(19)日本国特許庁(JP)　(12)特　許　公　報(B2)　(11)特許番号
特許第6406350号
(P6406350)
(45)発行日　平成30年10月17日(2018.10.17)　(24)登録日　平成30年9月28日(2018.9.28)

(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 13　(全 93 頁)

(21)出願番号　特願2016-510146(P2016-510146)
(86)(22)出願日　平成27年2月24日(2015.2.24)
(86)国際出願番号　PCT/JP2015/055281
(87)国際公開番号　WO2015/146431
(87)国際公開日　平成27年10月1日(2015.10.1)
審査請求日　平成30年1月9日(2018.1.9)
(31)優先権主張番号　特願2014-68292(P2014-68292)
(32)優先日　平成26年3月28日(2014.3.28)
(33)優先権主張国　日本国(JP)

(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】暗号処理装置、および暗号処理方法、並びにプログラム

(57)【特許請求の範囲】
【請求項１】
入力データに対するラウンド演算を実行して出力データを生成する暗号処理部と、
前記暗号処理部におけるラウンド演算において適用するラウンド鍵を前記暗号処理部に出力する鍵スケジュール部を有し、
前記暗号処理部は、
データ変換関数Ｅと、前記データ変換関数Ｅの逆関数E^{-1}をシーケンシャルに実行するインボリューション性を有し、
前記関数Ｅ、または、前記逆関数E^{-1}のいずれか一方において１回以上の定数を適用したラウンド演算を実行する暗号処理装置。
【請求項２】
前記暗号処理部は、
前記定数を適用したラウンド演算として、
前記定数と変換対象データ、または前記定数とラウンド鍵との排他的論理和演算を実行する請求項１に記載の暗号処理装置。
【請求項３】
前記ラウンド演算を実行するラウンド演算部は、線形変換処理を実行する線形変換処理部を有し、
前記暗号処理部は、
前記定数を、前記線形変換処理部と繋がる排他的論理和部に入力して、変換対象データ

、またはラウンド鍵との排他的論理和演算を行う請求項１に記載の暗号処理装置。
【請求項４】
前記入力データ、および前記定数は、
各要素が１ビット以上のｍ×ｎ個の要素からなるステートであり、
前記線形変換部は、前記ステートに対して線形変換行列を適用した行列演算を実行する構成であり、
前記定数は、
定数を入力する排他的論理和部と繋がる線形変換処理部が線形変換処理に適用する線形変換行列と、該定数との行列演算結果であるステートの構成要素が全て非ゼロとなる条件を満たすステートである請求項３に記載の暗号処理装置。
【請求項５】
前記入力データ、および前記定数は、
各要素が４ビットの４×４個の要素からなるステートであり、
前記線形変換部は、前記ステートに対して線形変換行列を適用した行列演算を実行する構成であり、
前記定数は、
定数を入力する排他的論理和部の隣接位置の線形変換処理部が線形変換処理に適用する線形変換行列と、該定数との行列演算結果であるステートの構成要素が全て非ゼロとなる条件を満たす４×４ステートである請求項３に記載の暗号処理装置。
【請求項６】
前記線形変換部は、
前記ステートの各列要素単位で行列を適用して線形変換を行う列拡散演算、
または、前記ステートの各行要素単位で行列を適用して線形変換を行う行拡散演算のいずれかの行列演算を実行する請求項３に記載の暗号処理装置。
【請求項７】
前記ラウンド演算を実行するラウンド演算部は、線形変換処理を実行する線形変換処理部を有し、
前記暗号処理部は、
１つおきのラウンドの線形変換処理部に繋がる排他的論理和部に前記定数を入力して、変換対象データ、またはラウンド鍵との排他的論理和を行う請求項１に記載の暗号処理装置。
【請求項８】
前記暗号処理部は、
平文Ｐを入力データとして前記ラウンド演算を繰り返して出力データとしての暗号文Ｃを出力し、
前記暗号文Ｃを入力データとして、前記ラウンド演算の実行シーケンスを逆順に設定したデータ変換処理により出力データとして前記平文Ｐを生成可能なインボリューション性を有する構成である請求項１に記載の暗号処理装置。
【請求項９】
前記鍵スケジュール部は、
平文Ｐから暗号文Ｃを生成する場合の鍵供給シーケンスと、
暗号文Ｃから平文Ｐを生成する場合の鍵供給シーケンスが一致するインボリューション性を有する鍵供給処理を行なう構成である請求項１に記載の暗号処理装置。
【請求項１０】
前記鍵スケジュール部は、
前記暗号処理部に対する鍵供給処理に際して、供給鍵の一部に定数による演算を施し、演算結果である鍵データを前記暗号処理部に出力する請求項１に記載の暗号処理装置。
【請求項１１】
前記暗号処理部が繰り返し実行するラウンド演算は、線形変換部による線形変換処理を含む演算であり、

前記線形変換部は、ラウンド遷移に応じて線形変換態様を変更する請求項１に記載の暗号処理装置。

【請求項１２】

前記ラウンド演算は非線形変換処理を含み、

前記非線形変換処理を実行するＳボックスは、入力値から得られる出力値を、再入力することで前記入力値が得られるインボリューション性を有する構成である請求項１に記載の暗号処理装置。

【請求項１３】

暗号処理装置において実行する暗号処理方法であり、

前記暗号処理装置は、

入力データに対するラウンド演算を実行して出力データを生成する暗号処理部と、

前記暗号処理部におけるラウンド演算において適用するラウンド鍵を前記暗号処理部に出力する鍵スケジュール部を有し、

前記暗号処理部は、

データ変換関数Ｅと、前記データ変換関数Ｅの逆関数E^{-1}をシーケンシャルに実行するインボリューション性を有し、

前記関数Ｅ、または、前記逆関数E^{-1}のいずれか一方において、１回以上の定数を適用したラウンド演算を実行する暗号処理方法。

【発明の詳細な説明】

【技術分野】

【０００１】

本開示は、暗号処理装置、および暗号処理方法、並びにプログラムに関する。さらに詳細には、共通鍵系暗号を実行する暗号処理装置、および暗号処理方法、並びにプログラムに関する。

【背景技術】

【０００２】

情報化社会が発展すると共に、扱う情報を安全に守るための情報セキュリティ技術の重要性が増してきている。情報セキュリティ技術の構成要素の一つとして暗号技術があり、現在では様々な製品やシステムで暗号技術が利用されている。

【０００３】

暗号処理アルゴリズムには様々なものがあるが、基本的な技術の一つとして、共通鍵ブロック暗号と呼ばれるものがある。共通鍵ブロック暗号では、暗号化用の鍵と復号用の鍵が共通のものとなっている。暗号化処理、復号処理共に、その共通鍵から複数の鍵を生成し、あるブロック単位、例えば６４ビット、１２８ビット、２５６ビット等のブロックデータ単位でデータ変換処理を繰り返し実行する。

【０００４】

代表的な共通鍵ブロック暗号のアルゴリズムとしては、過去の米国標準であるＤＥＳ（Ｄａｔａ　Ｅｎｃｒｙｐｔｉｏｎ　Ｓｔａｎｄａｒｄ）や現在の米国標準であるＡＥＳ（Ａｄｖａｎｃｅｄ　Ｅｎｃｒｙｐｔｉｏｎ　Ｓｔａｎｄａｒｄ）が知られている。他にも様々な共通鍵ブロック暗号が現在も提案され続けており、２００７年にソニー株式会社が提案したＣＬＥＦＩＡも共通鍵ブロック暗号の一つである。

【０００５】

なお、共通鍵ブロック暗号について開示した従来技術として、例えば特許文献１（特開２０１２－２１５８１３号公報）等がある。

【０００６】

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ル部とによって構成される。鍵スケジュール部は、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。
【０００７】
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として、ＳＰＮ（Ｓｕｂｓｔｉｔｕｔｉｏｎ－Ｐｅｒｍｕｔａｔｉｏｎ Ｎｅｔｗｏｒｋ）構造、Ｆｅｉｓｔｅｌ構造、拡張Ｆｅｉｓｔｅｌ構造等がある。
【０００８】
これらは、いずれも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して平文を暗号文に変換する構造を持つ。
【先行技術文献】
【特許文献】
【０００９】
【特許文献１】特開２０１２－２１５８１３号公報
【発明の概要】
【発明が解決しようとする課題】
【００１０】
例えば暗号アルゴリズムや秘密鍵の解読を試みる攻撃として差分攻撃、線形攻撃等がある。暗号処理装置は、これらの様々な攻撃に対する耐性や、高速処理、あるいは小型化などが求められている。
【００１１】
本開示は、例えば上述の状況に鑑みてなされたものであり、安全性、高速性、あるいは小型化等、暗号処理装置に要求される様々な要素の向上を実現する暗号処理装置、および暗号処理方法、並びに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】
本開示の第１の側面は、
入力データに対するラウンド演算を実行して出力データを生成する暗号処理部と、
前記暗号処理部におけるラウンド演算において適用するラウンド鍵を前記暗号処理部に出力する鍵スケジュール部を有し、
前記暗号処理部は、
データ変換関数Ｅと、前記データ変換関数Ｅの逆関数E^{-1}をシーケンシャルに実行するインボリューション性を有し、
前記関数Ｅ、または、前記逆関数E^{-1}のいずれか一方において１回以上の定数を適用したラウンド演算を実行する暗号処理装置にある。
【００１３】
さらに、本開示の第２の側面は、
入力データに対するラウンド演算を実行して出力データを生成する暗号処理部と、
前記暗号処理部におけるラウンド演算において適用するラウンド鍵を前記暗号処理部に出力する鍵スケジュール部を有し、
前記暗号処理部は、
データ変換関数Ｅと、前記データ変換関数Ｅの逆関数E^{-1}をシーケンシャルに実行するインボリューション性を有し、
前記関数Ｅ、および前記逆関数E^{-1}の双方において１回以上の定数を適用したラウンド演算を実行する構成を有し、前記関数Ｅ、および前記逆関数E^{-1}の非対応位置に定数適用位置が設定されている暗号処理装置にある。
【００１４】
さらに、本開示の第３の側面は、
暗号処理装置において実行する暗号処理方法であり、
前記暗号処理装置は、
入力データに対するラウンド演算を実行して出力データを生成する暗号処理部と、

前記暗号処理部におけるラウンド演算において適用するラウンド鍵を前記暗号処理部に出力する鍵スケジュール部を有し、
前記暗号処理部は、
データ変換関数Ｅと、前記データ変換関数Ｅの逆関数$Ｅ^{-1}$をシーケンシャルに実行するインボリューション性を有し、
前記関数Ｅ、または、前記逆関数$Ｅ^{-1}$のいずれか一方において、１回以上の定数を適用したラウンド演算を実行する暗号処理方法にある。

【００１５】

さらに、本開示の第４の側面は、
暗号処理装置において実行する暗号処理方法であり、
前記暗号処理装置は、
入力データに対するラウンド演算を実行して出力データを生成する暗号処理部と、
前記暗号処理部におけるラウンド演算において適用するラウンド鍵を前記暗号処理部に出力する鍵スケジュール部を有し、
前記暗号処理部は、
データ変換関数Ｅと、前記データ変換関数Ｅの逆関数$Ｅ^{-1}$をシーケンシャルに実行するインボリューション性を有し、前記関数Ｅ、および前記逆関数$Ｅ^{-1}$の非対応位置に定数適用位置が設定された構成であり、
前記関数Ｅ、および前記逆関数$Ｅ^{-1}$の双方において１回以上の定数を適用したラウンド演算を実行する暗号処理方法にある。

【００１６】

さらに、本開示の第５の側面は、
暗号処理装置において暗号処理を実行させるプログラムであり、
前記暗号処理装置は、
入力データに対するラウンド演算を実行して出力データを生成する暗号処理部と、
前記暗号処理部におけるラウンド演算において適用するラウンド鍵を前記暗号処理部に出力する鍵スケジュール部を有し、
前記暗号処理部は、
データ変換関数Ｅと、前記データ変換関数Ｅの逆関数$Ｅ^{-1}$をシーケンシャルに実行するインボリューション性を有し、
前記プログラムは、前記暗号処理部に、前記関数Ｅ、または、前記逆関数$Ｅ^{-1}$のいずれか一方において、１回以上の定数を適用したラウンド演算を実行させるプログラムにある。

【００１７】

さらに、本開示の第６の側面は、
暗号処理装置において暗号処理を実行させるプログラムであり、
前記暗号処理装置は、
入力データに対するラウンド演算を実行して出力データを生成する暗号処理部と、
前記暗号処理部におけるラウンド演算において適用するラウンド鍵を前記暗号処理部に出力する鍵スケジュール部を有し、
前記暗号処理部は、
データ変換関数Ｅと、前記データ変換関数Ｅの逆関数$Ｅ^{-1}$をシーケンシャルに実行するインボリューション性を有し、前記関数Ｅ、および前記逆関数$Ｅ^{-1}$の非対応位置に定数適用位置が設定された構成であり、
前記プログラムは、前記暗号処理部に前記関数Ｅ、および前記逆関数$Ｅ^{-1}$の双方において１回以上の定数を適用したラウンド演算を実行させるプログラムにある。

【００１８】

なお、本開示のプログラムは、例えば、様々なプログラム・コードを実行可能な情報処理装置やコンピュータ・システムに対して例えば記憶媒体によって提供されるプログラムである。このようなプログラムを情報処理装置やコンピュータ・システム上のプログラム

実行部で実行することでプログラムに応じた処理が実現される。

【００１９】

本開示のさらに他の目的、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】

【００２０】

本開示の一実施例の構成によれば、各種の攻撃に対する耐性の高い安全性の優れた暗号処理が実現される。

具体的には、入力データに対するラウンド演算を繰り返して出力データを生成する暗号処理部と、暗号処理部におけるラウンド演算において適用するラウンド鍵を暗号処理部に出力する鍵スケジュール部を有し、暗号処理部は、データ変換関数Ｅと、前記データ変換関数Ｅの逆関数E^{-1}をシーケンシャルに実行するインボリューション性を有し、関数Ｅ、または逆関数E^{-1}のいずれか一方のみにおいて、１回以上の定数を適用したラウンド演算を実行する。定数は、定数を入力する排他的論理和部の隣接位置の線形変換処理部において適用する線形変換行列との行列演算結果であるステートの構成要素が全て非ゼロとなる条件を満たすステートとして構成される。

本構成により各種の攻撃に対する耐性を向上させた安全性の高い暗号処理構成が実現される。

なお、本明細書に記載された効果はあくまで例示であって限定されるものではなく、また付加的な効果があってもよい。

【図面の簡単な説明】

【００２１】

【図１】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を説明する図である。

【図２】図１に示す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に対応する復号アルゴリズムを説明する図である。

【図３】鍵スケジュール部と暗号処理部の関係について説明する図である。

【図４】暗号処理部の構成例について説明する図である。

【図５】ＳＰＮ構造のラウンド関数の例について説明する図である。

【図６】Ｆｅｉｓｔｅｌ構造のラウンド関数の一例について説明する図である。

【図７】拡張Ｆｅｉｓｔｅｌ構造の一例について説明する図である。

【図８】拡張Ｆｅｉｓｔｅｌ構造の一例について説明する図である。

【図９】非線形変換部の構成例について説明する図である。

【図１０】線形変換部の構成例について説明する図である。

【図１１】ステート（ステート表現データ）に対するデータ変換処理例について説明する図である。

【図１２】ステートに対するデータ変換処理例について説明する図である。

【図１３】ステートに対するデータ変換処理例について説明する図である。

【図１４】ステートに対するデータ変換処理例について説明する図である。

【図１５】ステートに対する列拡散演算処理について説明する図である。

【図１６】ステートに対する列拡散演算処理について説明する図である。

【図１７】ステートに対する行拡散演算処理について説明する図である。

【図１８】ステートに対する行拡散演算処理について説明する図である。

【図１９】本開示の一実施例に係る暗号処理装置の構成例について説明する図である。

【図２０】本開示の一実施例に係る暗号処理装置の構成例について説明する図である。

【図２１】暗号処理部において実行するデータ変換処理例について説明する図である。

【図２２】暗号処理部の非線形変換部と線形変換部の構成と処理について説明する図である。

【図２３】暗号処理部の線形変換部の構成と処理について説明する図である。
【図２４】線形変換処理に適用する行列について説明する図である。
【図２５】線形変換部Ｐ１の実行する列拡散演算について説明する図である。
【図２６】線形変換部Ｐ２の実行する行拡散演算について説明する図である。
【図２７】線形変換部Ｐ３の実行する行拡散演算について説明する図である。
【図２８】暗号処理部の線形変換部の構成と処理について説明する図である。
【図２９】暗号処理部の線形変換部を同一の線形変換処理とした場合の構成について説明する図である。
【図３０】暗号処理部の線形変換部を異なる線形変換処理実行構成とした場合しと、同一の線形変換処理を実行する構成とした場合のアクティブＳボックスの数の比較データについて説明する図である。
【図３１】暗号処理部の線形変換部を異なる線形変換処理実行構成とした場合しと、同一の線形変換処理を実行する構成とした場合のアクティブＳボックスの数の比較データについて説明する図である。
【図３２】鍵スケジュール部の構成と処理について説明する図である。
【図３３】鍵スケジュール部の鍵変換部の構成と処理について説明する図である。
【図３４】鍵スケジュール部の鍵変換部の構成と処理について説明する図である。
【図３５】鍵スケジュール部の鍵変換部の変換処理にるデータ拡散処理について説明する図である。
【図３６】鍵変換処理の実行構成と実行しない構成との対比について説明する図である。
【図３７】鍵変換処理の実行構成と実行しない構成との対比について説明する図である。
【図３８】鍵変換処理の実行構成と実行しない構成との対比について説明する図である。
【図３９】鍵スケジュール部の構成と処理について説明する図である。
【図４０】鍵変換関数がインボリューション性を有していない場合の鍵スケジュール部の構成と処理について説明する図である。
【図４１】鍵変換関数がインボリューション性を有している場合の鍵スケジュール部の構成と処理について説明する図である。
【図４２】鍵スケジュール部の構成と処理について説明する図である。
【図４３】鍵スケジュール部の構成と処理について説明する図である。
【図４４】鍵変換関数Ｇがフルディフュージョン性を有する場合の暗号処理構成について説明する図である。
【図４５】鍵変換を実行しない場合の構成と処理について説明する図である。
【図４６】１６ビット置換処理について説明する図である。
【図４７】フルディフューション４ビット関数と１６ビット置換関数を適用した鍵変換処理例について説明する図である。
【図４８】フルディフューション４ビット関数と１６ビット置換関数を適用した鍵変換処理例について説明する図である。
【図４９】フルディフューション４ビット関数と１６ビット置換関数を適用した鍵変換処理例について説明する図である。
【図５０】フルディフューション４ビット関数と１６ビット置換関数を適用した鍵変換処理例について説明する図である。
【図５１】フルディフューション４ビット関数と１６ビット置換関数を適用した鍵変換処理例について説明する図である。
【図５２】分割鍵に対する置換関数Ｇ１，Ｇ２の設定例について説明する図である。
【図５３】暗号処理部に対する定数入力構成例について説明する図である。
【図５４】インボリーション性を有する暗号処理部の構成例について説明する図である。
【図５５】インボリーション性を有する暗号処理部の問題点について説明する図である。
【図５６】暗号処理部に対する定数入力構成例について説明する図である。
【図５７】暗号処理部に対する定数入力構成例について説明する図である。
【図５８】暗号処理部に対する定数入力構成例について説明する図である。

