行され、後半のラウンドでは、タイプ２のＳ－ｂｏｘを適用した処理として実行される。一般的に代数的攻撃（ＸＳＬ攻撃）は、すべてを同一タイプのＳ－ｂｏｘとして仮定した上で実行されるものであり、このように異なるタイプのＳ－ｂｏｘが混在した設定である場合には、攻撃、すなわち解析が困難となる。結果として、代数的攻撃（ＸＳＬ攻撃）などの暗号解析に対する耐性の高い暗号処理構成が実現される。

【０１４５】

図１５は、図１４と同様、６ラウンドからなり１ラウンドあたり１０個のＳ－ｂｏｘが含まれるＳＰＮブロック暗号の例を示している。

図１５に示す例は、

第１，３，５の奇数ラウンドのラウンド関数実行部３２１，３２３，３２５には、

タイプ１：拡大体ＧＦ（2^s）上の逆元写像：$Y=X^{-1}$や、べき乗関数$Y=X^p$を利用したＳ－ｂｏｘ

このタイプ１のＳ－ｂｏｘ［S_1］を配置し、

第２，４，６の偶数ラウンドのラウンド関数実行部３２２，３２４，３２６には、

タイプ２：４－ｂｉｔのような小さなＳ－ｂｏｘを複数組み合わせて作り出されたＳ－ｂｏｘ

このタイプ２のＳ－ｂｏｘ［S_2］を配置した構成例である。

【０１４６】

図１５の構成では、非線形変換処理は、奇数ラウンドでは、タイプ１のＳ－ｂｏｘを適用した処理として実行され、偶数ラウンドでは、タイプ２のＳ－ｂｏｘを適用した処理として実行される。本構成においても、図１４の構成と同様、異なるタイプのＳ－ｂｏｘが混在した設定であり、代数的攻撃（ＸＳＬ攻撃）などの暗号解析に対する耐性の高い暗号処理構成が実現される。

【０１４７】

図１６は、図１４、図１５と同様、６ラウンドからなり１ラウンドあたり１０個のＳ－ｂｏｘが含まれるＳＰＮブロック暗号の例を示している。

図１６に示す例は、

すべてのラウンドのラウンド関数実行部３４１～３４６において、

タイプ１：拡大体ＧＦ（2^s）上の逆元写像：$Y=X^{-1}$や、べき乗関数$Y=X^p$を利用したＳ－ｂｏｘ

このタイプ１のＳ－ｂｏｘ［S_1］と、

タイプ２：４－ｂｉｔのような小さなＳ－ｂｏｘを複数組み合わせて作り出されたＳ－ｂｏｘ

このタイプ２のＳ－ｂｏｘ［S_2］を、

半数ずつ、すなわち５個ずつ配置した構成例である。

【０１４８】

各ラウンド関数実行部３４１～３４６に入力するデータは１０分割されて各Ｓボックスに入力される。１０分割された分割データd_1～d_{10}の前半のデータd_1～d_5は、タイプ１のＳ－ｂｏｘに入力されてタイプ１のＳ－ｂｏｘを適用した非線形変換処理が実行され、後半のデータd_6～d_{10}は、タイプ２のＳ－ｂｏｘに入力されてタイプ２のＳ－ｂｏｘを適用した非線形変換処理が実行されることになる。

【０１４９】

図１６の構成においても、図１４、図１５の構成と同様、異なるタイプのＳ－ｂｏｘが混在した設定であり、代数的攻撃（ＸＳＬ攻撃）などの暗号解析に対する耐性の高い暗号処理構成が実現される。

【０１５０】

図１７は、図１４～図１６と同様、６ラウンドからなり１ラウンドあたり１０個のＳ－ｂｏｘが含まれるＳＰＮブロック暗号の例を示している。

図１７に示す例は、図１６に示す例と同様、

すべてのラウンドのラウンド関数実行部３６１～３６６において、
タイプ１：拡大体ＧＦ（２^{s}）上の逆元写像：Ｙ＝Ｘ$^{-1}$や、べき乗関数Ｙ＝Ｘ^{p}を利用したＳ－ｂｏｘ
このタイプ１のＳ－ｂｏｘ［Ｓ$_{1}$］と、
タイプ２：４－ｂｉｔのような小さなＳ－ｂｏｘを複数組み合わせて作り出されたＳ－ｂｏｘ
このタイプ２のＳ－ｂｏｘ［Ｓ$_{2}$］を、
半数ずつ、すなわち５個ずつ配置した構成例である。

【０１５１】

各ラウンド関数実行部３６１～３６６に入力するデータは１０分割されて各Ｓボックスに入力される。１０分割された分割データｄ$_{1}$～ｄ$_{10}$の奇数番目の分割データｄ$_{1}$，ｄ$_{3}$，ｄ$_{5}$，ｄ$_{7}$，ｄ$_{9}$は、タイプ１のＳ－ｂｏｘに入力されてタイプ１のＳ－ｂｏｘを適用した非線形変換処理が実行され、偶数番目の分割データｄ$_{2}$，ｄ$_{4}$，ｄ$_{6}$，ｄ$_{8}$，ｄ$_{10}$は、タイプ２のＳ－ｂｏｘに入力されてタイプ２のＳ－ｂｏｘを適用した非線形変換処理が実行されることになる。

【０１５２】

図１７の構成においても、図１４～図１６の構成と同様、異なるタイプのＳ－ｂｏｘが混在した設定であり、代数的攻撃（ＸＳＬ攻撃）などの暗号解析に対する耐性の高い暗号処理構成が実現される。

【０１５３】

図１６や図１７に示す構成では、各ラウンドにおいて並列に実行するＳ－ｂｏｘは、それぞれタイプ１のＳ－ｂｏｘと、タイプ２のＳ－ｂｏｘが５個ずつであり、これはすべてのラウンドにおいて共通している。従って、例えば、実装として、タイプ１とタイプ２のＳ－ｂｏｘが５個ずつ並列実行可能な構成とすれば、その構成を繰り返し適用してすべてのラウンドのラウンド関数を実行することが可能となり、実装面においてもコストダウン、小型化が図れるというメリットがある。

【０１５４】

図１８には、Ｆｅｉｓｔｅｌ構造における各ラウンド関数実行部３８１～３８６に複数の異なるタイプのＳ－ｂｏｘを配置した例を示している。
図１８に示す例は、
すべてのラウンドのラウンド関数実行部３８１～３８６において、
タイプ１：拡大体ＧＦ（２^{s}）上の逆元写像：Ｙ＝Ｘ$^{-1}$や、べき乗関数Ｙ＝Ｘ^{p}を利用したＳ－ｂｏｘ
このタイプ１のＳ－ｂｏｘ［Ｓ$_{1}$］と、
タイプ２：４－ｂｉｔのような小さなＳ－ｂｏｘを複数組み合わせて作り出されたＳ－ｂｏｘ
このタイプ２のＳ－ｂｏｘ［Ｓ$_{2}$］を、
半数ずつ、すなわち２個ずつ配置した構成例である。

【０１５５】

各ラウンド関数実行部３８１～３８６に入力するデータは４分割されて各Ｓボックスに入力される。４分割された分割データｄ$_{1}$～ｄ$_{4}$の奇数番目の分割データｄ$_{1}$，ｄ$_{3}$は、タイプ１のＳ－ｂｏｘに入力されてタイプ１のＳ－ｂｏｘを適用した非線形変換処理が実行され、偶数番目の分割データｄ$_{2}$，ｄ$_{4}$は、タイプ２のＳ－ｂｏｘに入力されてタイプ２のＳ－ｂｏｘを適用した非線形変換処理が実行されることになる。

【０１５６】

図１８の構成においても、図１４～図１７の構成と同様、異なるタイプのＳ－ｂｏｘが混在した設定であり、代数的攻撃（ＸＳＬ攻撃）などの暗号解析に対する耐性の高い暗号処理構成が実現される。

【０１５７】

なお、図１４～図１８に示す例では、タイプ１とタイプ２の２種類のタイプのＳ－ｂｏ

ｘを混在させて利用した構成例を示したが、異なるタイプのＳ－ｂｏｘの混在構成としては、先に説明したように、

（ａ）一部をタイプ１のＳ－ｂｏｘとし、その他をタイプ２のＳ－ｂｏｘとする構成

（ｂ）一部をタイプ１のＳ－ｂｏｘとし、その他をタイプ３のＳ－ｂｏｘとする構成

（ｃ）一部をタイプ２のＳ－ｂｏｘとし、その他をタイプ３のＳ－ｂｏｘとする構成

（ｄ）一部をタイプ１のＳ－ｂｏｘとし、他の一部をタイプ２のＳ－ｂｏｘとし、その他をタイプ３のＳ－ｂｏｘと構成

これらの各種の混在構成が可能であり、いずれの場合においても、代数的攻撃（ＸＳＬ攻撃）に対する耐性の向上が実現される。

【０１５８】

（２Ｃ）Ｓ－ｂｏｘを用いたＦｅｉｓｔｅｌ暗号もしくは拡張Ｆｅｉｓｔｅｌ型暗号において上記（２Ａ），（２Ｂ）を同時に実現する構成

次に、Ｓ－ｂｏｘを用いたＦｅｉｓｔｅｌ暗号もしくは拡張Ｆｅｉｓｔｅｌ型暗号において上記（２Ａ），（２Ｂ）、すなわち、

（２Ａ）Ｓ－ｂｏｘを用いたＦｅｉｓｔｅｌもしくは拡張Ｆｅｉｓｔｅｌ型暗号において２種類以上の異なるＳ－ｂｏｘを配置することで飽和攻撃に対する耐性を向上させた構成

（２Ｂ）Ｓ－ｂｏｘを用いたブロック暗号において、２種類以上の異なるＳ－ｂｏｘを混在させることで代数的攻撃（ＸＳＬ攻撃）に対する耐性を向上させた構成

これらの構成を同時に実現する構成例について説明する。

【０１５９】

上述した（２Ａ）の構成は、２種類以上のＳ－ｂｏｘをＦｅｉｓｔｅｌ構造または拡張Ｆｅｉｓｔｅｌ構造に適用することで飽和攻撃に対する耐性を向上させる構成であり、上述した（２Ｂ）の構成は、２種類以上のＳ－ｂｏｘをＳ－ｂｏｘを持つ任意のブロック暗号に対して用いることで代数的攻撃（ＸＳＬ攻撃）に対する耐性を向上させる構成である。

【０１６０】

これら（２Ａ），（２Ｂ）の構成は、併せて実現することが可能である。すなわち、（２Ａ），（２Ｂ）に必要な性質を満足する２種類以上のＳ－ｂｏｘを利用して同時に両方の攻撃に対する耐性を向上させたＦｅｉｓｔｅｌまたは拡張Ｆｅｉｓｔｅｌ構造を持つブロック暗号を構成することが可能となる。

【０１６１】

具体的には、上述した

（２Ａ）Ｓ－ｂｏｘを用いたＦｅｉｓｔｅｌもしくは拡張Ｆｅｉｓｔｅｌ型暗号において２種類以上の異なるＳ－ｂｏｘを配置することで飽和攻撃に対する耐性を向上させた構成

において説明した図１０～図１３の各構成において適用した異なる非線形変換処理を実行するＳ－ｂｏｘ［Ｓ１］，［Ｓ２］，［Ｓ３］，［Ｓ４］・・のそれぞれについて、

上述した

（２Ｂ）Ｓ－ｂｏｘを用いたブロック暗号において、２種類以上の異なるＳ－ｂｏｘを混在させることで代数的攻撃（ＸＳＬ攻撃）に対する耐性を向上させた構成

において説明した異なるタイプのＳ－ｂｏｘ、すなわち、

タイプ１：拡大体ＧＦ（２^{s}）上の逆元写像：Ｙ＝Ｘ$^{-1}$や、べき乗関数Ｙ＝Ｘ^{p}を利用したＳ－ｂｏｘ

タイプ２：４－ｂｉｔのような小さなＳ－ｂｏｘを複数組み合わせて作り出されたＳ－ｂｏｘ

タイプ３：ランダムに選択されたＳ－ｂｏｘ

これらの３タイプを対応付けて設定する。

【０１６２】

例えば、図１０に示す構成において、

Ｓ－ｂｏｘ［Ｓ１］と、Ｓ－ｂｏｘ［Ｓ２］とを、（２Ｂ）において説明した異なるタイプのＳ－ｂｏｘとして設定することで、
飽和攻撃に対しても、代数的攻撃（ＸＳＬ攻撃）に対しても耐性の高い構成が実現される。

【０１６３】
図１１～図１３に示す構成においても、同様であり、
Ｓ－ｂｏｘ［Ｓ１］，［Ｓ２］・・を、（２Ｂ）において説明した異なるタイプのＳ－ｂｏｘとして設定することで、
飽和攻撃に対しても、代数的攻撃（ＸＳＬ攻撃）に対しても耐性の高い構成が実現される。