【図５９】暗号処理部に対する定数入力構成例について説明する図である。
【図６０】アクティブＳボックスに基づく安全性評価処理について説明する図である。
【図６１】アクティブＳボックスに基づく安全性評価処理について説明する図である。
【図６２】アクティブＳボックスに基づく安全性評価処理について説明する図である。
【図６３】暗号処理部に対する定数入力構成例について説明する図である。
【図６４】暗号処理部の非線形変換部のＳボックス（Ｓ－ｂｏｘ）の構成例について説明する図である。
【図６５】暗号処理部の非線形変換部のＳボックス（Ｓ－ｂｏｘ）の構成例について説明する図である。
【図６６】暗号処理部の非線形変換部のＳボックス（Ｓ－ｂｏｘ）の構成例について説明する図である。
【図６７】暗号処理部の非線形変換部のＳボックス（Ｓ－ｂｏｘ）の線形変換層の構成例について説明する図である。
【図６８】暗号処理部の非線形変換部のＳボックス（Ｓ－ｂｏｘ）の構成例について説明する図である。
【図６９】暗号処理部の非線形変換部のＳボックス（Ｓ－ｂｏｘ）の構成例について説明する図である。
【図７０】暗号処理部の非線形変換部のＳボックス（Ｓ－ｂｏｘ）の構成例について説明する図である。
【図７１】暗号処理部の非線形変換部のＳボックス（Ｓ－ｂｏｘ）の構成例について説明する図である。
【図７２】暗号処理装置の一構成例について説明する図である。
【図７３】暗号処理装置としてのＩＣモジュール７００の構成例を示す図である。
【図７４】暗号処理実行機能を有するスマートフォンの構成例を示す図である。
【発明を実施するための形態】
【００２２】
以下、図面を参照しながら本開示に係る暗号処理装置、および暗号処理方法、並びに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う。
１．共通鍵ブロック暗号の概要
２．共通鍵ブロック暗号における安全性の指標について
３．安全性を高めた共通鍵暗号処理の全体構成概要について
４．暗号処理部の線形変換部の構成と処理について
５．鍵スケジュール部の構成と処理について
５－１．鍵スケジュール部の構成と処理の説明
５－２．鍵スケジュール部のフルディフュージョン性に基づく効果について
５－３．鍵変換部のインボリューション性に基づく効果について
５－３－ａ．アンロールド（Ｕｎｒｏｌｌｅｄ）実装における効果について
５－３－ｂ．ラウンド実装における効果について
５－４．本開示の鍵スケジュール部の構成と効果のまとめ
５－５．鍵スケジュール部のその他の構成例について
５－６．フルディフュージョン性を持つ鍵変換部を有する構成例について
６．定数入力による安全性向上を実現する構成について
６－１．定数入力による安全性向上を実現した従来構成とその問題点について
６－２．安全性の高い定数入力構成を持つ暗号処理装置の構成について
６－３．定数挿入位置のバリエーションについて
７．非線形変換部に適用するＳボックス（Ｓ－ｂｏｘ）の具体的構成例について
８．暗号処理装置の具体例について
９．暗号処理装置の実装例について
１０．本開示の構成のまとめ
【００２３】

［１．共通鍵ブロック暗号の概要］

まず、共通鍵ブロック暗号の概要について説明する。

（１－１．共通鍵ブロック暗号）

ここでは共通鍵ブロック暗号（以下ではブロック暗号と呼ぶ場合がある）は以下に定義するものを指すものとする。

ブロック暗号は入力として平文Ｐと鍵Ｋを取り、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、例えばブロックサイズ＝ｎとする。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、あらかじめひと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。

【００２４】

鍵のビット長を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、ｋ＝１２８またはｋ＝１９２またはｋ＝２５６の鍵サイズに対応するという構成もありうるものとする。

平文Ｐ：ｎビット

暗号文Ｃ：ｎビット

鍵Ｋ：ｋビット

【００２５】

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。

暗号化アルゴリズムＥに対応する復号アルゴリズムＤは暗号化アルゴリズムＥの逆関数E^{-1}と定義でき、入力として暗号文Ｃと鍵Ｋを受け取り，平文Ｐを出力する。図２に図１に示した暗号アルゴリズムＥに対応する復号アルゴリズムＤの図を示す。

【００２６】

（１－２．内部構成）

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。ひとつは秘密鍵Ｋを入力とし、ある定められたステップにより暗号処理部の各ラウンドで適用するラウンド鍵を出力する「鍵スケジュール部」と、もうひとつは平文Ｐと鍵スケジュール部からラウンド鍵を入力してデータ変換を行い暗号文Ｃを出力する「暗号処理部」である。

２つの部分の関係は図３に示される。

なお、暗号処理部は、暗号文Ｃを入力して平文Ｐを出力する復号処理も実行可能な構成である場合が多い。この場合も、鍵スケジュール部から供給されるラウンド鍵を適用した復号処理を実行する。

２つの部分の関係は図３に示される。

【００２７】

（１－３．暗号処理部）

以下の実施例において用いる暗号処理部はラウンド関数という処理単位に分割できるものとする。ラウンド関数は入力データに対して所定のデータ変換を施し、変換データを出力する。ラウンド関数に対する入力データは、例えば暗号化途中のｎビットデータである。あるラウンドにおけるラウンド関数の出力が次のラウンドの入力として供給される構成となる。また、ラウンド関数の一構成として、鍵スケジュール部から出力された鍵に基づいて生成されるラウンド鍵との演算構成が含まれる。具体的には暗号化途中のｎビットデータとラウンド鍵との排他的論理和演算が行われる。

またラウンド関数の総数は総ラウンド数と呼ばれ、暗号アルゴリズムごとにあらかじめ定められている値である。

【００２８】

暗号処理部の入力側から見て１ラウンド目の入力データをX_1とし、ｉ番目のラウンド関数に入力されるデータをX_i、ラウンド鍵をRK_iとすると、暗号処理部全体は図４のように示される。

【００２９】
（１－４．ラウンド関数）
ブロック暗号アルゴリズムによってラウンド関数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ここではＳＰＮ（Ｓｕｂｓｔｉｔｕｔｉｏｎ－Ｐｅｒｍｕｔａｔｉｏｎ　Ｎｅｔｗｏｒｋ）構造、Ｆｅｉｓｔｅｌ構造、拡張Ｆｅｉｓｔｅｌ構造を例示する。
【００３０】
（ア）ＳＰＮ（Ｓｕｂｓｔｉｔｕｔｉｏｎ－Ｐｅｒｍｕｔａｔｉｏｎ　Ｎｅｔｗｏｒｋ）構造ラウンド関数
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演算、非線形変換、線形変換処理などが適用される構成。各演算の順番は特に決まっていない。図５にＳＰＮ構造のラウンド関数の例を示す。線形変換部をＰ層（Ｐｅｒｍｕｔａｔｉｏｎ－ｌａｙｅｒ）と呼ぶこともある。
【００３１】
（イ）Ｆｅｉｓｔｅｌ構造
ｎビットの入力データはｎ／２ビットの２つのデータに分割される。うち片方のデータとラウンド鍵を入力として持つ関数（Ｆ関数）が適用され、出力がもう片方のデータに排他的論理和される。そののちデータの左右を入れ替えたものを出力データとする。Ｆ関数の内部構成にもさまざまなタイプのものがあるが、基本的にはＳＰＮ構造同様にラウンド鍵データとの排他的論理和演算、非線形演算、線形変換の組み合わせで実現される。図６にＦｅｉｓｔｅｌ構造のラウンド関数の一例を示す。
【００３２】
（ウ）拡張Ｆｅｉｓｔｅｌ構造
拡張Ｆｅｉｓｔｅｌ構造はＦｅｉｓｔｅｌ構造ではデータ分割数が２であったものを，３以上に分割する形に拡張したものである。分割数をｄとすると、ｄによってさまざまな拡張Ｆｅｉｓｔｅｌ構造を定義することができる。Ｆ関数の入出力のサイズが相対的に小さくなるため、小型実装に向いているとされる。図７にｄ＝４でかつ、ひとつのラウンド内に２つのＦ関数が並列に適用される場合の拡張Ｆｅｉｓｔｅｌ構造の一例を示す。また，図８にｄ＝８でかつ，ひとつのラウンド内に１つのＦ関数が適用される場合の拡張Ｆｅｉｓｔｅｌ構造の一例を示す。
【００３３】
（１－５．非線形変換部）
非線形変換部は、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対象データ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ビットとして、それらをｓビットずつのｍ個のデータに分割して、それぞれに対してｓビット入出力を持つ非線形変換を行う構成である。それらのｓビット単位の非線形変換実行部をＳボックス（Ｓ－ｂｏｘ）と呼ぶ。Ｓボックス（Ｓ－ｂｏｘ）の例を図９に示す。
【００３４】
図９に示す例は、ｍｓビットからなる入力データを、ｍ個のｓビットデータに分割し、各分割データを、各々ｓビットの非線形変換処理を実行するｍ個のＳボックスに入力して、各Ｓボックスの出力を連結してｍｓビットの非線形変換結果を得る構成である。
【００３５】
（１－６．線形変換部）
線形変換部はその性質上、行列として定義することが可能である。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ができる。図１０に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部の例を示す。
【００３６】

（１－７．ステートを用いたデータ表現）

各データ（平文、暗号文、鍵など）を表現する際に、データをｍ行（ｒｏｗ）、ｎ列（ｃｏｌｕｍｎ）のマトリックス型としたｍ×ｎ配列データとして表現することがある。このｍ×ｎ配列によって表現されたデータをステート（ｓｔａｔｅ）、あるいはステート表現データと呼ぶ。

【００３７】

図１１には、入力データをＡ、入力データＡに対するデータ変換後の出力データをＢとして、入力データＡ、出力データＢをそれぞれｍ×ｎ配列を持つステートとして表現した例を示している。

入力データＡは、拡大体ＧＦ（2^{s}）^{mn}の要素であり、

入力データＡ＝（$a_0 a_1 a_2 \cdots a_{mn-2} a_{mn-1}$）である。

なお、a_0はＭＳＢ、a_{mn-1}はＬＳＢ側のビットデータである。

同様に、出力データＢも、拡大体ＧＦ（2^{s}）^{mn}の要素であり、

出力データＢ＝（$b_0 b_1 b_2 \cdots b_{mn-2} b_{mn-1}$）である。

なお、b_0はＭＳＢ、b_{mn-1}はＬＳＢ側のビットデータである。

【００３８】

図に示すように、ｍ×ｎ配列のステートにはｍ×ｎ個の要素が含まれる。

例えば図１１に示すステートＡにはa_0～a_{nm-1}のｍｎ個の要素が含まれる。ステートＢの要素は、b_0～b_{nm-1}のｍｎ個の要素である。

これらのｍｎ個の要素の各々は、それぞれｓ（ｓ＝１以上）ビットデータからなる。具体的には、各要素は、例えば各々４ビットデータ、８ビット（１バイト）データ等のビットデータである。

なお、以下の実施例では、各要素を４ビットデータとした実施例について説明するが、本開示の処理は、４ビット要素データ以外の構成に対しても適用可能である。

【００３９】

図１２に４×４ステートに含まれる１６個の要素の各要素を４ビットデータとした場合の４×４ステートの例を示す。

図１２に示す例も図１１と同様、入力データをＡ、何らかのデータ変換後の出力データをＢとしている。

入力データＡは、拡大体ＧＦ（2^{4}）$^{4\times4}$の要素であり、

入力データＡ＝（$a_0 a_1 a_2 \cdots a_{14} a_{15}$）である。

なお、a_0はＭＳＢ、a_{15}はＬＳＢ側のビットデータである。

同様に、出力データＢも、拡大体ＧＦ（2^{4}）$^{4\times4}$の要素であり、

出力データＢ＝（$b_0 b_1 b_2 \cdots b_{14} b_{15}$）である。

なお、b_0はＭＳＢ、b_{15}はＬＳＢ側のビットデータである。

【００４０】

図１２に示す例は、入力データＡ、出力データＢが各要素が４ビットデータからなる４×４配列を持つステートとして表現した例である。

例えば図１２に示すステートＡにはa_0～a_{15}の１６個の要素が含まれ、これらの各要素は各々が４ビットデータである。

すなわち、６４ビットの入力データＡをステートとして示すと、図１２に示す各要素が４ビットデータからなる４×４配列を持つステートＡとして表現できる。

【００４１】

同様に、図１２に示すステートＢにはb_0～b_{15}の１６個の要素が含まれ、これらの各要素も各々が４ビットデータである。

すなわち、６４ビットの出力データＢをステートとして示すと、図１２に示す各要素が４ビットデータからなる４×４配列を持つステートＢとして表現できる。

【００４２】

（１－８．ステート表現データに対する基本的演算）

次に、ステート（ステート表現データ）に対する演算処理について説明する。

（１）非線形変換処理（Ｓ）
例えば、ステートの各要素４ビット単位で非線形変換を行う複数のＳボックスを適用して非線形変換処理を実行する。
図１３（１）に示すように、入力ステートＡに対する非線形変換処理によって、ステートＢが生成されるとする。
この場合の各要素４ビット単位の出力b_iと入力a_iの関係は、
$b_i = S(a_i)$
ｉ＝０，1，・・・，１５、
である。

【００４３】
（２）線形変換処理（Ｐ）
図１３（２）に示すように、入力ステートＡに対する線形変換処理によって、ステートＢが生成されるとする。
４×４ステートに対する線形変換処理は、例えば４×４ステートの行ごとの４つのデータをベクトルとみなし４×４の行列［Ｍ］による演算を施して値を更新する演算として実行する。これを行拡散演算と呼ぶ。
変換処理後のステート各要素４ビット単位の出力b_iと入力a_iの関係は、
$^t(b_i, b_{i+4}, b_{i+8}, b_{i+12}) = M \times {}^t(a_i, a_{i+4}, a_{i+8}, a_{i+12})$
ｉ＝０，１，２，３、
である。なお、$^t X$は、Ｘの転置行列を示している。線形変換処理としては、このような行拡散演算の他、列拡散演算や、ビット置換など、様々な処理方法がある。
（３）排他的論理和演算（鍵適用演算処理（Ｋ））
図１４に示すように、入力ステートＡに対する排他的論理和演算処理によって、ステートＢが生成されるとする。
例えば鍵スケジュール部から出力されたラウンド鍵Ｋと入力データＡとの排他的論理和演算により、出力データＢを算出する演算である。入力データＡ、ラウンド鍵Ｋ、出力データＢのいずれも、１６個の４ビット要素からなるステート表現された６４ビットデータである。
変換処理後のステート各要素４ビット単位の出力b_iと入力a_i、ラウンド鍵k_iとの関係は、
$b_i = a_i$（ＸＯＲ）k_i
ｉ＝０，1，・・・，１５、
である。なお、上記式において（ＸＯＲ）は排他的論理和演算を示している。

【００４４】
上記の各演算（１）～（３）を所定シーケンスで順次実行する演算の組み合わせによって、１つのラウンド演算が設定される。入力データに対して、ラウンド演算を繰り返し実行し出力データ、例えば暗号化データを生成して出力する。
なお、基本的なラウンド演算は、ラウンド鍵との排他的論理和演算と、線形変換処理と、非線形変換処理を各々１回ずつ実行するものとして設定される。ただし、暗号処理シーケンスにおいて実行されるラウンド演算の中には、変則的なラウンド演算構成も設定可能である。例えばラウンド鍵との排他的論理和演算を複数回含むラウンド演算や、線形変化処理を省略した構成など、他のラウンド演算とは異なるラウンド演算を設定することも可能である。
また、暗号処理シーケンスの最初や最後にラウンド鍵との演算のみを実行する構成も多く利用されている。この処理は鍵ホワイトニング処理と呼ばれ、一般的にはラウンド数としてはカウントしない。