【０１６４】
［３．暗号処理装置の構成例］
最後に、上述した実施例に従った暗号処理を実行する暗号処理装置としてのＩＣモジュール７００の構成例を図１９に示す。上述の処理は、例えばＰＣ、ＩＣカード、リーダライタ、その他、様々な情報処理装置において実行可能であり、図１９に示すＩＣモジュール７００は、これら様々な機器に構成することが可能である。

【０１６５】
図１９に示すＣＰＵ(Central processing Unit)７０１は、暗号処理の開始や、終了、データの送受信の制御、各構成部間のデータ転送制御、その他の各種プログラムを実行するプロセッサである。メモリ７０２は、ＣＰＵ７０１が実行するプログラム、あるいは演算パラメータなどの固定データを格納するＲＯＭ（Read-Only-Memory）、ＣＰＵ７０１の処理において実行されるプログラム、およびプログラム処理において適宜変化するパラメータの格納エリア、ワーク領域として使用されるＲＡＭ（Random Access Memory）等からなる。また、メモリ７０２は暗号処理に必要な鍵データや、暗号処理において適用する変換テーブル（置換表）や変換行列に適用するデータ等の格納領域として使用可能である。なおデータ格納領域は、耐タンパ構造を持つメモリとして構成されることが好ましい。

【０１６６】
暗号処理部７０３は、例えば上述した各種の暗号処理構成、すなわち、
（ア）ＳＰＮ（Substitution Permutation Network）構造、
（イ）Ｆｅｉｓｔｅｌ構造、
（ウ）拡張Ｆｅｉｓｔｅｌ構造、
これらの各構成のいずれかの構造を適用した共通鍵ブロック暗号処理アルゴリズムに従った暗号処理、復号処理を実行する。

【０１６７】
また、暗号処理部７０３は、上述した各実施例に対応した構成、すなわち、
（２Ａ）Ｓ－ｂｏｘを用いたＦｅｉｓｔｅｌもしくは拡張Ｆｅｉｓｔｅｌ型暗号において２種類以上の異なるＳ－ｂｏｘを配置した構成、
（２Ｂ）Ｓ－ｂｏｘを用いたブロック暗号において、２種類以上の異なるタイプのＳ－ｂｏｘを混在させた構成、
（２Ｃ）Ｓ－ｂｏｘを用いたＦｅｉｓｔｅｌ暗号もしくは拡張Ｆｅｉｓｔｅｌ型暗号において上記（２Ａ），（２Ｂ）を同時に実現する構成、
これらの構成のいずれかに対応する構成を持つ非線形変換処理部としてのＳ－ｂｏｘを持つ。

【０１６８】
なお、ここでは、暗号処理手段を個別モジュールとした例を示したが、このような独立した暗号処理モジュールを設けず、例えば暗号処理プログラムをＲＯＭに格納し、ＣＰＵ７０１がＲＯＭ格納プログラムを読み出して実行するように構成してもよい。

【０１６９】
乱数発生器７０４は、暗号処理に必要となる鍵の生成などにおいて必要となる乱数の発生処理を実行する。

【０１７０】
送受信部７０５は、外部とのデータ通信を実行するデータ通信処理部であり、例えばリーダライタ等、ＩＣモジュールとのデータ通信を実行し、ＩＣモジュール内で生成した暗号文の出力、あるいは外部のリーダライタ等の機器からのデータ入力などを実行する。
【０１７１】
このＩＣモジュール７００は、上述した実施例に従った非線形変換処理部としてのＳ－ｂｏｘの配列を有しており、結果として、
（２Ａ）Ｓ－ｂｏｘを用いたＦｅｉｓｔｅｌもしくは拡張Ｆｅｉｓｔｅｌ型暗号において２種類以上の異なるＳ－ｂｏｘを配置することで飽和攻撃に対する耐性を向上させた構成、
（２Ｂ）Ｓ－ｂｏｘを用いたブロック暗号において、２種類以上の異なるＳ－ｂｏｘを混在させることで代数的攻撃（ＸＳＬ攻撃）に対する耐性を向上させた構成、
（２Ｃ）Ｓ－ｂｏｘを用いたＦｅｉｓｔｅｌ暗号もしくは拡張Ｆｅｉｓｔｅｌ型暗号において上記（２Ａ），（２Ｂ）を同時に実現する構成、
これらのいずれかの構成を持ち、飽和攻撃や代数的攻撃（ＸＳＬ攻撃）に対する耐性が向上した構成を持つ。
【０１７２】
以上、特定の実施例を参照しながら、本発明について詳解してきた。しかしながら、本発明の要旨を逸脱しない範囲で当業者が該実施例の修正や代用を成し得ることは自明である。すなわち、例示という形態で本発明を開示してきたのであり、限定的に解釈されるべきではない。本発明の要旨を判断するためには、特許請求の範囲の欄を参酌すべきである。
【０１７３】
なお、明細書中において説明した一連の処理はハードウェア、またはソフトウェア、あるいは両者の複合構成によって実行することが可能である。ソフトウェアによる処理を実行する場合は、処理シーケンスを記録したプログラムを、専用のハードウェアに組み込まれたコンピュータ内のメモリにインストールして実行させるか、あるいは、各種処理が実行可能な汎用コンピュータにプログラムをインストールして実行させることが可能である。
【０１７４】
例えば、プログラムは記録媒体としてのハードディスクやＲＯＭ（Read Only Memory）に予め記録しておくことができる。あるいは、プログラムはフレキシブルディスク、ＣＤ－ＲＯＭ(Compact Disc Read Only Memory)，ＭＯ(Magneto optical)ディスク，ＤＶＤ(Digital Versatile Disc)、磁気ディスク、半導体メモリなどのリムーバブル記録媒体に、一時的あるいは永続的に格納（記録）しておくことができる。このようなリムーバブル記録媒体は、いわゆるパッケージソフトウエアとして提供することができる。
【０１７５】
なお、プログラムは、上述したようなリムーバブル記録媒体からコンピュータにインストールする他、ダウンロードサイトから、コンピュータに無線転送したり、ＬＡＮ(Local Area Network)、インターネットといったネットワークを介して、コンピュータに有線で転送し、コンピュータでは、そのようにして転送されてくるプログラムを受信し、内蔵するハードディスク等の記録媒体にインストールすることができる。
【０１７６】
なお、明細書に記載された各種の処理は、記載に従って時系列に実行されるのみならず、処理を実行する装置の処理能力あるいは必要に応じて並列的にあるいは個別に実行されてもよい。また、本明細書においてシステムとは、複数の装置の論理的集合構成であり、各構成の装置が同一筐体内にあるものには限らない。
【産業上の利用可能性】
【０１７７】
上述したように、本発明の一実施例の構成によれば、共通鍵ブロック暗号を適用した復

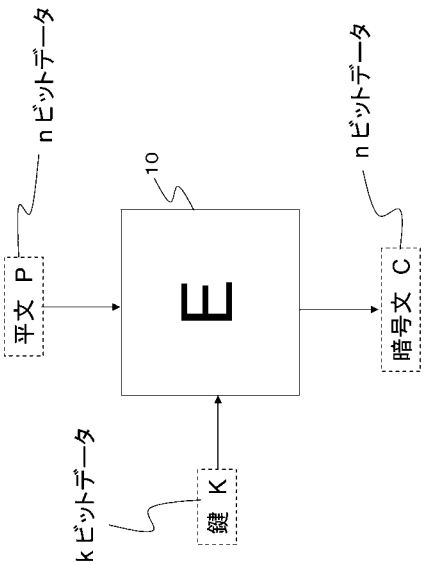
号処理において、ラウンド関数実行部に設定される非線形変換処理部としてのＳボックスに、少なくとも２種類以上の複数の異なるＳボックスを利用した構成とした。本構成により、飽和攻撃に対する耐性を高めることが可能となる。また、Ｓボックスのタイプとして、異なるタイプのものを混在させる本発明の一実施例の構成によれば、代数的攻撃（ＸＳＬ攻撃）に対する耐性を高めることが可能となり、安全性の高い復号処理装置が実現される。

【符号の説明】

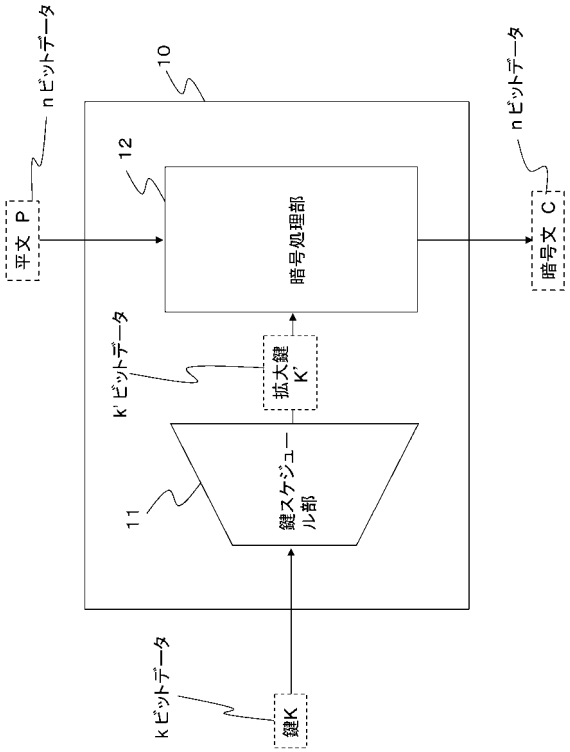
【０１７８】

１０　共通鍵ブロック暗号処理部Ｅ
１１　鍵スケジューリング部
１２　暗号処理部
２０　ラウンド関数実行部
２１　排他的論理和演算部
２２　非線形変換処理部
２３　線形変換処理部
３０　Ｆ関数部
３１　排他的論理和演算部
３２　非線形変換処理部
３３　線形変換処理部
３４　排他的論理和演算部
４１，４２　Ｆ関数部
５０　非線形変換処理部
５１　Ｓボックス
１０１，１０２　Ｆ関数
２０１，２０２　Ｆ関数
２１１～２３１　Ｆ関数
２２１，２２２　Ｆ関数
２３１～２３３　Ｆ関数
３０１～３０６　ラウンド関数実行部
３２１～３２６　ラウンド関数実行部
３４１～３４６　ラウンド関数実行部
３６１～３６６　ラウンド関数実行部
３８１～３８６　ラウンド関数実行部
７００　ＩＣモジュール
７０１　ＣＰＵ(Central processing Unit)
７０２　メモリ
７０３　暗号処理部
７０４　乱数発生器
７０５　送受信部