【００４５】
（１－９．ステート表現データに対する列拡散演算）
次にｍ×ｎのマトリクス配列として示されたステート表現データに対する列拡散演算処

理について、図１５、図１６を参照して説明する。

【００４６】

X_0，X_1，・・・，X_{n-1}の各々を、各要素がＧＦ（2^s）上の要素からなるｍ×ｍ行列とする。

図１５に示すように、

ＭＣ［X_0，X_1，・・・，X_{n-1}］

上記演算を、ステート表現データの要素に対して、ステートの各列（０～ｎ－１）の要素と、各列対応の行列X_0，X_1，・・・，X_{n-1}を適用した行列演算を列拡散演算と定義する。

なお、ＭＣは、列（Ｃｏｌｕｍｎ）単位の拡散（Ｍｉｘ）、すなわち（ＭｉｘＣｏｌｕｍｎ）を意味する。

【００４７】

列拡散演算では、ステートの１つの列の要素に対して１つの行列X_kを適用した行列演算を行う。

なお、ステートを構成する複数の列各々に対して適用する行列X_kは、同じ行列とする設定と、異なる行列とする設定とのいずれの設定も可能である。

【００４８】

例えば、入力データであるステートＡに対して、列拡散演算を実行して出力データであるステートＢを算出する演算式は、以下のように表現できる。

Ｂ＝ＭＣ［X_0，X_1，・・・，X_{n-1}］（Ａ）

この列拡散演算処理は、図１５の下段に示すように、以下の式によって示される処理である。

すなわち、上記演算式によって算出されるステートＢの要素は、以下の通りである。

${}^t(b_0\ b_1\ \cdots\ b_{m-1}) = X_0 \times {}^t(a_0\ a_1\ \cdots\ a_{m-1})$、

${}^t(b_m\ b_{m+1}\ \cdots\ b_{2m-1}) = X_1 \times {}^t(a_m\ a_{m+1}\ \cdots\ a_{2m-1})$、

・・・

${}^t(b_{(n-1)m}\ b_{(n-1)m+1}\ \cdots\ b_{nm-1}) = X_{n-1} \times {}^t(a_{(n-1)m}\ a_{(n-1)m+1}\ \cdots\ a_{nm-1})$、

なお、上記式において${}^t(b_1\ b_2\ \cdots\ b_k)$は、$(b_1\ b_2\ \cdots\ b_k)$の転置行列を示す。

実際のステートＡ，Ｂの要素配列に従って上記演算式を示すと、図１５の下段に示すように以下の演算式となる。

【００４９】

【数１】

$$\begin{pmatrix} b_0 \\ b_1 \\ \vdots \\ b_{m-1} \end{pmatrix} = X_0 \cdot \begin{pmatrix} a_0 \\ a_1 \\ \vdots \\ a_{m-1} \end{pmatrix}, \quad \begin{pmatrix} b_m \\ b_{m+1} \\ \vdots \\ b_{2m-1} \end{pmatrix} = X_1 \cdot \begin{pmatrix} a_m \\ a_{m+1} \\ \vdots \\ a_{2m-1} \end{pmatrix}, \quad \cdots,$$

$$\cdots, \quad \begin{pmatrix} b_{(n-1)m} \\ b_{(n-1)m+1} \\ \vdots \\ b_{nm-1} \end{pmatrix} = X_{n-1} \cdot \begin{pmatrix} a_{(n-1)m} \\ a_{(n-1)m+1} \\ \vdots \\ a_{nm-1} \end{pmatrix}$$

【００５０】
図１６は、
入力データＡを６４ビットデータとしてステートＡを１６個の４ビットデータ要素からなるステートＡとし、
出力データＢも６４ビットデータとして、ステートＢを１６個の４ビットデータ要素からなるステートＢとした場合の、
列拡散演算：ＭＣ［Ｘ$_0$，Ｘ$_1$，Ｘ$_2$，Ｘ$_3$］
の適用処理例を示した図である。
【００５１】
すなわち、図１５を参照して説明したと同様、
Ｂ＝ＭＣ［Ｘ$_0$，Ｘ$_1$，Ｘ$_2$，Ｘ$_3$］（Ａ）
上記列拡散演算処理によりステートＢの各要素の算出処理例を示している。
【００５２】
すなわち、上記演算式によって算出されるステートＢの要素は、以下の通りである。
^{t}（ｂ$_0$ｂ$_1$ｂ$_2$ｂ$_3$）＝Ｘ$_0$×^{t}（ａ$_0$ａ$_1$ａ$_2$ａ$_3$）、
^{t}（ｂ$_4$ｂ$_5$ｂ$_6$ｂ$_7$）＝Ｘ$_1$×^{t}（ａ$_4$ａ$_5$ａ$_6$ａ$_7$）、
^{t}（ｂ$_8$ｂ$_9$ｂ$_{10}$ｂ$_{11}$）＝Ｘ$_2$×^{t}（ａ$_8$ａ$_9$ａ$_{10}$ａ$_{11}$）、
^{t}（ｂ$_{12}$ｂ$_{13}$ｂ$_{14}$ｂ$_{15}$）＝Ｘ$_3$×^{t}（ａ$_{12}$ａ$_{13}$ａ$_{14}$ａ$_{15}$）、
実際のステートＡ，Ｂの要素配列に従って上記演算式を示すと、図１６の下段に示すように、以下の演算式となる。
【００５３】

【数２】

$$\begin{pmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \end{pmatrix} = X_0 \cdot \begin{pmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \end{pmatrix}, \begin{pmatrix} b_4 \\ b_5 \\ b_6 \\ b_7 \end{pmatrix} = X_1 \cdot \begin{pmatrix} a_4 \\ a_5 \\ a_6 \\ a_7 \end{pmatrix}, \begin{pmatrix} b_8 \\ b_9 \\ b_{10} \\ b_{11} \end{pmatrix} = X_2 \cdot \begin{pmatrix} a_8 \\ a_9 \\ a_{10} \\ a_{11} \end{pmatrix}, \begin{pmatrix} b_{12} \\ b_{13} \\ b_{14} \\ b_{15} \end{pmatrix} = X_3 \cdot \begin{pmatrix} a_{12} \\ a_{13} \\ a_{14} \\ a_{15} \end{pmatrix}$$

【００５４】

なお、ステートの各列要素に対して同じ行列Ｘを適用した行列演算行う場合、

ＭＣ［Ｘ］

と表現することもある。

つまり、ＭＣ［Ｘ］とＭＣ［Ｘ，Ｘ，・・・，Ｘ］は同じ演算である。

【００５５】

（１－１０．ステート表現データに対する行拡散演算）

次にｍ×ｎのマトリクス配列として示されたステート表現データに対する行拡散演算処理について、図１７、図１８を参照して説明する。

【００５６】

$X_0, X_1, \cdots, X_{m-1}$の各々を、各要素がＧＦ（$2^s$）上の要素からなるｎ×ｎ行列とする。

図１７に示すように、

$MR[X_0, X_1, \cdots, X_{m-1}]$

上記演算を、ステート表現データの要素に対して、ステートの各行（０～ｎ－１）の要素と、各行対応の行列$X_0, X_1, \cdots, X_{m-1}$を適用した行列演算を行拡散演算と定義する。

なお、ＭＲは、行（Ｒｏｗ）単位の拡散（Ｍｉｘ）、すなわち（ＭｉｘＲｏｗ）を意味する。

【００５７】

行拡散演算では、ステートの１つの行の要素に対して１つの行列X_kを適用した行列演算を行う。

なお、ステートを構成する複数の行各々に対して適用する行列X_kは、同じ行列とする設定と、異なる行列とする設定とのいずれの設定も可能である。

【００５８】

例えば、入力データであるステートＡに対して、行拡散演算を実行して出力データであるステートＢを算出する演算式は、以下のように表現できる。

$B = MR[X_0, X_1, \cdots, X_{m-1}]$（Ａ）

この行拡散演算処理は、図１７の下段に示す処理である。

【００５９】

すなわち、上記演算式によって算出されるステートＢの要素は、以下の通りである。

$^t(b_0\ b_m \cdots b_{(n-1)m}) = X_0 \times {}^t(a_0\ a_m \cdots a_{(n-1)m})$、

$^t(b_1\ b_{m+1} \cdots b_{(n-1)m+1}) = X_1 \times {}^t(a_1\ a_{m+1} \cdots a_{(n-1)m+1})$、

・・・

$^t(b_{m-1}\ b_{2m-1} \cdots b_{nm-1}) = X_{m-1} \times {}^t(a_{m-1}\ a_{2m-1} \cdots a_{nm-1})$、

なお、上記式において$^t(b_1\ b_2 \cdots b_k)$は、$(b_1\ b_2 \cdots b_k)$の転置行列であることを示す。

実際のステートＡ，Ｂの要素配列に従って上記演算式を示すと、図１７の下段に示すよ

うに、以下の演算式となる。

【００６０】

【数３】

$$\begin{pmatrix} b_0 \\ b_m \\ \vdots \\ b_{(n-1)m} \end{pmatrix} = X_0 \cdot \begin{pmatrix} a_0 \\ a_m \\ \vdots \\ a_{(n-1)m} \end{pmatrix}, \quad \begin{pmatrix} b_1 \\ b_{m+1} \\ \vdots \\ b_{(n-1)m+1} \end{pmatrix} = X_1 \cdot \begin{pmatrix} a_1 \\ a_{m+1} \\ \vdots \\ a_{(n-1)m+1} \end{pmatrix}, \quad \cdots,$$

$$\cdots, \quad \begin{pmatrix} b_{m-1} \\ b_{2m-1} \\ \vdots \\ b_{nm-1} \end{pmatrix} = X_{m-1} \cdot \begin{pmatrix} a_{m-1} \\ a_{2m-1} \\ \vdots \\ a_{nm-1} \end{pmatrix}$$

【００６１】

図１８は、

入力データＡを６４ビットデータとしてステートＡを１６個の４ビットデータ要素からなるステートＡとし、

出力データＢも６４ビットデータとして、ステートＢを１６個の４ビットデータ要素からなるステートＢとした場合の、

行拡散演算：ＭＲ［Ｘ$_0$，Ｘ$_1$，Ｘ$_2$，Ｘ$_3$］

の適用処理例を示した図である。

【００６２】

すなわち、図１７を参照して説明したと同様、

Ｂ＝ＭＲ［Ｘ$_0$，Ｘ$_1$，Ｘ$_2$，Ｘ$_3$］（Ａ）

上記行拡散演算処理によりステートＢの各要素の算出処理例を示している。

【００６３】

すなわち、上記演算式によって算出されるステートＢの要素は、以下の通りである。

^{t}（ｂ$_0$ｂ$_4$ｂ$_8$ｂ$_{12}$）＝Ｘ$_0$×^{t}（ａ$_0$ａ$_4$ａ$_8$ａ$_{12}$）、

^{t}（ｂ$_1$ｂ$_5$ｂ$_9$ｂ$_{13}$）＝Ｘ$_1$×^{t}（ａ$_1$ａ$_5$ａ$_9$ａ$_{13}$）、

^{t}（ｂ$_2$ｂ$_6$ｂ$_{10}$ｂ$_{14}$）＝Ｘ$_2$×^{t}（ａ$_2$ａ$_6$ａ$_{10}$ａ$_{14}$）、

^{t}（ｂ$_3$ｂ$_7$ｂ$_{11}$ｂ$_{15}$）＝Ｘ$_3$×^{t}（ａ$_3$ａ$_7$ａ$_{11}$ａ$_{15}$）、

実際のステートＡ，Ｂの要素配列に従って上記演算式を示すと、図１８の下段に示すように、以下の演算式となる。

【００６４】

【数４】

$$\begin{pmatrix} b_0 \\ b_4 \\ b_8 \\ b_{12} \end{pmatrix} = X_0 \cdot \begin{pmatrix} a_0 \\ a_4 \\ a_8 \\ a_{12} \end{pmatrix}, \begin{pmatrix} b_1 \\ b_5 \\ b_9 \\ b_{13} \end{pmatrix} = X_1 \cdot \begin{pmatrix} a_1 \\ a_5 \\ a_9 \\ a_{13} \end{pmatrix}, \begin{pmatrix} b_2 \\ b_6 \\ b_{10} \\ b_{14} \end{pmatrix} = X_2 \cdot \begin{pmatrix} a_2 \\ a_6 \\ a_{10} \\ a_{14} \end{pmatrix}, \begin{pmatrix} b_3 \\ b_7 \\ b_{11} \\ b_{15} \end{pmatrix} = X_3 \cdot \begin{pmatrix} a_3 \\ a_7 \\ a_{11} \\ a_{15} \end{pmatrix}$$

【００６５】
なお、ステートの各行要素に対して同じ行列Ｘを適用した行列演算行う場合、
ＭＲ［Ｘ］
と表現することもある。
つまり、ＭＲ［Ｘ］とＭＲ［Ｘ，Ｘ，・・・，Ｘ］は同じ演算である。
【００６６】
（１－１１．インボリューション性について）
平文Ｐから暗号文Ｃを生成する共通鍵ブロック暗号において、各ラウンドに適用するラウンド鍵をＫ１，Ｋ２，・・・，ＫＲとしたとき、
平文Ｐから、暗号文Ｃを算出する暗号化関数Ｅは、以下のように示すことができる。
Ｃ＝Ｅ（Ｐ，Ｋ１，Ｋ２，・・・，ＫＲ）
【００６７】
このとき、暗号文Ｃから平文Ｐを算出する復号関数Ｄは、
Ｐ＝Ｄ（Ｃ，ｋ１，ｋ２，・・・ｋｒ）
となるが、
上記の復号関数Ｄが、以下を満たすとき、
Ｄ（Ｃ，ｋ１，ｋ２，・・・ｋｒ）＝Ｅ（Ｃ，ＫＲ，・・・，Ｋ２，Ｋ１）
すなわち、復号関数Ｄが、暗号化関数Ｅにおけるラウンド鍵の適用順を逆順にするのみで、他は、同一関数を利用できる構成である場合、
この共通鍵ブロック暗号はインボリューション性を持つという。
【００６８】
このように、暗号化関数Ｅを用い，そのラウンド鍵の入力順を変更するのみで，復号関数Ｄが構成できるような共通鍵ブロック暗号はインボリューション性を持つといえる。例えば，Ｆｅｉｓｔｅｌ型共通鍵ブロック暗号は、通常、使用するラウンド鍵の使用順序を逆にするだけで暗号化関数と復号関数が同じ回路で行えることが知られており、インボリューション性を持つといえる。
インボリューション性を持つ共通鍵ブロック暗号は、基本的には暗号化関数を実装するのみで、暗号化機能と復号機能を実現できるため、必要な回路が少なく、軽量化（小型化）が可能であり、実装効率が高くなる。
【００６９】
［２．共通鍵ブロック暗号における安全性の指標について］
共通鍵ブロック暗号に対する攻撃、例えば秘密鍵の解読等を目的とした様々な攻撃が知られている。具体的には、差分攻撃、線形攻撃などがある。
差分攻撃は、暗号装置に対して特定の差分を持つデータを入力し、出力から入力差分を反映するデータを検出して鍵の推定を行なおうとする攻撃である。なお、差分値の伝播確率を差分確率と呼ぶ。
線形攻撃は、入力の特定ビットの排他的論理和と出力の特定ビットの排他的論理和の間の相関を観測して、強い相関関係が見つけることで、鍵の推定を行なおうとする攻撃である。なお、入出力の特定ビットの相関係数を線形確率と呼ぶ。
【００７０】

安全性の高い暗号とは、上記のような各種の攻撃に対する耐性が高い暗号、すなわち暗号処理に適用される秘密情報、例えば鍵等の解読困難性の高い暗号である。

以下、暗号アルゴリズムの安全性指標となる複数のデータについて説明する。

【００７１】

（２－１．分岐数について）

共通鍵ブロック暗号においては、例えば、上述した線形変換、非線形変換、あるいは排他的論理和演算など、様々なデータ変換が実行される。

このようなデータ変換の解読困難性に関する安全性指標として分岐数がある。

例えば、ｎ×ａビットデータからｎ×ｂビットデータへの写像θを、

$\theta : \{0,1\}^{na} \rightarrow \{0,1\}^{nb}$

とする。

上記の写像θに対して分岐数（$Branch_n(\theta)$）を次のように定義する。

【００７２】

$Branch_n(\theta) = min_{\alpha \neq 0}\{hw_n(\alpha) + hw_n(\theta(\alpha))\}$

ただし、

$min_{\alpha \neq 0}\{X\alpha\}$は，$\alpha \neq 0$を満たすすべての$X_\alpha$のうちの最小値を表し、

$hw_n(Y)$はビット列Ｙをｎビット毎に区切って表したときに、ｎビットのデータ全てが０ではない（非ゼロ）要素の数を返す関数とする。

一般的に、分岐数が高いほど解読困難性が高くなり、差分攻撃や線形攻撃に対する耐性が向上すると言われる。

【００７３】

なお、分岐数$Branch_n(\theta)$がｂ＋１であるような写像θは、最適拡散変換（Ｏｐｔｉｍａｌ　Ｄｉｆｆｕｓｉｏｎ　Ｍａｐｐｉｎｇｓ）と呼ばれる。

分岐数の高い線形変換用の行列として、例えば、最適拡散変換を実行するＭＤＳ（Ｍａｘｉｍｕｍ　Ｄｉｓｔａｎｃｅ　Ｓｅｐａｒａｂｌｅ）行列がある。ＭＤＳ行列は、行列を構成する任意の小行列が正則行列となる行列である。なお、正則行列は、逆行列を持つ行列であり、行列をＡとし、逆行列をA^{-1}とすると、