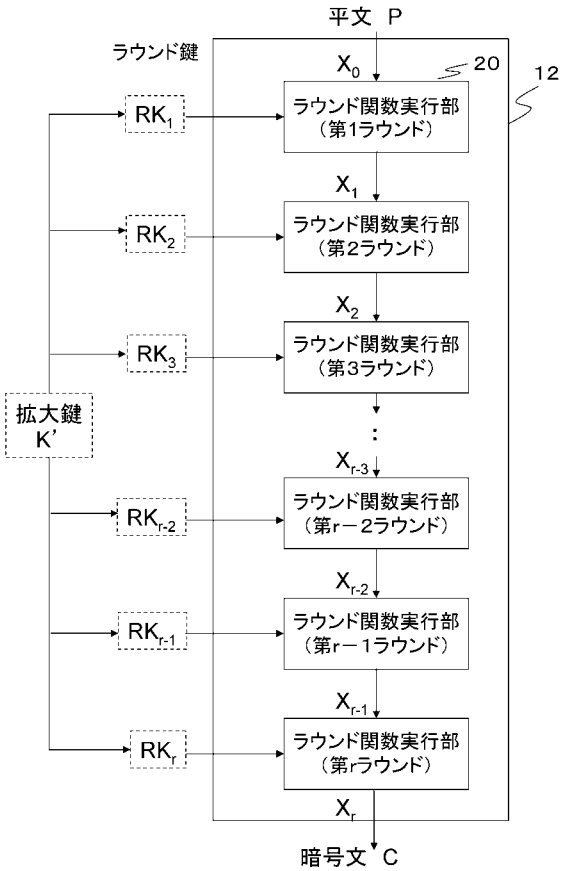
【図１】


nビットデータ
10
nビットデータ
平文 P
E
暗号文 C
kビットデータ
鍵 K

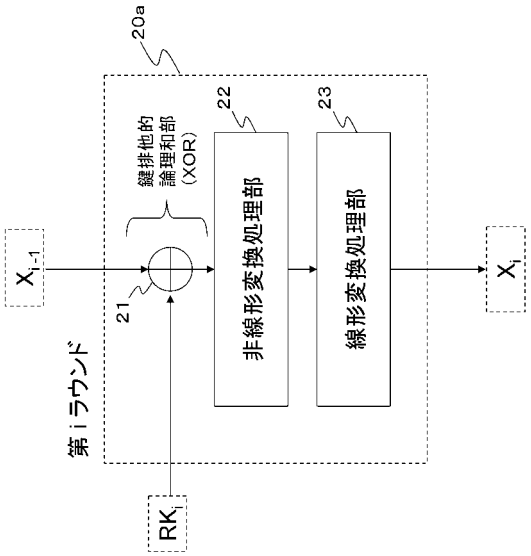
【図２】


nビットデータ
10
nビットデータ
12
平文 P
暗号処理部
暗号文 C
k'ビットデータ
拡大鍵 K'
鍵スケジュール部
11
kビットデータ
鍵K

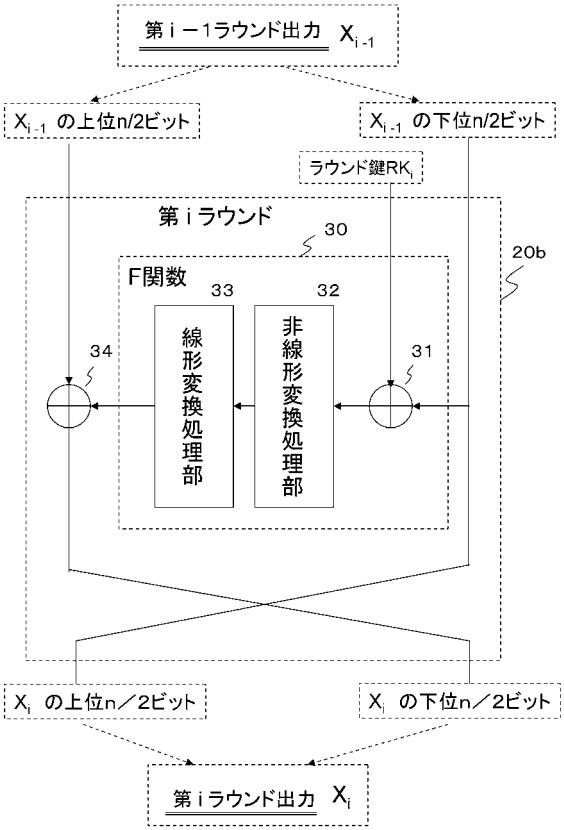
【図３】


平文 P
ラウンド鍵
X0
20
12
RK1
ラウンド関数実行部（第1ラウンド）
X1
RK2
ラウンド関数実行部（第2ラウンド）
X2
RK3
ラウンド関数実行部（第3ラウンド）
拡大鍵 K'
Xr-3
RKr-2
ラウンド関数実行部（第r－2ラウンド）
Xr-2
RKr-1
ラウンド関数実行部（第r－1ラウンド）
Xr-1
RKr
ラウンド関数実行部（第rラウンド）
Xr
暗号文 C

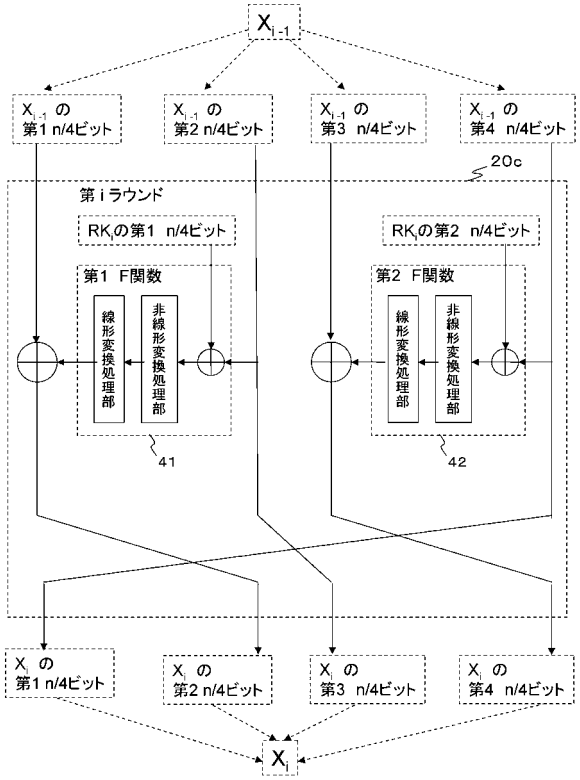
【図４】


20a
鍵排他的論理和部（XOR）
22
23
Xi-1
非線形変換処理部
線形変換処理部
Xi
21
第 i ラウンド
RKi

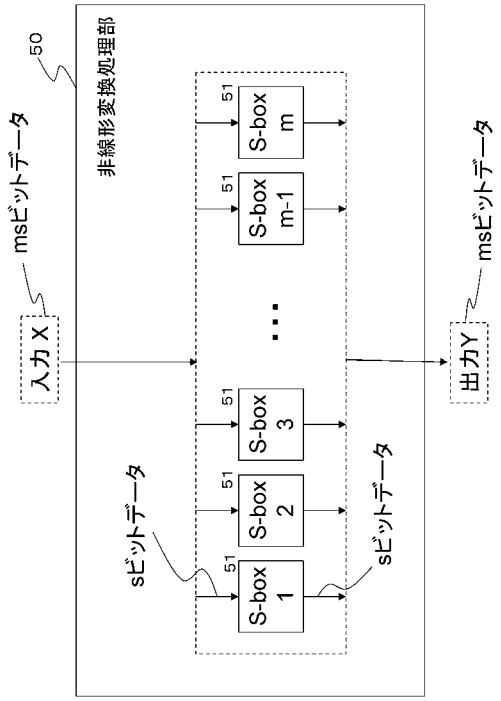
【図５】


第ｉ－1ラウンド出力 X_{i-1}
X_{i-1} の上位n/2ビット
X_{i-1} の下位n/2ビット
ラウンド鍵RK_i
第ｉラウンド
30
20b
F関数
33
32
線形変換処理部
非線形変換処理部
34
31
X_i の上位n／2ビット
X_i の下位n／2ビット
第ｉラウンド出力 X_i

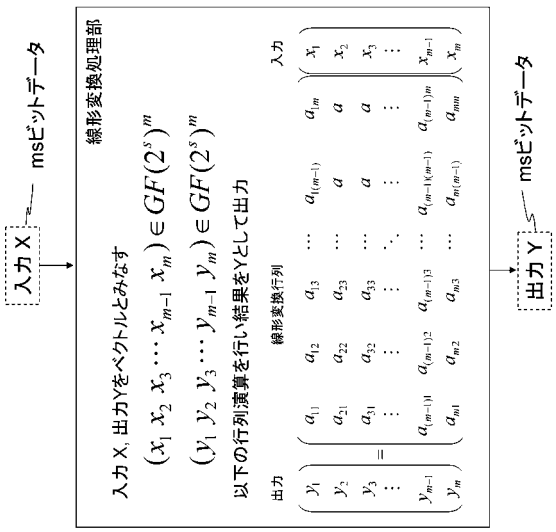
【図６】


X_{i-1}
X_{i-1} の第1 n/4ビット
X_{i-1} の第2 n/4ビット
X_{i-1} の第3 n/4ビット
X_{i-1} の第4 n/4ビット
20c
第ｉラウンド
RK_iの第1 n/4ビット
RK_iの第2 n/4ビット
第1 F関数
第2 F関数
線形変換処理部
非線形変換処理部
41
42
X_i の第1 n/4ビット
X_i の第2 n/4ビット
X_i の第3 n/4ビット
X_i の第4 n/4ビット
X_i

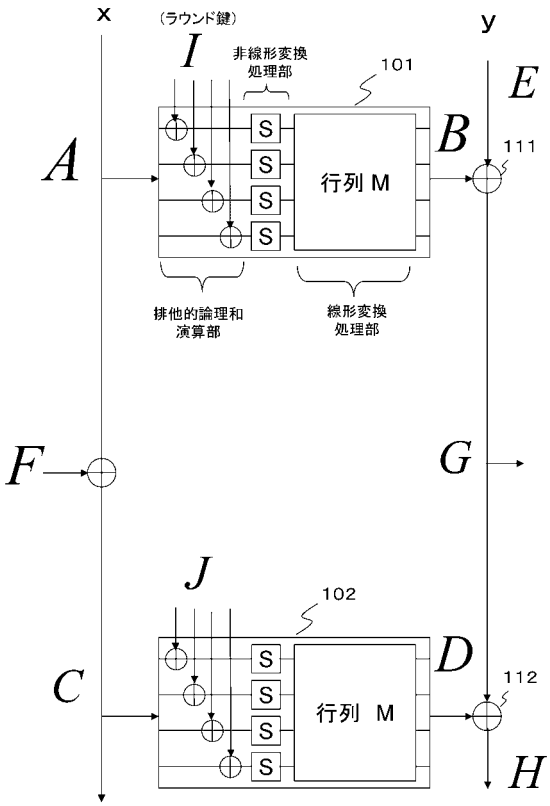
【図７】


50
非線形変換処理部
msビットデータ
入力X
出力Y
msビットデータ
51
S-box 1
S-box 2
S-box 3
S-box m-1
S-box m
sビットデータ
sビットデータ

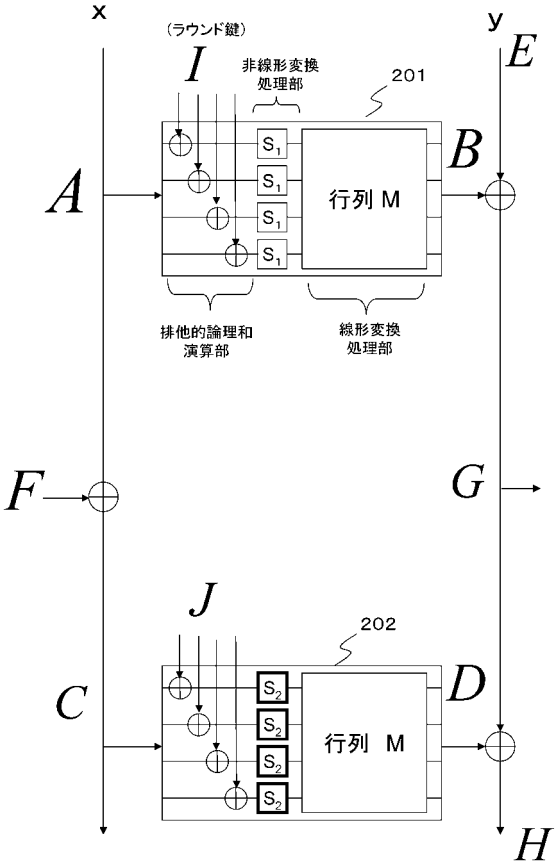
【図８】


msビットデータ
入力X
線形変換処理部
入力X, 出力Yをベクトルとみなす
$(x_1\ x_2\ x_3 \cdots x_{m-1}\ x_m) \in GF(2^s)^m$
$(y_1\ y_2\ y_3 \cdots y_{m-1}\ y_m) \in GF(2^s)^m$
以下の行列演算を行い結果をYとして出力
線形変換行列
出力
入力
出力Y
msビットデータ