$AA^{-1} = A^{-1}A = E$、

ただしＥは単位行列、

上記式が成立する逆行列A^{-1}を持つ行列Ａが正則行列である。

【００７４】

（２－２．最小差分アクティブＳ－ｂｏｘ数について）

前述したように共通鍵ブロック暗号に設定される非線形変換部には、ｓビット単位の非線形変換を実行するＳボックス（Ｓ－ｂｏｘ）が用いられる。

差分攻撃に対する耐性を図る指標として，差分の接続関係を表現した差分パスに含まれる差分アクティブＳ－ｂｏｘの最小数、すなわち、最小差分アクティブＳ－ｂｏｘ数がある。

【００７５】

差分パスとは、暗号化関数中の鍵データを除くすべてのデータ部分に対して特定の差分値を指定したものである。差分値は自由に決められるものではなく変換処理の前後の差分値は互いに関連しあっている。線形変換処理の前後では、入力差分と出力差分の関係は一対一に決定される。非線形変換の前後では、入力差分と出力差分の関係は一対一にはきまらないが、確率という概念が導入される。ある入力差分と出力差分に対する確率は事前に計算することができるものとする。すべての出力に対する確率をすべて足し合わせると１となっている。

【００７６】

一般的な暗号（ブロック暗号など）において、非線形変換はＳ－ｂｏｘによる処理の部分のみである。したがって、この場合、０以外の確率をもつ差分パスとは、平文（入力）に対する差分値から始まって暗号文（出力）の差分値までに至る差分データの集合であり、すべてのＳ－ｂｏｘの前後で与えられる差分値は０以外の確率をもつものである。０以

外の確率をもつある差分パスのＳ－ｂｏｘに入力される差分値が０でないものを差分アクティブＳ－ｂｏｘと呼ぶものとする。０以外の確率を持つすべての差分パスの差分アクティブＳ－ｂｏｘ数のうちで最も少ない数を最小差分アクティブＳ－ｂｏｘ数とよび、この数値が差分攻撃に対する安全性指標としてよく知られている。

【００７７】

一般的には、最小差分アクティブＳ－ｂｏｘ数が十分大きくなることを保証することで差分攻撃に対する安全性を示すことが可能であり、少ないラウンド関数の繰り返し回数でより多くの最小差分アクティブＳ－ｂｏｘ数を保証できるような暗号は、より性能の高い暗号と考えることができる。なお、すべての差分値が０であるような差分パスは、確率が１となり攻撃として意味をなさない。

【００７８】

（２－３．最小線形アクティブＳボックス（Ｓ－ｂｏｘ）数について）

線形攻撃に対する耐性を示す指標の一つとして、線形マスクの接続関係を表現した線形パスに含まれる線形アクティブＳボックス（Ｓ－ｂｏｘ）の最小数が挙げられる。

なお、線形パスは、線形近似と呼ばれることも多いが、差分と対応させるためここではパスという言葉を用いる。

線形パスとは、暗号化関数中の鍵データを除くすべてのデータ部分に対して特定の線形マスク値を指定したものである。線形マスク値は自由に決められるものではなく変換処理の前後の線形マスク値は互いに関連しあっている。線形変換処理の前後では、入力線形マスク値と出力線形マスク値の関係は一対一に決定される。非線形変換の前後では、入力線形マスク値と出力線形マスク値の関係は一対一には決まらないが、確率という概念が導入される。入力線形マスク値に対して、出力されうる一つ以上の線形マスク値の集合が存在し、それぞれが出力される確率を事前に計算することができる。すべての出力に対する確率をすべて足し合わせると１となっている。

【００７９】

一般的な暗号（ブロック暗号など）では、非線形変換はＳボックス（Ｓ－ｂｏｘ）による処理の部分のみである。したがって、この場合、０以外の確率を持つ線形パスとは平文（入力）に対する線形マスク値から始まって暗号文（出力）の線形マスク値までに至る線形マスク値データの集合であり、すべてのＳボックス（Ｓ－ｂｏｘ）の前後で与えられる線形マスク値は０以外の確率を持つものである。０以外の確率をもつある線形パスのＳボックス（Ｓ－ｂｏｘ）に入力される線形マスク値が０でないものを線形アクティブＳボックス（Ｓ－ｂｏｘ）と呼ぶものとする。０以外の確率を持つすべての線形パスのアクティブＳボックス（Ｓ－ｂｏｘ）数のうちで最も少ない数を最小線形アクティブＳボックス（Ｓ－ｂｏｘ）数とよび、この数値が線形攻撃に対する安全性指標としてよく知られている。

【００８０】

一般的には、最小線形アクティブＳボックス（Ｓ－ｂｏｘ）数が十分大きくなることを保証することで線形攻撃に対する安全性を示すことが可能であり、少ないラウンド関数の繰り返し回数でより多くの最小線形アクティブＳボックス（Ｓ－ｂｏｘ）数を保証できるような暗号は、より性能の高い暗号と考えることができる。なお、すべての線形マスク値が０であるような線形パスは、確率が１となり攻撃として意味をなさない。

【００８１】

［３．安全性を高めた共通鍵暗号処理の全体構成概要について］

次に、安全性を高めた本開示の共通鍵暗号処理装置の構成と処理について説明する。

以下に説明する本開示の暗号処理装置は、共通鍵ブロック暗号（ブロック暗号）を実行する装置であり、ＳＰＮ（Ｓｕｂｓｔｉｔｕｔｉｏｎ－Ｐｅｒｍｕｔａｔｉｏｎ Ｎｅｔｗｏｒｋ）構造ラウンド関数を有する装置である。

【００８２】

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演算、非線形変換、線形変換処理を、複数ラウンド繰り返し実行する構成である。

本開示の共通鍵暗号処理装置の１つの具体的構成例を、図１９に示す。
図１９に示すように、暗号処理装置１００は、鍵スケジュール部１１０と、暗号処理部１２０を有する。
【００８３】
鍵スケジュール部１１０は、秘密鍵Ｋを入力とし、所定の鍵生成アルゴリズムに従って、暗号処理部１２０の各ラウンドで適用するラウンド鍵を出力する。暗号処理部１２０は、鍵スケジュール部１１０からラウンド鍵を入力して平文Ｐのデータ変換を行い暗号文Ｃを出力する。
なお、暗号処理部１２０は、暗号文Ｃを入力して平文Ｐを出力する復号処理も実行可能である。復号処理を実行する際は、鍵スケジュール部１１０から供給されるラウンド鍵を暗号化処理とは逆順に適用した処理を実行する。
【００８４】
暗号処理部１２０は、
入力データと、ラウンド鍵との排他的論理和演算を実行する排他的論理和部１２１、
入力データに対して非線形変換処理を実行する非線形変換部１２２、
入力データに対して線形変換処理を実行する線形変換部１２３、
を有する。
【００８５】
図に示すように、本開示の暗号処理装置１００の暗号処理部１２０は、排他的論理和部１２１、非線形変換部１２２、線形変換部１２３、これら３つの異なるデータ変換処理を繰り返し実行する構成を有する。
【００８６】
なお、入力データとしての平文Ｐ、出力データとしての暗号文Ｃは、図２０に示すように、前述したステート表現データであり、各要素の各々を４ビットデータとして、４×４の１６要素によって構成される６４ビットデータである。
なお、鍵スケジュール部１１０から入力するラウンド鍵も、１６個の４ビットデータ要素からなるステート表現された６４ビットデータである。
【００８７】
暗号処理部１２０では、図２１に示すように、以下の３種類のデータ変換処理が繰り返し実行される。
（ａ）排他的論理和演算処理
（ｂ）非線形変換処理
（ｃ）線形変換処理
これらの各処理をステートに対する処理として実行する。ステートに対するこれらの処理については、図１３、図１４を参照して説明した通りである。
【００８８】
暗号処理部１２０の非線形変換部において実行する非線形変換処理は、例えば図２２（１）に示すように、複数のＳボックス（Ｓ－ｂｏｘ）を利用して実行される。
各Ｓボックスは例えば、４ビット入出力構成を持つ非線形変換部であり、１６個のＳボックスによる並列処理によって４×１６＝６４ビットの非線形変換処理を実行する。
また、暗号処理部１２０の線形変換部において実行する線形変換処理は、例えば図２２（２）に示すように、行列演算処理として実行する。
【００８９】
［４．暗号処理部の線形変換部の構成と処理について］
図１９を参照して説明したように、本開示の暗号処理装置１００の暗号処理部１２０は、ラウンド鍵との排他的論理和演算、非線形変換、線形変換処理を複数ラウンド繰り返し実行する構成を持つ。
【００９０】
本開示の暗号処理装置の特徴の１つは、各ラウンドにおいて実行する線形変換処理をラウンド毎に異なる処理として実行する構成としたことにある。

以下、本開示の暗号処理装置の実行する線形変換処理の詳細について説明する。

【００９１】

図２３は、本開示の暗号処理装置の暗号処理部に構成する異なる線形変換部の構成例について説明する図である。

なお、図２３の構成図は、排他的論理和部は省略して示した構成図である。

【００９２】

図２３に示す例では、３つの異なる線形変換処理を実行する線形変換部を有する構成としている。すなわち、

線形変換部Ｐ１，２０１、

線形変換部Ｐ２，２０２、

線形変換部Ｐ３，２０３、

これらの３つの異なる線形変換部を有し、これら３種類の異なる線形変換処理のいずれかを各ラウンドにおいて実行する構成とし、連続ラウンドでは同じ線形変更処理を続けることなく、異なる線形変換処理を行なう設定としたことにある。

【００９３】

図２３に示す例では、

平文Ｐの入力側から順に、

線形変換部Ｐ１，

線形変換部Ｐ２，

線形変換部Ｐ１，

線形変換部Ｐ３，

線形変換部Ｐ１，

上記シーケンスで、５回の線形変換処理を行なう。

この５回の線形変換処理において、連続ラウンドでは同じ線形変更処理を続けることなく、ラウンド切り替えに応じて、異なる線形変換処理を実行する。

上記の例では、３種類の異なる線形変換Ｐ１，Ｐ２，Ｐ３を組み合わせて実行することで、連続するラウンドにおいては同じ線形変換を実行しない設定としている。

【００９４】

このように、暗号処理において、ラウンド遷移に応じて線形変換態様を変更することで、最小差分アクティブＳボックス、および最小線形アクティブＳボックスの数を増加させることが可能となり、差分攻撃や線形攻撃に対する耐性を向上させることができる。

【００９５】

３種類の線形変換処理の具体的な処理について、図２４以下を参照して説明する。

線形変換処理Ｐ１～Ｐ３には、図２４（１）に示す４つの異なる行列M_0～M_3を組み合わせて利用する。すなわち、以下に示す４×４の行列（Ｍａｔｒｉｘ）M_0～M_3を組み合わせて構成する。

【００９６】

【数５】

$$M_0=\begin{pmatrix}0&1&1&1\\1&0&1&1\\1&1&0&1\\1&1&1&0\end{pmatrix}\qquad M_1=\begin{pmatrix}1&1&0&1\\1&1&1&0\\0&1&1&1\\1&0&1&1\end{pmatrix}$$

$$M_2=\begin{pmatrix}1&1&1&0\\1&1&0&1\\1&0&1&1\\0&1&1&1\end{pmatrix}\qquad M_3=\begin{pmatrix}1&0&1&1\\0&1&1&1\\1&1&1&0\\1&1&0&1\end{pmatrix}$$

【００９７】

線形変換処理Ｐ１～Ｐ３には、上記の４つの異なる行列M_0～M_3を組み合わせて利用する。

なお、上記の４つの行列は、前述したＭＤＳ（Ｍａｘｉｍｕｍ　Ｄｉｓｔａｎｃｅ　Ｓｅｐａｒａｂｌｅ）行列ではないｎｏｎ－ＭＤＳ行列である。

以下に説明する実施例では、ｎｏｎ－ＭＤＳ行列を利用した線形変換処理例について説明するが、ＭＤＳ行列を適用した構成としてもよい。

【００９８】

図２４（２）は、線形変換処理Ｐ１～Ｐ３の具体的な行列演算の態様を説明する図である。

図２４（２）に示す４×４の矩形は、線形変換処理対象となる各要素４ビットの１６個の要素からなるステートを示している。すなわち６４ビットの４×４ステートである。

この４×４ステートの入力データに対して、行列M_0～M_3を組み合わせて利用した行列演算を実行する。

【００９９】

線形変換処理Ｐ１は、

４×４ステートの入力データの各列の要素に対して、各列単位で、１つの行列M_0を適用した行列演算を行う。

これは、先に、図１５、図１６を参照して説明した列拡散演算（ＭｉｘＣｏｌｕｍ）である。

【０１００】

すなわち、線形変換処理Ｐ１は、

ＭＣ［M_0］

上記式によって示される列拡散演算（ＭＣ）である。

なお、ＭＣ［M_0］は、ステートの各列に対して、同一の行列M_0を適用した行列演算を示す式であり、ステートの各列に対して適用する行列を個別に示した式、

ＭＣ［M_0，M_0，M_0，M_0］

上記式と同じ意味である。

【０１０１】

次に、線形変換処理Ｐ２について説明する。

線形変換処理Ｐ２は、図２４（２）に示すように、４×４ステートの入力データの各行の要素に対して、各行単位で異なる行列を適用した行列演算を行う。上位の第１行から第４行に対して、以下の行列を適用した行列演算を実行する。

第１行：適用行列M_0、

第２行：適用行列M_1、
第３行：適用行列M_2、
第４行：適用行列M_3、
これは、先に、図１７、図１８を参照して説明した行拡散演算（ＭｉｘＲｏｗ）である。

【０１０２】
すなわち、線形変換処理Ｐ２は、
$MR[M_0, M_1, M_2, M_3]$
上記式によって示される行拡散演算（ＭｉｘＲｏｗ）である。

【０１０３】
次に、線形変換処理Ｐ３について説明する。
線形変換処理Ｐ３も、線形変換処理Ｐ２と同様、図２４（２）に示すように、４×４ステートの入力データの各行の要素に対して、各行単位で異なる行列を適用した行列演算を行う。線形変換処理Ｐ３は、線形変換処理Ｐ２とは異なり、上位の第１行から第４行に対して、以下の行列を適用した行列演算を実行する。
第１行：適用行列M_2、
第２行：適用行列M_0、
第３行：適用行列M_1、
第４行：適用行列M_3、
これは、先に、図１７、図１８を参照して説明した行拡散演算（ＭｉｘＲｏｗ）である。

【０１０４】
すなわち、線形変換処理Ｐ３は、
$MR[M_2, M_0, M_1, M_3]$
上記式によって示される行拡散演算（ＭｉｘＲｏｗ）である。

【０１０５】
なお、以下では、線形変換処理Ｐ２と、Ｐ３を区別するため、
線形変換処理Ｐ２を行拡散演算タイプ１（ＭｉｘＲｏｗ１）、
線形変換処理Ｐ３を行拡散演算タイプ２（ＭｉｘＲｏｗ２）、
と呼ぶ。
線形変換処理Ｐ１は、列拡散演算（ＭｉｘＣｏｌｕｍｎ）である。

【０１０６】
これら３つの線形変換処理Ｐ１～Ｐ３の具体的な行列演算の計算処理例について、図２５以下を参照して説明する。

【０１０７】
図２５は、線形変換処理Ｐ１、すなわち列拡散演算（ＭｉｘＣｏｌｕｍｎ）の具体的な計算処理例を説明する図である。
図２５（１）には。線形変換部Ｐ１に対する入出力データの例を示している。
入力Ａはｎビットデータの１６個の要素a_0～a_{15}からなるステートである。
出力Ｂもｎビットデータの１６個の要素b_0～b_{15}からなるステートである。
なお、入出力データの各要素a_i，b_i（ただしｉ＝０～１５）は、０，１のいずれかの値から構成されるｎビットデータである。
なお、本実施例では、ｎ＝４であり、各要素は４ビットデータあり、入力Ａ、出力Ｂとも６４ビットである。

【０１０８】
図２５（２）には、線形変換処理Ｐ１、すなわち列拡散演算（ＭｉｘＣｏｌｕｍｎ）の具体的な計算処理例を示している。
線形変換処理Ｐ１として行われる列拡散演算（ＭｉｘＣｏｌｕｍｎ）は、以下の式に従った行列演算である。

【０１０９】

【数６】

$$\begin{pmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \end{pmatrix} = \begin{pmatrix} 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{pmatrix} \cdot \begin{pmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \end{pmatrix}, \begin{pmatrix} b_4 \\ b_5 \\ b_6 \\ b_7 \end{pmatrix} = \begin{pmatrix} 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{pmatrix} \cdot \begin{pmatrix} a_4 \\ a_5 \\ a_6 \\ a_7 \end{pmatrix}$$

$$\begin{pmatrix} b_8 \\ b_9 \\ b_{10} \\ b_{11} \end{pmatrix} = \begin{pmatrix} 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{pmatrix} \cdot \begin{pmatrix} a_8 \\ a_9 \\ a_{10} \\ a_{11} \end{pmatrix}, \begin{pmatrix} b_{12} \\ b_{13} \\ b_{14} \\ b_{15} \end{pmatrix} = \begin{pmatrix} 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{pmatrix} \cdot \begin{pmatrix} a_{12} \\ a_{13} \\ a_{14} \\ a_{15} \end{pmatrix}$$

【０１１０】

出力Ｂの１６個の要素ｂ$_{0}$～ｂ$_{15}$は、行列Ｍ$_{0}$と、入力Ａの１６個の要素ａ$_{0}$～ａ$_{15}$に基づいて以下の演算によって算出される。

ｂ$_{0}$＝ａ$_{1}$（＋）ａ$_{2}$（＋）ａ$_{3}$，
ｂ$_{1}$＝ａ$_{0}$（＋）ａ$_{2}$（＋）ａ$_{3}$，
ｂ$_{2}$＝ａ$_{0}$（＋）ａ$_{1}$（＋）ａ$_{3}$，
ｂ$_{3}$＝ａ$_{0}$（＋）ａ$_{1}$（＋）ａ$_{2}$，
ｂ$_{4}$＝ａ$_{5}$（＋）ａ$_{6}$（＋）ａ$_{7}$，
ｂ$_{5}$＝ａ$_{4}$（＋）ａ$_{6}$（＋）ａ$_{7}$，
ｂ$_{6}$＝ａ$_{4}$（＋）ａ$_{5}$（＋）ａ$_{7}$，
ｂ$_{7}$＝ａ$_{4}$（＋）ａ$_{5}$（＋）ａ$_{6}$，
ｂ$_{8}$＝ａ$_{9}$（＋）ａ$_{10}$（＋）ａ$_{11}$，
ｂ$_{9}$＝ａ$_{8}$（＋）ａ$_{10}$（＋）ａ$_{11}$，
ｂ$_{10}$＝ａ$_{8}$（＋）ａ$_{9}$（＋）ａ$_{11}$，
ｂ$_{11}$＝ａ$_{8}$（＋）ａ$_{9}$（＋）ａ$_{10}$，
ｂ$_{12}$＝ａ$_{13}$（＋）ａ$_{14}$（＋）ａ$_{15}$，
ｂ$_{13}$＝ａ$_{12}$（＋）ａ$_{14}$（＋）ａ$_{15}$，
ｂ$_{14}$＝ａ$_{12}$（＋）ａ$_{13}$（＋）ａ$_{15}$，
ｂ$_{15}$＝ａ$_{12}$（＋）ａ$_{13}$（＋）ａ$_{14}$

なお、上記式において、演算子（＋）は、排他的論理和演算を意味する。

【０１１１】

線形変換処理Ｐ１として行われる列拡散演算（ＭｉｘＣｏｌｕｍｎ）は、上記演算処理に従って、行列Ｍ$_{0}$と、入力Ａの１６個の要素ａ$_{0}$～ａ$_{15}$に基づいて出力Ｂの１６個の要素ｂ$_{0}$～ｂ$_{15}$を算出する。

【０１１２】

図２６は、線形変換処理Ｐ２、すなわち行拡散演算タイプ１（ＭｉｘＲｏｗ１）の具体的な計算処理例を説明する図である。

図２６（１）には。線形変換部Ｐ２に対する入出力データの例を示している。

入力Ａはｎビットデータの１６個の要素ａ$_{0}$～ａ$_{15}$からなるステートである。

出力Ｂもｎビットデータの１６個の要素ｂ$_{0}$～ｂ$_{15}$からなるステートである。

なお、入出力データの各要素ａ$_{i}$，ｂ$_{i}$（ただしｉ＝０～１５）は、０，１のいずれかの値から構成されるｎビットデータである。

なお、本実施例では、ｎ＝４であり、各要素は４ビットデータあり、入力Ａ、出力Ｂとも６４ビットである。

【０１１３】

図２６（２）には、線形変換処理Ｐ２、すなわち行拡散演算タイプ１（ＭｉｘＲｏｗ１）の具体的な計算処理例を示している。

線形変換処理Ｐ２として行われる行拡散演算タイプ１（ＭｉｘＲｏｗ１）は、以下の式に従った行列演算である。

【０１１４】

【数７】

$$\begin{pmatrix} b_0 \\ b_4 \\ b_8 \\ b_{12} \end{pmatrix} = \begin{pmatrix} 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{pmatrix} \cdot \begin{pmatrix} a_0 \\ a_4 \\ a_8 \\ a_{12} \end{pmatrix}, \begin{pmatrix} b_1 \\ b_5 \\ b_9 \\ b_{13} \end{pmatrix} = \begin{pmatrix} 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 \end{pmatrix} \cdot \begin{pmatrix} a_1 \\ a_5 \\ a_9 \\ a_{13} \end{pmatrix}$$

$$\begin{pmatrix} b_2 \\ b_6 \\ b_{10} \\ b_{14} \end{pmatrix} = \begin{pmatrix} 1 & 1 & 1 & 0 \\ 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 0 & 1 & 1 & 1 \end{pmatrix} \cdot \begin{pmatrix} a_2 \\ a_6 \\ a_{10} \\ a_{14} \end{pmatrix}, \begin{pmatrix} b_3 \\ b_7 \\ b_{11} \\ b_{15} \end{pmatrix} = \begin{pmatrix} 1 & 0 & 1 & 1 \\ 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 \\ 1 & 1 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} a_3 \\ a_7 \\ a_{11} \\ a_{15} \end{pmatrix}$$

【０１１５】

出力Ｂの１６個の要素ｂ$_{0}$～ｂ$_{15}$は、行列Ｍ$_{0}$，Ｍ$_{1}$，Ｍ$_{2}$，Ｍ$_{3}$と、入力Ａの１６個の要素ａ$_{0}$～ａ$_{15}$に基づいて以下の演算によって算出される。

ｂ$_{0}$＝ａ$_{4}$（＋）ａ$_{8}$（＋）ａ$_{12}$，
ｂ$_{1}$＝ａ$_{1}$（＋）ａ$_{5}$（＋）ａ$_{13}$，
ｂ$_{2}$＝ａ$_{2}$（＋）ａ$_{6}$（＋）ａ$_{10}$，
ｂ$_{3}$＝ａ$_{3}$（＋）ａ$_{11}$（＋）ａ$_{15}$，
ｂ$_{4}$＝ａ$_{0}$（＋）ａ$_{8}$（＋）ａ$_{12}$，
ｂ$_{5}$＝ａ$_{1}$（＋）ａ$_{5}$（＋）ａ$_{9}$，
ｂ$_{6}$＝ａ$_{2}$（＋）ａ$_{6}$（＋）ａ$_{14}$，
ｂ$_{7}$＝ａ$_{7}$（＋）ａ$_{11}$（＋）ａ$_{15}$，
ｂ$_{8}$＝ａ$_{0}$（＋）ａ$_{4}$（＋）ａ$_{12}$，
ｂ$_{9}$＝ａ$_{5}$（＋）ａ$_{9}$（＋）ａ$_{13}$，
ｂ$_{10}$＝ａ$_{2}$（＋）ａ$_{10}$（＋）ａ$_{14}$，
ｂ$_{11}$＝ａ$_{3}$（＋）ａ$_{7}$（＋）ａ$_{11}$，
ｂ$_{12}$＝ａ$_{0}$（＋）ａ$_{4}$（＋）ａ$_{8}$，
ｂ$_{13}$＝ａ$_{1}$（＋）ａ$_{9}$（＋）ａ$_{13}$，
ｂ$_{14}$＝ａ$_{6}$（＋）ａ$_{10}$（＋）ａ$_{14}$，
ｂ$_{15}$＝ａ$_{3}$（＋）ａ$_{7}$（＋）ａ$_{15}$

なお、上記式において、演算子（＋）は、排他的論理和演算を意味する。

【０１１６】

線形変換処理Ｐ２として行われる行拡散演算タイプ１（ＭｉｘＲｏｗ１）は、上記演算処理に従って、行列Ｍ$_{0}$，Ｍ$_{1}$，Ｍ$_{2}$，Ｍ$_{3}$と、入力Ａの１６個の要素ａ$_{0}$～ａ$_{15}$に基

づいて出力Ｂの１６個の要素ｂ０～ｂ１５を算出する。

【０１１７】

図２７は、線形変換処理Ｐ３、すなわち行拡散演算タイプ２（ＭｉｘＲｏｗ２）の具体的な計算処理例を説明する図である。

図２７（１）には。線形変換部Ｐ２に対する入出力データの例を示している。

入力Ａはｎビットデータの１６個の要素ａ０～ａ１５からなるステートである。

出力Ｂもｎビットデータの１６個の要素ｂ０～ｂ１５からなるステートである。

なお、入出力データの各要素ａｉ，ｂｉ（ただしｉ＝０～１５）は、０，１のいずれかの値から構成されるｎビットデータである。

なお、本実施例では、ｎ＝４であり、各要素は４ビットデータあり、入力Ａ、出力Ｂとも６４ビットである。

【０１１８】

図２７（２）には、線形変換処理Ｐ２、すなわち行拡散演算タイプ２（ＭｉｘＲｏｗ２）の具体的な計算処理例を示している。

線形変換処理Ｐ３として行われる行拡散演算タイプ２（ＭｉｘＲｏｗ２）は、以下の式に従った行列演算である。

【０１１９】

【数８】

$$\begin{pmatrix} b_0 \\ b_4 \\ b_8 \\ b_{12} \end{pmatrix} = \begin{pmatrix} 1 & 1 & 1 & 0 \\ 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 0 & 1 & 1 & 1 \end{pmatrix} \cdot \begin{pmatrix} a_0 \\ a_4 \\ a_8 \\ a_{12} \end{pmatrix}, \begin{pmatrix} b_1 \\ b_5 \\ b_9 \\ b_{13} \end{pmatrix} = \begin{pmatrix} 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{pmatrix} \cdot \begin{pmatrix} a_1 \\ a_5 \\ a_9 \\ a_{13} \end{pmatrix}$$

$$\begin{pmatrix} b_2 \\ b_6 \\ b_{10} \\ b_{14} \end{pmatrix} = \begin{pmatrix} 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 \end{pmatrix} \cdot \begin{pmatrix} a_2 \\ a_6 \\ a_{10} \\ a_{14} \end{pmatrix}, \begin{pmatrix} b_3 \\ b_7 \\ b_{11} \\ b_{15} \end{pmatrix} = \begin{pmatrix} 1 & 0 & 1 & 1 \\ 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 \\ 1 & 1 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} a_3 \\ a_7 \\ a_{11} \\ a_{15} \end{pmatrix}$$

【０１２０】

出力Ｂの１６個の要素ｂ０～ｂ１５は、行列Ｍ０，Ｍ１，Ｍ２，Ｍ３と、入力Ａの１６個の要素ａ０～ａ１５に基づいて以下の演算によって算出される。

ｂ０＝ａ０（＋）ａ４（＋）ａ８，

ｂ１＝ａ５（＋）ａ９（＋）ａ１３，

ｂ２＝ａ２（＋）ａ６（＋）ａ１４，

ｂ３＝ａ３（＋）ａ１１（＋）ａ１５，

ｂ４＝ａ０（＋）ａ４（＋）ａ１２，

ｂ５＝ａ１（＋）ａ９（＋）ａ１３，

ｂ６＝ａ２（＋）ａ６（＋）ａ１０，

ｂ７＝ａ７（＋）ａ１１（＋）ａ１５，

ｂ８＝ａ０（＋）ａ８（＋）ａ１２，

ｂ９＝ａ１（＋）ａ５（＋）ａ１３，

ｂ１０＝ａ６（＋）ａ１０（＋）ａ１４，

ｂ１１＝ａ３（＋）ａ７（＋）ａ１１，

$b_{12}=a_4$（＋）a_8（＋）a_{12}，
$b_{13}=a_1$（＋）a_5（＋）a_9，
$b_{14}=a_2$（＋）a_{10}（＋）a_{14}，
$b_{15}=a_3$（＋）a_7（＋）a_{15}
なお、上記式において、演算子（＋）は、排他的論理和演算を意味する。

【０１２１】

線形変換処理Ｐ３として行われる行拡散演算タイプ２（ＭｉｘＲｏｗ２）は、上記演算処理に従って、行列M_0，M_1，M_2，M_3と、入力Ａの１６個の要素a_0～a_{15}に基づいて出力Ｂの１６個の要素b_0～b_{15}を算出する。

【０１２２】

このように、１回の暗号処理シーケンスにおいて実行する複数の線形変換処理をラウンド毎に変更することで、最小差分アクティブＳボックス、および最小線形アクティブＳボックスの数を増加させることが可能となり、差分攻撃や線形攻撃に対する耐性を向上させることができる。

【０１２３】

アクティブＳボックスの数についての検証結果について説明する。

本開示の暗号処理装置、すなわち、３種類の異なる線形変換処理Ｐ１～Ｐ３を実行する暗号処理装置と、従来型の単一種類の線形変換処理を繰り返し実行する暗号処理装置におけるアクティブＳボックスの数を検証した。

【０１２４】

本開示の暗号処理装置は、図２８に示すように、暗号処理シーケンスにおいて、３種類の線形変換処理を用い、これらをラウンド毎に切り替えて実行する構成である。

なお、図２８に示す図はラウンド鍵との排他的論理和演算部を省略して示している。

図２８に示す暗号処理装置は、非線形変換部数＝６であるので６ラウンド構成とする。なお、ラウンド数の定義のしかたは、いくつかの方法があるが、ここでは、非線形変換部の数をラウンド数としている。

図２８に示す例ではラウンド鍵RK_1～RK_7の7個のラウンド鍵を適用しているが非線形変換部は6層存在するので６ラウンドの暗号処理装置であるとする。

【０１２５】

図２８に示すように、暗号処理過程において、３つの異なる線形変換処理を少なくとも１回実行する。

線形変換処理Ｐ１は、行列M_0を適用した列拡散演算（ＭｉｘＣｏｌｕｍｎ）である。

線形変換処理Ｐ２は、行列M_0、M_1、M_2、M_3を適用した行拡散演算タイプ１（ＭｉｘＲｏｗ１）である。

線形変換処理Ｐ３は、行列M_0、M_1、M_2、M_3を適用した行拡散演算タイプ２（ＭｉｘＲｏｗ２）である。

【０１２６】

図２８に示す暗号処理装置に対して、従来型の単一の線形変換処理を実行する暗号処理装置の例を図２９に示す。

図２９に示す暗号処理装置も６ラウンド構成であるが、各ラウンドの線形変換処理は同じ線形変換処理を行なう構成である。

図２９の暗号処理装置の全ての線形変換処理部は、図２８の暗号処理装置でも用いた行列M_0のみを利用した線形変換処理Ｐ１、すなわち、行列M_0を用いた列拡散演算を行う設定とした。

【０１２７】

図２８に示す複数の異なる線形変換処理を実行する暗号処理装置と、図２９に示す単一の線形変換処理を実行する従来型の装置について、様々なラウンド数の装置を構成して、最小差分アクティブＳボックス、および最小線形アクティブＳボックスの数を検証した。

【０１２８】

図２８、図２９の構成とも、６４ビットの入力平文Ｐに対する暗号化処理を実行して６

４ビット暗号文Ｃを出力する設定である。
Ｓボックスは、各非線形変換部に設定されており、各Ｓボックスは、先に図２２を参照して説明したように４ビット入出力の非線形変換を実行する構成である。
図２８、図２９の暗号処理装置の各非線形変換部には、４ビット入出力Ｓボックスが１６個設けられており、４×１６＝６４ビットデータの非線形変換を実行する。
図２８、図２９に示す６ラウンド型の暗号処理装置には、６つの非線形変換部が設定されているのまで、Ｓボックスの総数は１６×６＝９６となる。
この総数９６個のＳボックスのうち、全ての入力パターンにおけるアクティブＳボックスの数をカウントし、最小差分アクティブＳボックス、および最小線形アクティブＳボックスの数を検証した。
この検証結果を図３０、図３１に示す。

【０１２９】

図３０に示すように、ラウンド数４～２４の異なるラウンド数の暗号処理装置を構成して最小差分／線形差分アクティブＳボックスの数をカウントした結果である。
ラウンド数＝４では、
従来型の同一線形変換部を繰り返し実行する構成でも、本開示の異なる線形変換処理を実行する構成でも、アクティブＳボックスの数は４であり、同じ値となるが、ラウンド数＝６～２４の場合(８を除く)は、いずれの場合も、本開示の異なる線形変換処理を実行する構成の方がアクティブＳボックスの数が多くなっている。
この結果をグラフとして示したのが図３１に示すグラフである。

【０１３０】

なお、上述した実施例では、入力データを各要素が４ビットの４×４個の要素からなるステートとし、線形変換部が４種類の行列M_0，M_1，M_2，M_3を利用した行列演算による線形変換処理を実行する構成について説明したが、上記処理を一般化した構成として説明すると以下のような設定となる。

【０１３１】

入力データを各要素が１ビット以上のｍ×ｎ個の要素からなるステートとした場合、線形変換部は、ステートの各列要素単位で行列を適用して線形変換を行う列拡散演算と、ステートの各行要素単位で行列を適用して線形変換を行う行拡散演算のいずれかの行列演算をラウンド演算において実行する構成となる。
ここで、線形変換部は、複数種類の行列Ｍ０～Ｍｋ（ｋは１以上の整数）を利用した行列演算による線形変換処理を実行する構成であり、ステートの各列要素単位で行列Ｍ０～Ｍｋから選択した選択行列を特定の順番で各列に適用して線形変換を行う列拡散演算と、ステートの各行要素単位で行列Ｍ０～Ｍｋから選択した選択行列を特定の順番で各行に適用して線形変換を行う行拡散演算をラウンド遷移に応じて切り替えて実行する。

【０１３２】

具体的な線形変換処理構成の一例は、例えば以下の構成となる。
（ａ）ステートの各列要素単位で行列Ｍ０～Ｍｋから選択した選択行列を特定の順番で各列に適用して線形変換を行う列拡散演算と、
（ｂ）ステートの各行要素単位で行列Ｍ０～Ｍｋから選択した選択行列を特定の順番Ａで各行に適用して線形変換を行う行拡散演算タイプ１と、
（ｃ）ステートの各行要素単位で行列Ｍ０～Ｍｋから選択した選択行列を特定の順番Ａと異なる順番Ｂで各行に適用して線形変換を行う行拡散演算タイプ２を、
ラウンド遷移に応じて切り替えて実行する暗号処理装置。