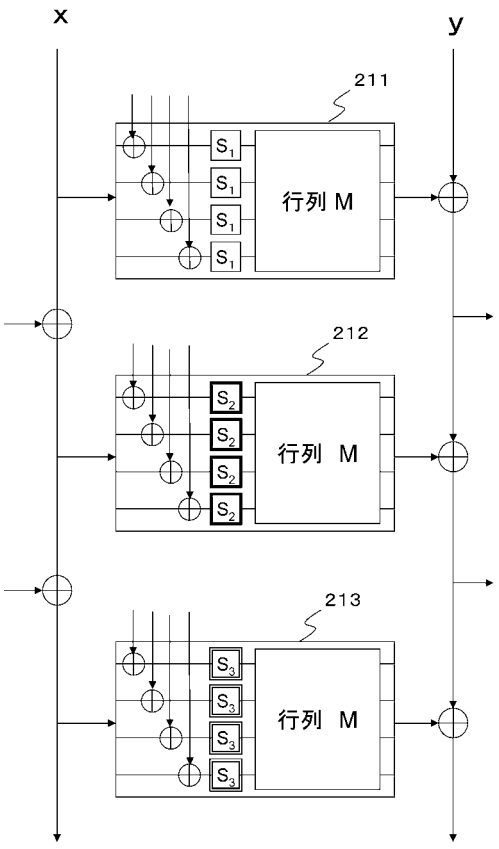
【図９】



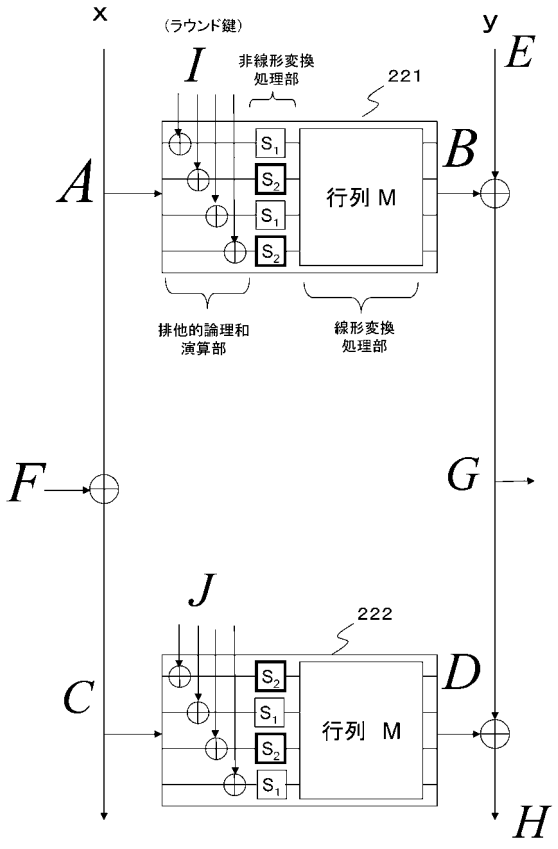
【図１０】



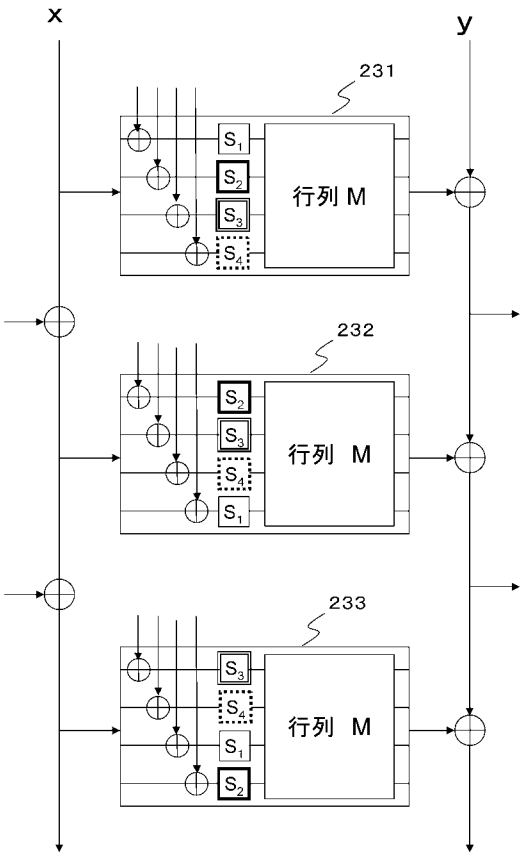
【図１１】



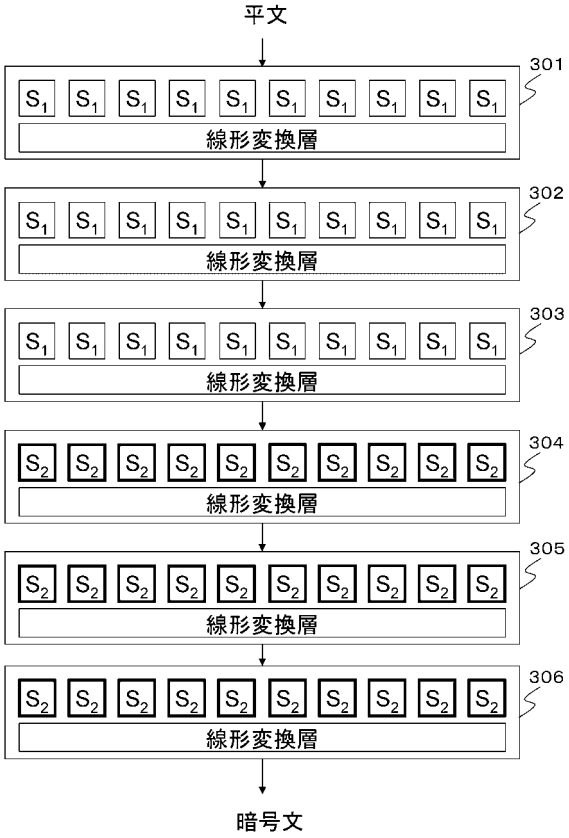
【図１２】



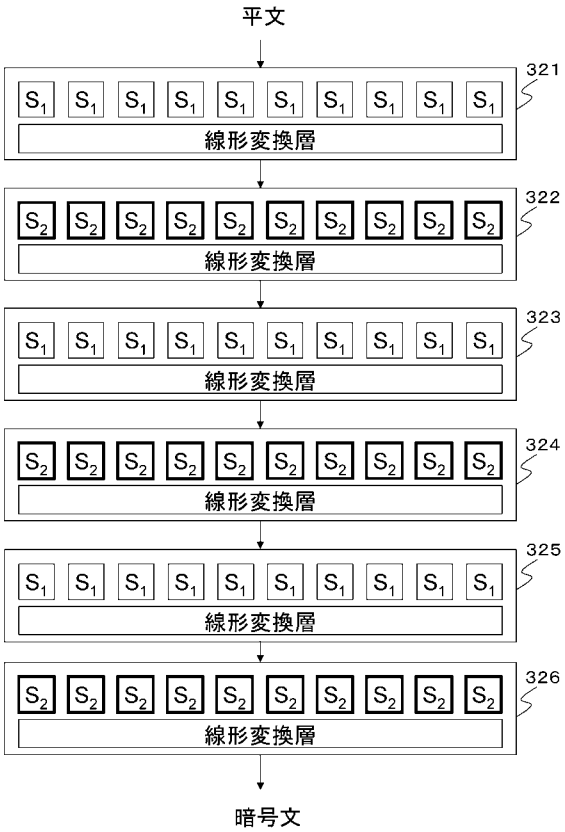
【図１３】



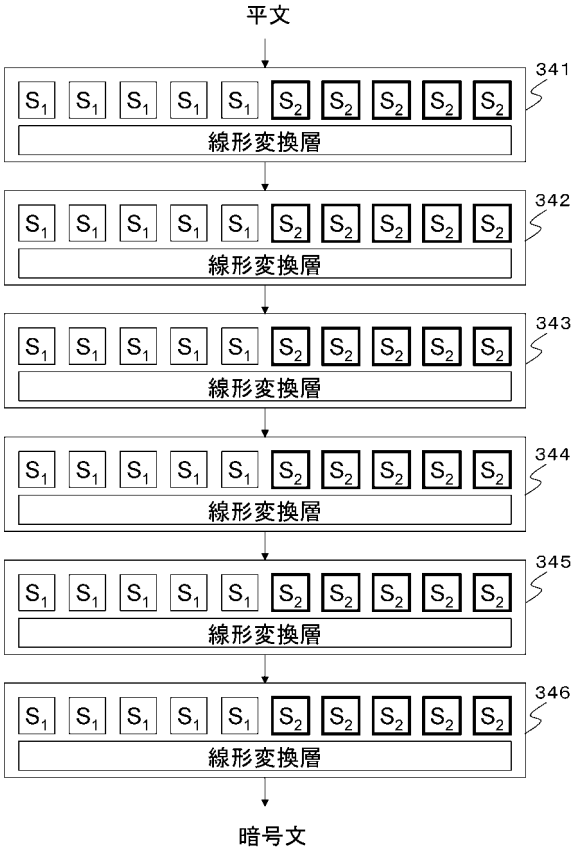
【図１４】



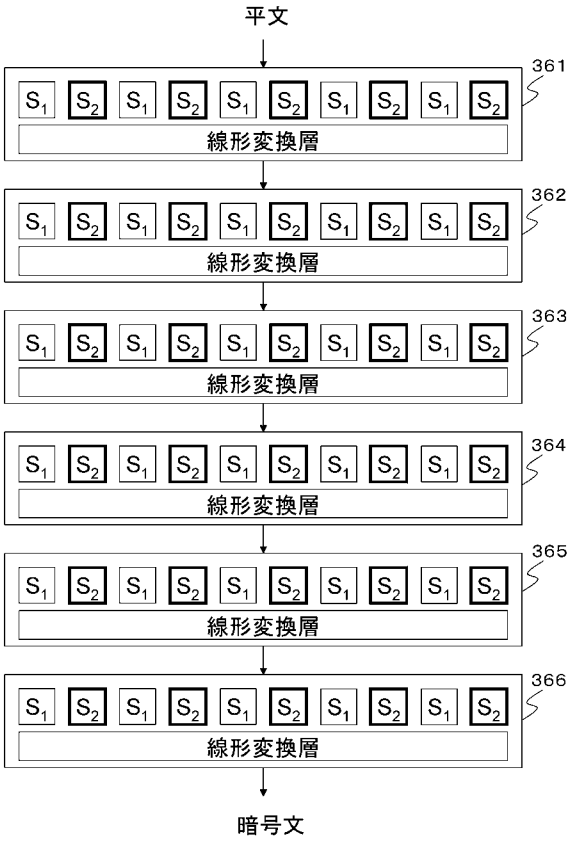
【図１５】



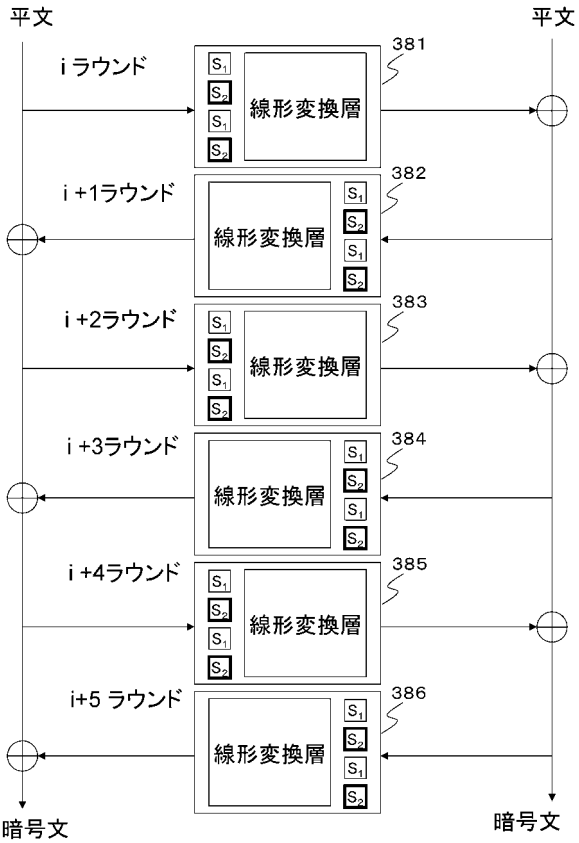
【図１６】



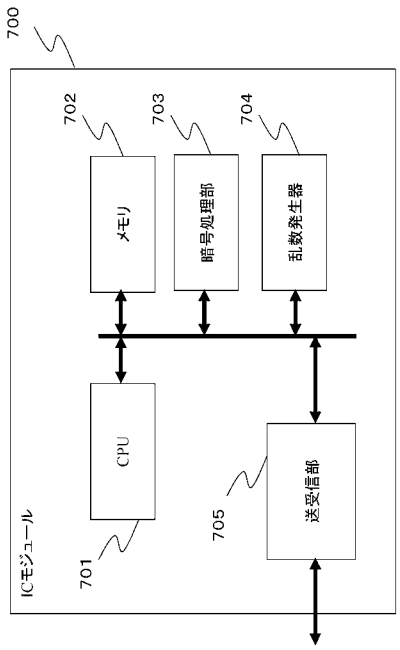
【図１７】


平文
S1 S2 S1 S2 S1 S2 S1 S2 S1 S2
線形変換層
361
362
363
364
365
366
暗号文

【図１８】


平文
平文
i ラウンド
i +1ラウンド
i +2ラウンド