【０１３３】

さらに、上記の構成における列拡散演算と行拡散演算を入れ替えた以下の構成としてもよい。
（ａ）ステートの各行要素単位で行列Ｍ０～Ｍｋから選択した選択行列を特定の順番で各行に適用して線形変換を行う列拡散演算と、
（ｂ）ステートの各列要素単位で行列Ｍ０～Ｍｋから選択した選択行列を特定の順番Ａ

で各列に適用して線形変換を行う行拡散演算タイプ１と、
（ｃ）ステートの各列要素単位で行列Ｍ０～Ｍｋから選択した選択行列を特定の順番Ａと異なる順番Ｂで各列に適用して線形変換を行う行拡散演算タイプ２を、
ラウンド遷移に応じて切り替えて実行する暗号処理装置。
【０１３４】
また、入力データが各要素４ビットの４×４個の要素からなるステートの場合の線形変換処理の具体化構成としては以下の構成が可能である。
線形変換部は、４種類の行列Ｍ０，Ｍ１，Ｍ２，Ｍ３を利用した行列演算による線形変換処理を実行する構成であり、
（ａ）ステートの各列要素単位で行列Ｍ０を適用して線形変換を行う列拡散演算と、
（ｂ）ステートの各行要素単位で行列Ｍ０，Ｍ１，Ｍ２，Ｍ３の順に各行列を適用して線形変換を行う行拡散演算タイプ１と、
（ｃ）ステートの各行要素単位で前記タイプ１と異なる順に各行列を適用して線形変換を行う行拡散演算タイプ２、
上記３種類の行列演算をラウンド遷移に応じて切り替えて実行する暗号処理装置。
【０１３５】
なお、ここで、行拡散演算タイプ１においてステートの各行要素単位で適用する行列と、行拡散演算タイプ２においてステートの各行要素単位で適用する行列との組み合わせは、ステートの任意の２つの行に対してタイプ１で適用する２つの行列とタイプ２で適用する２つの行列の計４つの行列が少なくとも３種類以上の行列によって構成される組み合わせとする。
【０１３６】
例えば、タイプ１において、４×４ステートの各行：第１～４行に適用する行列を、
Ｍ１，Ｍ３，Ｍ０，Ｍ２
としたとき、
タイプ２において４×４ステートの各行：第１～４行に適用する行列を、
Ｍ０，Ｍ２，Ｍ３，Ｍ１
このような設定とする。
上記設定では、４×４ステートの任意の２つの行に対してタイプ１で適用する２つの行列とタイプ２で適用する２つの行列の計４つの行列が少なくとも３種類以上の行列によって構成される組み合わせとなる。
【０１３７】
すなわち、上記設定において、４×４ステートの第１行に適用する行列は、
タイプ１＝Ｍ１、
タイプ２＝Ｍ０、
４×４ステートの第２行に適用する行列は、
タイプ１＝Ｍ３、
タイプ２＝Ｍ２、
このような組み合わせとなり、第１行と第２行に対してタイプ１，２の双方で適用される行列がＭ０～Ｍ３の４種類となる。
上記設定は、その他の任意の２行の組み合わせにおいて、タイプ１で適用する２つの行列とタイプ２で適用する２つの行列の計４つの行列が少なくとも３種類以上の行列によって構成される組み合わせとなる。
【０１３８】
さらに、上記の構成における列拡散演算と行拡散演算を入れ替えた以下の構成としてもよい。
線形変換部は、４種類の行列Ｍ０，Ｍ１，Ｍ２，Ｍ３を利用した行列演算による線形変換処理を実行する構成であり、
（ａ）ステートの各行要素単位で行列Ｍ０を適用して線形変換を行う行拡散演算と、
（ｂ）ステートの各列要素単位で行列Ｍ０，Ｍ１，Ｍ２，Ｍ３の順に各行列を適用して

線形変換を行う列拡散演算タイプ１と、
（ｃ）ステートの各列要素単位で前記タイプ１と異なる順に各行列を適用して線形変換を行う列拡散演算タイプ２、
上記３種類の行列演算をラウンド遷移に応じて切り替えて実行する暗号処理装置。
【０１３９】
なお、この構成においても、行拡散演算タイプ１においてステートの各列要素単位で適用する行列と、行拡散演算タイプ２においてステートの各列要素単位で適用する行列との組み合わせは、ステートの任意の２つの列に対してタイプ１で適用する２つの行列とタイプ２で適用する２つの行列の計４つの行列が少なくとも３種類以上の行列によって構成される組み合わせとする。
【０１４０】
このように１回の暗号処理シーケンスにおいて実行する線形変換処理をラウンド毎に変更することで、最小差分アクティブＳボックス、および最小線形アクティブＳボックスの数を増加させることが可能となり、差分攻撃や線形攻撃に対する耐性を向上させることができる。
【０１４１】
［５．鍵スケジュール部の構成と処理について］
次に、本開示の暗号処理装置における鍵スケジュール部の構成と処理について説明する。
【０１４２】
［５－１．鍵スケジュール部の構成と処理の説明］
先に図１９を参照して説明したように、本開示の暗号処理装置１００は、鍵スケジュール部１１０と、暗号処理部１２０を有する。
【０１４３】
鍵スケジュール部１１０は、例えば秘密鍵Ｋに基づいて所定の鍵生成アルゴリズムに従って、暗号処理部１２０の各ラウンドで適用するラウンド鍵を生成して暗号処理部１２０に出力する。暗号処理部１２０は、鍵スケジュール部１１０からラウンド鍵を入力して平文Ｐのデータ変換を行い暗号文Ｃを出力する。
なお、復号処理に際しても同様の処理が行われる。
以下、このラウンド鍵生成、供給処理を実行する鍵スケジュール部１１０の構成と処理について説明する。
【０１４４】
図３２は、本開示の暗号処理装置における鍵スケジュール部の一構成例を示す図である。
鍵スケジュール部３００は、秘密鍵K_1を格納した記憶部としての鍵供給部（鍵レジスタ）３０１を有する。
鍵スケジュール部３００は、この鍵K_1を、暗号処理部３２０の第１ラウンドの排他的論理和部（ラウンド鍵演算部）３２１に出力する。すなわち鍵K_1が、第１ラウンドのラウンド鍵として利用される。
【０１４５】
さらに、鍵スケジュール部３００は、鍵K_1を鍵変換部３０２ａに入力する。鍵変換部３０２ａは、鍵K_1に対する所定の演算を実行して変換鍵Kd_1を生成する。
さらに、鍵変換部３０２ａの生成した変換鍵Kd_1を暗号処理部３２０の第２ラウンドの排他的論理和部（ラウンド鍵演算部）３２２に出力する。すなわち変換鍵Kd_1が、第２ラウンドのラウンド鍵として利用される。
【０１４６】
さらに、鍵スケジュール部３００は、変換鍵Kd_1を鍵変換部３０２ｂに入力する。鍵変換部３０２ｂは、変換鍵Kd_1に対する所定の演算を実行して鍵K_1を生成する。
この鍵K_1は、変換鍵Kd_1の生成元となった鍵K_1と同じ鍵である。
鍵スケジュール部３００は、鍵変換部３０２ｂの生成した鍵K_1を暗号処理部３２０の

第３ラウンドの排他的論理和部（ラウンド鍵演算部）３２３に出力する。すなわち鍵K_1が、第３ラウンドのラウンド鍵として利用される。

【０１４７】

以下、同様の処理を繰り返し、鍵変換部３０２ｃ～ｆにおいて、鍵K_1と鍵K_{1d}が交互に生成し、生成した鍵を暗号処理部の排他的論理和部３２４～３２７に出力する。

【０１４８】

鍵変換部３０２ａ～３０２ｆはいずれも同じ演算を実行する。すなわち同一の演算処理により、

鍵K_1から変換鍵Kd_1を生成し、

変換鍵Kd_1から鍵K_1を生成する。

【０１４９】

変換関数Ｇ、および逆関数G^{-1}を用いて式で示すと以下の通りとなる。

$Kd_1 = G(K_1)$

$K_1 = G^{-1}(Kd_1)$

である。

なお、

$G = G^{-1}$が成立する。

すなわち、鍵変換部３０２ａ～３０２ｆにおいて鍵変換に適用するデータ変換関数Ｇは、インボリューション性、すなわち、図３３に示すように、順方向関数Ｇと逆方向関数G^{-1}とが同じ関数であるという性質を持つ。

【０１５０】

図３３には、図３２で説明した鍵K_1をベース鍵Ｋ、変換鍵Kd_1を変換鍵Ｋｄとして示している。図３３に示す各鍵は１６個の４ビット要素の４×４ステートとして表現している。すなわち、いずれも６４ビット鍵データである。

【０１５１】

鍵変換部３０２の実行する鍵変換処理例について、図３４を参照して説明する。

図３４は、ベース鍵Ｋから変換鍵Ｋｄを生成する処理を説明する図である。

ベース鍵Ｋから変換鍵Ｋｄを生成する処理は、以下の２つのステップによって構成される。

（Ｓ１）ベース鍵Ｋに対して、中間鍵生成列拡散演算（ＭｉｘＣｏｌｕｍｎ＿ＫＳＦ（））を適用した演算を実行して中間鍵Ｓを生成する。

（Ｓ２）中間鍵Ｓに対して、変換鍵生成列拡散演算（ＭｉｘＲｏｗ＿ＫＳＦ（））を適用した演算を実行して変換鍵Ｋｄを生成する。

【０１５２】

ステップＳ１で実行する列拡散演算（ＭｉｘＣｏｌｕｍｎ）、およびステップＳ２で実行する行拡散演算（ＭｉｘＲｏｗ）は、先に図２４～図２７を参照して説明したと同様の行列適用演算である。

ただし、この鍵変換処理において適用する行列M_Dは、以下に示す行列である。

【０１５３】

【数９】

$$M_D = \begin{pmatrix} 1 & 2 & 4 & 6 \\ 2 & 1 & 6 & 4 \\ 4 & 6 & 1 & 2 \\ 6 & 4 & 2 & 1 \end{pmatrix}$$

【０１５４】
上記に示す行列M_Dは、アダマール（Ｈａｄａｍａｒｄ）ＭＤＳ行列とよばれる行列である。
ＭＤＳ行列は、行列を構成する任意の小行列が正則行列となる行列である。なお、正則行列は、逆行列を持つ行列であり、行列をＡとし、逆行列をA^{-1}とすると、
$AA^{-1}=A^{-1}A=E$、
ただしＥは単位行列、
上記式が成立する逆行列A^{-1}を持つ行列Ａが正則行列である。
前述したように、分岐数Ｂｒａｎｃｈ（θ）がｂ＋１であるような写像θは、最適拡散変換（Ｏｐｔｉｍａｌ　Ｄｉｆｆｕｓｉｏｎ　Ｍａｐｐｉｎｇｓ）と呼ばれ、ＭＤＳ行列は、最適拡散変換を実行する行列である。
【０１５５】
このアダマール（Ｈａｄａｍａｒｄ）ＭＤＳ行列M_Dを適用して、図３４に示すステップＳ１の列拡散演算と、ステップＳ２の行拡散演算を実行する。
ステップＳ１の列拡散演算は、以下の演算式によって示される。
$MC[M_D]=MC[M_D, M_D, M_D, M_D]$
また、ステップＳ２の行拡散演算は、以下の演算式によって示される。
$MR[M_D]=MR[M_D, M_D, M_D, M_D]$
【０１５６】
すなわち、ステップＳ１の列拡散演算は、４ビット要素からなる４×４のステート表現データの４つの全ての列に対して、同一のアダマール（Ｈａｄａｍａｒｄ）ＭＤＳ行列M_Dを適用した行列演算を実行する。
また、ステップＳ２の行拡散演算は、４ビット要素からなる４×４のステート表現データの４つの全ての行に対して、同一のアダマール（Ｈａｄａｍａｒｄ）ＭＤＳ行列M_Dを適用した行列演算を実行する。
【０１５７】
アダマール（Ｈａｄａｍａｒｄ）ＭＤＳ行列M_Dを適用した行列演算のアルゴリズムは以下のように示すことができる。
$M_D()：\{0, 1\}^{16} \rightarrow \{0, 1\}^{16}$
Ｉｎｐｕｔ：$\{x_0, x_1, x_2, x_3\}$, $x_i \in \{0, 1\}^4$
Ｏｕｔｐｕｔ：$\{y_0, y_1, y_2, y_3\}$, $y_i \in \{0, 1\}^4$
Ｏｐｅｒａｔｉｏｎ
・$y_0 = x_0$（＋）２（×）x_1（＋）４（×）x_3（＋）６（×）x_4
・$y_1 = 2$（×）x_0（＋）x_2（＋）６（×）x_3（＋）４（×）x_4
・$y_2 = 4$（×）x_0（＋）６（×）x_2（＋）x_3（＋）２（×）x_4
・$y_3 = 6$（×）x_0（＋）４（×）x_2（＋）２（×）x_3（＋）x_4
ただし、
（＋）は、排他的論理和演算、
（×）は、既約多項式：x^4+x+1によって規定される拡大体ＧＦ（2^4）上の乗算を示す。
【０１５８】
図３４に示すステップＳ１の列拡散演算$MC[M_D]$は、以下の式に従った行列演算である。
【０１５９】

【数１０】

$$\begin{pmatrix} s_0 \\ s_1 \\ s_2 \\ s_3 \end{pmatrix} = \begin{pmatrix} 1 & 2 & 4 & 6 \\ 2 & 1 & 6 & 4 \\ 4 & 6 & 1 & 2 \\ 6 & 4 & 2 & 1 \end{pmatrix} \cdot \begin{pmatrix} k_0 \\ k_1 \\ k_2 \\ k_3 \end{pmatrix}, \ldots, \begin{pmatrix} s_{12} \\ s_{13} \\ s_{14} \\ s_{15} \end{pmatrix} = \begin{pmatrix} 1 & 2 & 4 & 6 \\ 2 & 1 & 6 & 4 \\ 4 & 6 & 1 & 2 \\ 6 & 4 & 2 & 1 \end{pmatrix} \cdot \begin{pmatrix} k_{12} \\ k_{13} \\ k_{14} \\ k_{15} \end{pmatrix}$$

【０１６０】

上記の列拡散演算ＭＣ［M_D］のアルゴリズムは以下のように示すことができる。

ＭｉｘＣｏｌｕｍｎ＿ＫＳＦ（）：｛０，１｝^{64}→｛０，１｝^{64}

Ｉｎｐｕｔ：｛k_0，k_1，・・・，k_{15}｝，$k_i \in \{0,1\}^4$

Ｏｕｔｐｕｔ：｛s_0，s_1，・・・，s_{15}｝，$s_i \in \{0,1\}^4$

Ｏｐｅｒａｔｉｏｎ

・（s_0，s_1，s_2，s_3）＝M_D（k_0，k_1，k_2，k_3）

・（s_4，s_5，s_6，s_7）＝M_D（k_4，k_5，k_6，k_7）

・（s_8，s_9，s_{10}，s_{11}）＝M_D（k_8，k_9，k_{10}，k_{11}）

・（s_{12}，s_{13}，s_{14}，s_{15}）＝M_D（k_{12}，k_{13}，k_{14}，k_{15}）

【０１６１】

また、図３４に示すステップＳ２の行拡散演算ＭＲ［M_D］は、以下の式に従った行列演算である。

【０１６２】

【数１１】

$$\begin{pmatrix} kd_0 \\ kd_4 \\ kd_8 \\ kd_{12} \end{pmatrix} = \begin{pmatrix} 1 & 2 & 4 & 6 \\ 2 & 1 & 6 & 4 \\ 4 & 6 & 1 & 2 \\ 6 & 4 & 2 & 1 \end{pmatrix} \cdot \begin{pmatrix} s_0 \\ s_4 \\ s_8 \\ s_{12} \end{pmatrix}, \ldots, \begin{pmatrix} kd_3 \\ kd_7 \\ kd_{11} \\ kd_{15} \end{pmatrix} = \begin{pmatrix} 1 & 2 & 4 & 6 \\ 2 & 1 & 6 & 4 \\ 4 & 6 & 1 & 2 \\ 6 & 4 & 2 & 1 \end{pmatrix} \cdot \begin{pmatrix} s_3 \\ s_7 \\ s_{11} \\ s_{15} \end{pmatrix}$$

【０１６３】

上記の行拡散演算ＭＲ［M_D］のアルゴリズムは以下のように示すことができる。

ＭｉｘＲｏｗ＿ＫＳＦ（）：｛０，１｝^{64}→｛０，１｝^{64}

Ｉｎｐｕｔ：｛s_0，s_1，・・・，s_{15}｝，$s_i \in \{0,1\}^4$

Ｏｕｔｐｕｔ：｛kd_0，kd_1，・・・，kd_{15}｝，$kd_i \in \{0,1\}^4$

Ｏｐｅｒａｔｉｏｎ

・（kd_0，kd_4，kd_8，kd_{12}）＝M_D（s_0，s_4，s_8，s_{12}）

・（kd_1，kd_5，kd_9，kd_{13}）＝M_D（s_1，s_5，s_9，s_{13}）

・（kd_2，kd_6，kd_{10}，kd_{14}）＝M_D（s_2，s_6，s_{10}，s_{14}）

・（kd_3，kd_7，kd_{11}，kd_{15}）＝M_D（s_3，s_7，s_{11}，s_{15}）

【０１６４】

このように、図３２に示す鍵スケジュール部３００の鍵変換部３０２では、図３４に示

すステップＳ１において、行列M_Dを適用した列拡散演算ＭＣ［M_D］を実行し、ステップＳ２において行列M_Dを適用した行拡散演算ＭＲ［M_D］を実行する。
これらの２つの行列演算を連続して実行することで、ベース鍵Ｋから変換鍵Ｋｄを生成する。

【０１６５】
なお、変換鍵Ｋｄからベース鍵Ｋを生成する場合も、図３４に示す処理と同様の処理を行なう。
すなわち、先に図３３を参照して説明したように、列拡散演算ＭＣ［M_D］と、行拡散演算ＭＲ［M_D］の連続処理からなる関数Ｇは、インボリューション性を持ち、順方向関数Ｇと、逆方向関数G^{-1}が同一であるため、２回繰り返すことで、元の値が算出される。

【０１６６】
さらに、鍵変換部３０２は、行列M_Dを適用した列拡散演算ＭＣ［M_D］と、行列M_Dを適用した行拡散演算ＭＲ［M_D］を実行することで、入力データを構成する４×４のステートの構成要素のすべて、すなわち１６個の構成要素のすべてが、出力データの１６個の構成要素のすべてに影響を及ぼすことが可能となる。
すなわち、入出力ステート全要素間でデータ拡散（ｄｉｆｆｕｓｉｏｎ）がなされている。このようなデータ変換態様を「全拡散（ｆｕｌｌ　ｄｉｆｆｕｓｉｏｎ）変換」、あるいはフルディフュージョン性を有する拡散であると定義する。

【０１６７】
例えば、入力、出力がそれぞれｎビット要素１６個からなるステートであり、入力に対して適用する変換関数ｆとし、
Ｂ＝ｆ（Ａ）
上記式に従って出力ステートＢを算出する設定とする。
入力ステートＡ＝（a_0，a_1，a_2，・・・a_{15}）
出力ステートＢ＝（b_0，b_1，b_2，・・・b_{15}）
である。
ただし、a_i，b_iはステートＡ，Ｂの要素である。
このとき、
出力ステートＢの任意の要素b_iが以下の式によって表現できる場合、関数ｆは、全拡散（ｆｕｌｌ　ｄｉｆｆｕｓｉｏｎ）変換であるという。
$b_i=f(a_0, a_1, a_2, \cdots a_{15})$

【０１６８】
このように、鍵変換部３０２の実行関数Ｇは、以下の２つの性質を有する関数である。
（１）全拡散（ｆｕｌｌ　ｄｉｆｆｕｓｉｏｎ）変換を実現するフルディフュージョン性、
（２）順方向関数Ｇと、逆方向関数G^{-1}が同一であるインボリューション性、
これら２つの性質を持つ。

【０１６９】
［５－２．鍵スケジュール部のフルディフュージョン性に基づく効果について］
鍵変換部３０２の性質であるフルディフュージョン性は、結果として、図３２に示す暗号処理部３２０における変換対象データと鍵との間にもフルディフュージョン性を保証することになる。図３５を参照して説明する。
図３５には、
ベース鍵Ｋ＝（k_0，k_1，k_2，・・・k_{15}）
変換鍵Ｋｄ＝（kd_0，kd_1，kd_2，・・・kd_{15}）
を示している。

【０１７０】
ベース鍵Ｋは、暗号処理部の排他的論理和部３３１に入力されて、入力ステートＡとの排他的論理和演算がなされる。その後、さらに、非線形／線形変換部（Ｓ＆Ｐ）３３２に

おいて非線形変換処理と、線形変換処理がなされる。
さらにその出力に対して、排他的論理和演算部３３３において、変換鍵Ｋｄとの排他的論理和演算が実行される。
排他的論理和演算部３３３の出力をステートＢとする。
【０１７１】
この時、ベース鍵Ｋと変換鍵Ｋｄとの間の関係は、
$Kd_i = f(k_0, k_1, k_2, \cdots k_{15})$
ただし、ｉ＝０～１５
上記関係式が成立する。
すなわち、フルディフュージョン性が保証されている。
上記関係式から、
ステートＢと、ベース鍵Ｋとの間にも、以下の関係式が成立する。
$b_i = f(k_0, k_1, k_2, \cdots k_{15})$
ただし、ｉ＝０～１５
上記関係式が成立する。
すなわち、ベース鍵Ｋと変換データＢとの間でもフルディフュージョン性が保証される。
【０１７２】
この性質は、暗号処理装置の安全性や、実装性能に貢献をもたらす性質であると言える。
具体的には、鍵によるデータ拡散性の向上が実現され、ラウンド数を削減しても高い拡散性能を発揮させることが可能になる。この結果、各種の攻撃に対する耐性を高めることが可能となる。たとえば、鍵依存度を利用した中間値一致攻撃等に基づく鍵解析処理に対する耐性をより向上させることができる。
【０１７３】
上述したように、本開示の構成を適用することで、暗号処理部における変換対象データの拡散性能が向上し、より少ないラウンド数で安全な暗号処理、例えば鍵解析等の各種攻撃に対する耐性の高い暗号処理が実現されることになる。
【０１７４】
図３６には、
（１）各ラウンドにおいて鍵変換を実行せず、同じラウンド鍵を適用する暗号処理構成
（２）本開示の鍵変換を実行して、交互に２種類のラウンド鍵を適用する暗号処理構成
これら２つの暗号処理構成例を示している。
なお、図３６（２）に示すＧのボックスは、図３２に示す鍵変換部３０２に相当する。
【０１７５】
図３６（１）の鍵変換を実行しない暗号処理構成では、適用鍵の構成情報（ビット列）が変換対象データである入力平文Ｐの全ビットに拡散（Ｆｕｌｌ　ｄｉｆｆｕｓｉｏｎ）するのに必要なラウンド数は、ラウンド関数（Ｒ）の処理に依存することになる。
従って、ラウンド関数の拡散性能が低い場合、拡散レベルを高くするためには、多くのラウンド数が必要となり、結果として、高速処理や軽量化を実現することが困難となる。
【０１７６】
一方、図３６（２）に示す本開示の鍵変換を実行する暗号処理構成では、適用鍵の構成情報（ビット列）が変換対象データである入力平文Ｐの全ビットに拡散（Ｆｕｌｌ　ｄｉｆｆｕｓｉｏｎ）するのに必要なラウンド数は、ベース鍵K_1と、変換鍵Kd_1が２つ利用されるラウンド数となる。
図の例では１ラウンドになる。
すなわち、本開示の処理を適用すれば、変換対象データに対する鍵データの構成情報の拡散が１ラウンドで実現され、ラウンド関数（Ｒ）の処理に依存することなくより大きな拡散性能が保証される。
すなわち、少ないラウンド数で攻撃に対する耐性の高い安全な暗号処理が実現される。

結果として、高速処理や軽量化が実現される。

【０１７７】

［５－３．鍵変換部のインボリューション性に基づく効果について］

先に図３２、図３３等を参照して説明したように、本開示の鍵スケジュール部３００の鍵変換部３０２は、順方向関数Ｇと逆方向関数$Ｇ^{-1}$が同一の関数で実現されるインボリューション性を有する。

以下、このインボリューション性に基づく効果について説明する。

【０１７８】

なお、同一のデータ変換処理であるラウンド関数を繰り返し実行する暗号処理装置のハードウェア実装構成としては、以下の２つのタイプがある。

（ａ）規定ラウンド数に相当する数のラウンド関数実行部をハードウェアとして構成する「アンロールド（Ｕｎｒｏｌｌｅｄ）実装」、

（ｂ）ハードウェアとしてのラウンド関数実行部を１つ構成し、そのラウンド関数実行部の出力を同じラウンド関数実行部にフィードバック入力して、規定ラウンド数のラウンド関数を繰り返し実行する「ラウンド実装」、

以下、これら２つのタイプにおけるインボリューション性に基づく効果について、順次、説明する。

【０１７９】

［５－３－ａ．アンロールド（Ｕｎｒｏｌｌｅｄ）実装における効果について］

まず、暗号処理部をアンロールド実装した場合に、本開示の鍵変換部のインボリューション性がもたらす効果について説明する。

【０１８０】

アンロールド（Ｕｎｒｏｌｌｅｄ）実装では、暗号処理部には、規定ラウンド数に相当する数のラウンド関数実行部がハードウェアとして構成される。

図３７、図３８を参照して、暗号処理部をアンロールド実装した場合に、本開示の鍵変換部のインボリューション性がもたらす効果について説明する。

【０１８１】

図３７、図３８に、以下の暗号処理装置の実装例を示す。

図３７（１）鍵変換部（Ｆ）がインボリューション性を有していない場合のハードウェア実装例

図３８（２ａ）、（２ｂ）鍵変換部（Ｇ）がインボリューション性を有している場合のハードウェア実装例

【０１８２】

図３８（２ａ），（２ｂ）は、本開示の鍵変換部、すなわちインボリューション性を有する場合の実装例に相当する。

図３７は、鍵変換部Ｆがインボリューション性を有していないため、鍵変換部Ｆによる変換結果として得られる鍵は、逐次、異なる鍵となる。図３７に示すように、鍵変換部Ｆの変換処理によって、鍵Ｋ１に基づいて、Ｋｄ１、Ｋｄ２、Ｋｄ３、Ｋｄ４、Ｋｄ５、Ｋｄ６が順次、生成され、これらの各鍵がラウンド鍵として暗号処理部の排他的論理和部（ラウンド鍵演算部）に順次入力する構成となる。

【０１８３】

これに対して、図３８（２ａ），（２ｂ）に示す例は、鍵変換部（Ｇ）がインボリューション性を有している場合のハードウェア実装例である。

図３８（２ａ）に示す例は、図３７（１）に示すハードウェア構成と同様、暗号処理部の排他的論理和部（ラウンド鍵演算部）に対する入力鍵（ラウンド鍵）を生成するための鍵変換部Ｇを各ラウンドに対応付けて設定した構成である。

【０１８４】

一方、図３８（２ｂ）は、鍵変換部Ｇを１つのみの設定として、予め保持するベース鍵$Ｋ_{1}$と、鍵変換部Ｇによって生成される変換鍵$Ｋｄ_{1}$を暗号処理部の排他的論理和部（ラウンド鍵演算部）に対して交互に入力する設定とした構成である。

【０１８５】
鍵変換部Ｇは、インボリューション性を有するため、鍵変換部による変換処理の繰り返しによって生成される鍵は、K_1，Kd_1，K_1，Kd_1，K_1・・・の繰り返しとなる。この性質に基づいて、図３８（２ｂ）に示すように鍵変換部Ｇを１つとして、ベース鍵K_1と、鍵変換部Ｇによる１回の鍵変換処理によって生成した変換鍵Kd_1を暗号処理部の排他的論理和部（ラウンド鍵演算部）に交互に入力することが可能となる。
この結果、鍵変換部Ｇの数を１つとすることが可能となり、ハードウェア実装の軽量化（小型化）が実現される。
【０１８６】
図３８（２ｂ）に示す構成に対応する暗号処理装置の構成例を図３９に示す。
図３９に示すように、鍵スケジュール部３００の鍵変換部（Ｇ）３０２は、１つのみとして暗号処理部３２０の各排他的論理和部（ラウンド鍵演算部）に対して予め保持するベース鍵K_1と、鍵変換部Ｇによって生成される変換鍵Kd_1を交互に入力することが可能となる。
【０１８７】
［５－３－ｂ．ラウンド実装における効果について］
次に、暗号処理装置をラウンド実装した場合に、本開示の鍵変換部のインボリューション性がもたらす効果について説明する。
【０１８８】
ラウンド実装では、暗号処理部に設定された１つのラウンド関数を繰り返し利用する構成となる。
図４０に、鍵変換部がインボリューション性を有していない場合の（ａ１）暗号処理構成と、（ａ２）ラウンド実装例を示す。
図４１に、鍵変換部がインボリューション性を有している場合の（ｂ１）暗号処理構成と、（ｂ２）ラウンド実装例を示す。
【０１８９】
図４０（ａ１）に示す暗号処理構成は、先に図３７を参照して説明した構成と同様の構成である。
すなわち、鍵変換部Ｆがインボリューション性を有していないため、鍵変換部Ｆによる変換結果として得られる鍵は、逐次、異なる鍵となる。図４０（ａ１）に示すように、鍵変換部Ｆの変換処理によって、鍵Ｋ１に基づいて、Ｋｄ１、Ｋｄ２、Ｋｄ３、Ｋｄ４、Ｋｄ５、Ｋｄ６が順次、生成され、これらの各鍵がラウンド鍵として暗号処理部の排他的論理和部（ラウンド鍵演算部）に順次入力する構成となる。
【０１９０】
この構成をラウンド型のハードウェア実装とした場合、図４０（ａ２）に示す構成となる。暗号処理部３５０は、１つの排他的論理和部（ラウンド鍵演算部）３５１と、１つの非線形／線形変換部３５２の構成とすることが可能となる。
一方、鍵スケジュール部３６０は、ベース鍵K_1を格納し供給する鍵レジスタ３６１と、変換鍵Kd_1～Kd_6を格納し供給するための鍵レジスタ３６２と、鍵変換部（Ｆ）３６３、鍵レジスタ３６１，３６２の出力切り換えを実行するスイッチ３６４を有する構成となる。
【０１９１】
これに対して、図４１（ｂ１）に示す暗号処理構成は、先に図３８（２ｂ）を参照して説明した構成と同様、鍵変換部Ｇがインボリューション性を有する構成とした暗号処理構成である。
すなわち、鍵変換部Ｇがインボリューション性を有しているため、鍵変換部による変換処理の繰り返しによって生成される鍵は、K_1，Kd_1，K_1，Kd_1，K_1・・・の繰り返しとなる。この性質に基づいて、図４１（ｂ１）に示すように鍵変換部Ｇを１つとして、ベース鍵K_1と、鍵変換部Ｇによる１回の鍵変換処理によって生成した変換鍵Kd_1を暗号処理部の排他的論理和部（ラウンド鍵演算部）に交互に入力することが可能となる

。
【０１９２】
この構成をラウンド型のハードウェア実装とした場合、図４１（ｂ２）に示す構成となる。暗号処理部３５０は、１つの排他的論理和部（ラウンド鍵演算部）３５１と、１つの非線形／線形変換部３５２の構成とすることが可能となる。
一方、鍵スケジュール部３７０は、ベース鍵K_1、および変換鍵Kd_1を格納し供給する鍵レジスタ３７１と、鍵変換部（Ｇ）３７２を有する構成となる。
【０１９３】
図４０（ａ２）に示す鍵変換部（Ｆ）がインボリューション性を持たない場合のラウンド実装構成では、鍵スケジュール部３６０には、２つの鍵レジスタ、１つの鍵変換部、１つのスイッチが必要となる。これに対して、図４１（ｂ２）に示す鍵変換部（Ｇ）がインボリューション性を有する場合のラウンド実装構成の鍵スケジュール部３７０は、１つの鍵レジスタと１つの鍵変換部によって構成され、ハードウェア構成の軽量化（小型化）が実現されることが証明される。
例えば、図４０（ａ２）に示すインボリューション性を持たない鍵変換部（Ｆ）を持つ場合のラウンド実装構成では、複数の異なる変換鍵を、順次、生成して格納、供給するための鍵レジスタが必要となり、この鍵レジスタ用のゲート数分の新たなハードウェア回路が必要となる。
【０１９４】
［５－４．本開示の鍵スケジュール部の構成と効果のまとめ］
上述したように、本開示の暗号処理装置に構成される鍵スケジュール部の鍵変換部は、以下の２つの特性を持つ。
（１）全拡散（ｆｕｌｌ　ｄｉｆｆｕｓｉｏｎ）変換を実現するフルディフュージョン性、
（２）順方向関数Ｇと、逆方向関数G^{-1}が同一であるインボリューション性、
これら２つの特性を持つ。
【０１９５】
これらの２つの特性に基づいて、以下の効果が発揮される。
（効果１）フルディフュージョン性に基づいて、変換対象とするデータについてのフルディフュージョン性を少ないラウンドで実現できる。
この結果、少ないラウンドで安全性の高い暗号処理を行なうことができ、処理の高速化（低遅延化）、および装置の軽量化（小型化）が実現される。
（効果２）ハードウェア構成をアンロール実装とした場合、インボリューション性に基づいて、鍵変換部を１つのみの構成とすることが可能となり、装置の軽量化（小型化）が実現される。
（効果３）ハードウェア構成をラウンド実装とした場合、インボリューション性に基づいて、鍵レジスタと、鍵変換部を各々１つのみとした鍵スケジュール部の実装が可能となり、装置の軽量化（小型化）が実現される。
【０１９６】
［５－５．鍵スケジュール部のその他の構成例について］
次に、上述した以下の２つの特性、すなわち、
（１）全拡散（ｆｕｌｌ　ｄｉｆｆｕｓｉｏｎ）変換を実現するフルディフュージョン性、
（２）順方向関数Ｇと、逆方向関数G^{-1}が同一であるインボリューション性、
これら２つの特性を持つ関数Ｇを適用した鍵変換処理を実行する鍵変換部を有する鍵スケジュール部のその他の構成例について、図４２以下を参照して説明する。
【０１９７】
図４２は、上述の２つの特性を有する鍵変換部を持つ鍵スケジュール部３８０を有する暗号処理装置の一構成例を示す図である。
図４２に示す暗号処理装置は、鍵スケジュール部３８０と、暗号処理部３８５を有する