i +3ラウンド
i +4ラウンド
i+5 ラウンド
S1 S2 S1 S2
線形変換層
381
382
383
384
385
386
暗号文
暗号文

【図１９】


700
702
703
704
メモリ
暗号処理部
乱数発生器
CPU
送受信部
705
701
ICモジュール

フロントページの続き

(72)発明者　白井　太三
東京都港区港南１丁目７番１号　ソニー株式会社内
(72)発明者　渋谷　香士
東京都港区港南１丁目７番１号　ソニー株式会社内
(72)発明者　秋下　徹
東京都港区港南１丁目７番１号　ソニー株式会社内
(72)発明者　盛合　志帆
東京都港区南青山二丁目６番２１号　株式会社ソニー・コンピュータエンタテインメント内

審査官　中里　裕正

(56)参考文献　米国特許第５００３５９７（ＵＳ，Ａ）
国際公開第０１／６７４２５（WO，Ａ１）
通信・放送機構　情報通信セキュリティ技術に関する研究開発プロジェクト，共通鍵ブロック暗号の選択／設計／評価に関するドキュメント，通信・放送機構，２０００年１２月，p.97-106

(58)調査した分野(Int.Cl.，ＤＢ名)
Ｈ０４Ｌ　　９／０６
Ｇ０９Ｃ　　１／００
ＪＳＴＰｌｕｓ／ＪＭＥＤＰｌｕｓ／ＪＳＴ７５８０（ＪＤｒｅａｍＩＩＩ）