。
鍵スケジュール部３８０の鍵レジスタ３８１には、予め生成された秘密鍵Ｋが格納される。
秘密鍵Ｋは、鍵Ｋ$_1$と鍵Ｋ$_2$の連結データである。
例えば鍵Ｋ$_1$、Ｋ$_2$は６４ビット鍵であり、その連結データである秘密鍵Ｋは１２８ビットデータである。
図に示すＧは、鍵変換部であり、先に図３２以下を参照して説明した鍵変換部３０２と同様、フルディフュージョン性と、インボリューション性、これら２つの特性を持つ関数Ｇを適用した鍵変換処理を行なう鍵変換部である。
【０１９８】
図４２に示す鍵スケジュール部３８０は、鍵レジスタ３８１に格納された秘密鍵Ｋの分割データである鍵Ｋ$_1$、Ｋ$_2$、さらに、これらの鍵を鍵変換部（Ｇ）において変換した変換鍵Ｋｄ$_1$、Ｋｄ$_2$を暗号処理部３８５の排他的論理和部（ラウンド鍵演算部）に順次出力する。
なお、図に示す例において、鍵Ｋ$_1$、Ｋ$_2$が６４ビット鍵である場合、暗号処理部３８５の変換対象となる平文Ｐも６４ビットデータとなる。
【０１９９】
図４２に示す例では、鍵の出力順は、以下の通りである。
鍵Ｋ$_1$、
鍵Ｋ$_2$、
変換鍵Ｋｄ$_1$、
変換鍵Ｋｄ$_2$、
鍵Ｋ$_1$、
鍵Ｋ$_2$、
変換鍵Ｋｄ$_1$、
この順番で、４種類の鍵を暗号処理部３８５に入力する。
なお、鍵入力順は様々な設定が可能である。
【０２００】
なお、図４２には、鍵変換部（Ｇ）を複数、示しているが、先に図４１を参照して説明したラウンド実装を行う場合、この鍵変換部（Ｇ）は１つのみの構成とすることが可能である。
【０２０１】
図４３を参照して鍵スケジュール部のもう１つの構成例について説明する。
図４３には、以下の各図を示している。
（ａ）鍵スケジュール部の構成
（ｂ）鍵スケジュール部による鍵出力構成
【０２０２】
図４３（ａ）に示すように、鍵スケジュール部の鍵レジスタ３９１には、予め生成された秘密鍵Ｋが格納される。
秘密鍵Ｋは、鍵Ｋ$_1$と鍵Ｋ$_2$の連結データである。
例えば鍵Ｋ$_1$、Ｋ$_2$は６４ビット鍵であり、その連結データである秘密鍵Ｋは１２８ビットデータである。
【０２０３】
図４３（ａ）に示す鍵スケジュール部は、鍵変換部Ｇ３９３と、排他的論理和部３９２，３９４を有する。
鍵変換部Ｇ３９３は、先に図３２以下を参照して説明した鍵変換部３０２と同様、フルディフュージョン性と、インボリューション性、これら２つの特性を持つ関数Ｇを適用した鍵変換処理を行なう鍵変換部である。
【０２０４】
図４３（ａ）に示す鍵スケジュール部は、これらの各構成に基づいて、以下の６種類の

鍵を生成する。
鍵K_1、
鍵K_2、
変換鍵Kd_1、
変換鍵Kd_2、
排他的論理和演算鍵K_1（＋）K_2、
排他的論理和演算変換鍵Kd_1（＋）Kd_2、
図４３（ａ）に示す鍵スケジュール部は、これら６種類の鍵を生成して、暗号処理部に順次、出力する。

【０２０５】
なお、上記６種類の鍵は、鍵Ｋ＝１２８ビットである場合、すべて６４ビット鍵となる。この場合、暗号処理部の変換対象となる平文Ｐも６４ビットデータとなる。

【０２０６】
図４３（ｂ）に示す例では、鍵の出力順は、以下の通りである。
鍵K_1、
鍵K_2、
変換鍵Kd_1、
変換鍵Kd_2、
排他的論理和演算鍵K_1（＋）K_2、
排他的論理和演算変換鍵Kd_1（＋）Kd_2、
排他的論理和演算鍵K_1（＋）K_2、
排他的論理和演算変換鍵Kd_1（＋）Kd_2、
排他的論理和演算鍵K_1（＋）K_2、
変換鍵Kd_2、
変換鍵Kd_1、
鍵K_2、
鍵K_1、
この順番で、６種類の鍵を暗号処理部に入力する。

【０２０７】
上記鍵の入力シーケンスは、逆の順番も同一シーケンスとなる。
これは、平文Ｐから暗号文Ｃを生成する暗号処理における鍵入力順と、暗号文Ｃから平文Ｐを生成する復号処理における鍵入力順を同じ設定とすることが可能であることを意味する。これは、暗号処理および復号処理に適用するハードウェアやプログラムを共通化可能であることを意味し、装置の軽量化（小型化）に寄与する設定である。
なお、図４３に示す鍵スケジュール部を有する暗号処理装置の具体的構成については、さらに、後段で説明する。

【０２０８】
［５－６．フルディフュージョン性を持つ鍵変換部を有する構成例について］
上述した実施例では、秘密鍵Ｋに対して変換関数Ｇを適用した変換処理により変換鍵Ｋｄを生成する鍵変換部がインボリューション性と、フルディフュージョン性の２つの性質を持つものとして説明したが、インボリューション性を持たずフルディフュージョン性のみを有する鍵変換部を適用した場合においても、入力データに対する拡散性能が向上し、各種攻撃に対する耐性の高い安全な暗号処理構成が実現される。
以下、フルディフュージョン性を有する暗号処理構成についての実施例について説明する。

【０２０９】
図４４は、鍵変換関数Ｇがフルディフュージョン性を有する場合、入力データ（Ｐ）の内部状態Ｓのフルディフュージョン性が保証されることを示す図である。先の項目［５－２．鍵スケジュール部のフルディフュージョン性に基づく効果について］において図３５等を参照して説明したように、鍵変換部の鍵変換関数Ｇがフルディフュージョン性を有す

る場合、変換対象データと鍵との間にもフルディフュージョン性を保証することになる。

【０２１０】

図４４に示す構成において、ベース鍵K_1は、暗号処理部の排他的論理和部に入力されて、入力ステートとの排他的論理和演算がなされる。その後、さらに、ラウンド演算部R_1において非線形変換処理と、線形変換処理がなされる。

さらにその出力に対して、排他的論理和演算部において、変換鍵Kd_1との排他的論理和演算が実行される。

排他的論理和演算部の出力（Ｓ）について考察する。

【０２１１】

ベース鍵K_1と変換鍵Kd_1との間にはフルディフュージョン性が保証されている。

変換対象データは、ラウンド演算部R_1において非線形変換処理と、線形変換処理がなされた後、変換鍵Kd_1との排他的論理和演算が実行される。

この結果、ベース鍵K_1と変換データとの間でもフルディフュージョン性が保証される。

この性質は、暗号処理装置の安全性や、実装性能に貢献をもたらす性質であると言える。

具体的には、鍵によるデータ拡散性の向上が実現され、ラウンド数を削減しても高い拡散性能を発揮させることが可能になる。この結果、各種の攻撃に対する耐性を高めることが可能となる。たとえば、鍵依存度を利用した中間値一致攻撃等に基づく鍵解析処理に対する耐性をより向上させることができる。

【０２１２】

なお、図４５に示すような鍵変換を実行しない暗号処理構成では、適用鍵の構成情報（ビット列）が変換対象データである入力平文Ｐの全ビットに拡散（Ｆｕｌｌ　ｄｉｆｆｕｓｉｏｎ）するのに必要なラウンド数は、ラウンド関数（Ｒ）の処理に依存することになる。

これに対して、図４４に示す鍵変換を行う構成では、適用鍵の構成情報（ビット列）が変換対象データである入力平文Ｐの全ビットに拡散（Ｆｕｌｌ　ｄｉｆｆｕｓｉｏｎ）するのに必要なラウンド数は、ベース鍵K_1と、変換鍵Kd_1が２つ利用されるラウンド数となる。

図の例では１ラウンドになる。

すなわち、本開示の処理を適用すれば、変換対象データに対する鍵データの構成情報の拡散が１ラウンドで実現され、ラウンド関数（Ｒ）の処理に依存することなくより大きな拡散性能が保証される。

すなわち、少ないラウンド数で攻撃に対する耐性の高い安全な暗号処理が実現される。結果として、高速処理や軽量化が実現される。

【０２１３】

フルディフュージョン性を持つＧ関数の具体例について説明する。

以下で説明するＧ関数は、以下の２つの関数の組み合わせによって構成される。

（ａ）フルディフュージョン４ビット関数（Df_4）

（ｂ）１６ビット置換関数（Bp_{16}）

【０２１４】

（ａ）フルディフュージョン４ビット関数は、入出力４ビットとする変換関数であり、出力４ビットのすべてのビットに入力４ビットの影響が表れるフルディフュージョン性を持つ関数である。

すなわち、

入力：x_0，x_1，x_2，x_3（各１ｂｉｔ）

出力：y_0，y_1，y_2，y_3（各１ｂｉｔ）

このとき、関数ｆは、

$y_i = f(x_0, x_1, x_2, x_3)$

ただしｉ＝０，１，２，３

上記性質を持つ関数である。

【０２１５】

次に、（ｂ）１６ビット置換関数（Bp_{16}）について図４６を参照して説明する。

図４６には、１６ビット置換関数（Bp_{16}）の一例を示している。

入力Ｘをx_0，x_1，x_2，・・・x_{15}の１６ビットデータとし、

入力Ｘを変換関数Ｇに入力して変換された後の、

出力Ｙをy_0，y_1，y_2，・・・y_{15}の１６ビットデータとする。

なお、x_i，y_iはそれぞれ０または１の１ビットデータである

【０２１６】

１６ビット置換関数（Bp_{16}）による入出力ビットの関係は以下の対応関係となる。

入力：x_0，x_1，・・・，x_{15}（各１ｂｉｔ）

出力：y_0，y_1，・・・，y_{15}（各１ｂｉｔ）

関数：$y_0=x_0$，$y_1=x_4$，$y_2=x_8$，$y_3=x_{12}$

$y_4=x_1$，$y_5=x_5$，$y_6=x_9$，$y_7=x_{13}$

$y_8=x_2$，$y_9=x_6$，$y_{10}=x_{10}$，$y_{11}=x_{14}$

$y_{12}=x_3$，$y_{13}=x_7$，$y_{14}=x_{11}$，$y_{15}=x_{15}$

【０２１７】

図４７は、

（ａ）フルディフュージョン４ビット関数（Df_4）

（ｂ）１６ビット置換関数（Bp_{16}）

これらの２つの関数から構成されるフルディフュージョン性を持つＧ関数を適用した鍵変換処理例（処理例１）を示す図である。

ベース鍵をＡとし、変換鍵Ｂとする。いずれも各要素４ビットの４×４ステートデータである。

図４７に示す例において、ベース鍵Ａから変換鍵Ｂを生成する処理は、以下の４つのステップによって構成される。

（Ｓ１１）ベース鍵Ａの１６個の４ビット要素各々に対してフルディフュージョン４ビット関数（Df_4）を適用して変換する。

（Ｓ１２）ステップＳ１１の変換処理によって生成されたデータ（４×４ステート）の各列１６ビットデータ各々に対して、１６ビット置換関数（Bp_{16}）を適用して変換する。

（Ｓ１３）ステップＳ１２の変換処理によって生成されたデータ（４×４ステート）の１６個の４ビット要素各々に対してフルディフュージョン４ビット関数（Df_4）を適用して変換する。

（Ｓ１４）ステップＳ１３の変換処理によって生成されたデータ（４×４ステート）の各行１６ビットデータ各々に対して、１６ビット置換関数（Bp_{16}）を適用して変換する。

【０２１８】

これらの処理によって、ベース鍵Ａから変換鍵Ｂを生成する。

変換鍵のＢの各要素b_0～b_{15}は、ベース鍵Ａの各要素a_0～a_{15}の影響を受けたデータとなりベース鍵Ａと変換鍵Ｂとの間にはフルディフュージョン性が保証される。

【０２１９】

図４８も、

（ａ）フルディフュージョン４ビット関数（Df_4）

（ｂ）１６ビット置換関数（Bp_{16}）

これらの２つの関数から構成されるフルディフュージョン性を持つＧ関数を適用した鍵変換処理例（処理例２）を示す図である。

ベース鍵をＡとし、変換鍵Ｂとする。いずれも各要素４ビットの４×４ステートデータである。

図４８に示す例において、ベース鍵Ａから変換鍵Ｂを生成する処理は、以下の５つのス

テップによって構成される。
（Ｓ２１）ベース鍵Ａの１６個の４ビット要素各々に対してフルディフュージョン４ビット関数（Df_4）を適用して変換する。
（Ｓ２２）ステップＳ２１の変換処理によって生成されたデータ（４×４ステート）の各列１６ビットデータ各々に対して、１６ビット置換関数（Bp_{16}）を適用して変換する。
（Ｓ２３）ステップＳ２２の変換処理によって生成されたデータ（４×４ステート）の１６個の４ビット要素各々に対してフルディフュージョン４ビット関数（Df_4）を適用して変換する。
（Ｓ２４）ステップＳ２３の変換処理によって生成されたデータ（４×４ステート）の各行１６ビットデータ各々に対して、１６ビット置換関数（Bp_{16}）を適用して変換する。
（Ｓ２５）ステップＳ２４の変換処理によって生成されたデータ（４×４ステート）の１６個の４ビット要素各々に対してフルディフュージョン４ビット関数（Df_4）を適用して変換する。
【０２２０】
これらの処理によって、ベース鍵Ａから変換鍵Ｂを生成する。
変換鍵のＢの各要素b_0～b_{15}は、ベース鍵Ａの各要素a_0～a_{15}の影響を受けたデータとなりベース鍵Ａと変換鍵Ｂとの間にはフルディフュージョン性が保証される。
【０２２１】
図４９も、
（ａ）フルディフュージョン４ビット関数（Df_4）
（ｂ）１６ビット置換関数（Bp_{16}）
これらの２つの関数から構成されるフルディフュージョン性を持つＧ関数を適用した鍵変換処理例（処理例３）を示す図である。
ベース鍵をＡとし、変換鍵Ｂとする。いずれも各要素４ビットの４×４ステートデータである。
この処理例３において適用する
（ａ）フルディフュージョン４ビット関数（Df_4）
は、インボリューション性も有する関数である。
【０２２２】
図４９に示す例において、ベース鍵Ａから変換鍵Ｂを生成する処理は、以下の５つのステップによって構成される。
（Ｓ３１）ベース鍵Ａの１６個の４ビット要素各々に対してインボリューション性を有し、かつフルディフュージョン性を有する４ビット関数（Df_4）を適用して変換する。
（Ｓ３２）ステップＳ３１の変換処理によって生成されたデータ（４×４ステート）の各列１６ビットデータ各々に対して、１６ビット置換関数（Bp_{16}）を適用して変換する。
（Ｓ３３）ステップＳ３２の変換処理によって生成されたデータ（４×４ステート）の１６個の４ビット要素各々に対してインボリューション性を有し、かつフルディフュージョン性を有する４ビット関数（Df_4）を適用して変換する。
（Ｓ３４）ステップＳ３３の変換処理によって生成されたデータ（４×４ステート）の各行１６ビットデータ各々に対して、１６ビット置換関数（Bp_{16}）を適用して変換する。
（Ｓ３５）ステップＳ３４の変換処理によって生成されたデータ（４×４ステート）の１６個の４ビット要素各々に対してインボリューション性を有し、かつフルディフュージョン性を有する４ビット関数（Df_4）を適用して変換する。
【０２２３】
これらの処理によって、ベース鍵Ａから変換鍵Ｂを生成する。
変換鍵のＢの各要素b_0～b_{15}は、ベース鍵Ａの各要素a_0～a_{15}の影響を受けた

データとなりベース鍵Ａと変換鍵Ｂとの間にはフルディフュージョン性が保証される。さらに、（ａ）フルディフュージョン４ビット関数（Ｄｆ$_4$）、（ｂ）１６ビット置換関数（Ｂｐ$_{16}$）これらの関数が双方ともインボリューション性を有しているため、ベース鍵Ａと変換鍵Ｂとの間にはインボリューション性も保証される。

【０２２４】

図５０も、

（ａ）フルディフュージョン４ビット関数（Ｄｆ$_4$）

（ｂ）１６ビット置換関数（Ｂｐ$_{16}$）

これらの２つの関数から構成されるフルディフュージョン性を持つＧ関数を適用した鍵変換処理例（処理例４）を示す図である。

ベース鍵をＡとし、変換鍵Ｂとする。いずれも各要素４ビットの４×４ステートデータである。

この処理例４において適用する

（ａ）フルディフュージョン４ビット関数（Ｄｆ$_4$）

は、インボリューション性も有する関数である。

【０２２５】

図５０に示す例において、ベース鍵Ａから変換鍵Ｂを生成する処理は、以下の５つのステップによって構成される。

（Ｓ４１）ベース鍵Ａ（４×４ステート）の各列１６ビットデータ各々に対して、１６ビット置換関数（Ｂｐ$_{16}$）を適用して変換する。

（Ｓ４２）ステップＳ４１の変換処理によって生成されたデータ（４×４ステート）の１６個の４ビット要素各々に対してインボリューション性を有し、かつフルディフュージョン性を有する４ビット関数（Ｄｆ$_4$）を適用して変換する。

（Ｓ４３）ステップＳ４２の変換処理によって生成されたデータ（４×４ステート）の各列１６ビットデータ各々に対して、１６ビット置換関数（Ｂｐ$_{16}$）を適用して変換する。

（Ｓ４４）ステップＳ４３の変換処理によって生成されたデータ（４×４ステート）の１６個の４ビット要素各々に対してインボリューション性を有し、かつフルディフュージョン性を有する４ビット関数（Ｄｆ$_4$）を適用して変換する。

（Ｓ４５）ステップＳ４４の変換処理によって生成されたデータ（４×４ステート）の各行１６ビットデータ各々に対して、１６ビット置換関数（Ｂｐ$_{16}$）を適用して変換する。

【０２２６】

これらの処理によって、ベース鍵Ａから変換鍵Ｂを生成する。

変換鍵のＢの各要素ｂ$_0$～ｂ$_{15}$は、ベース鍵Ａの各要素ａ$_0$～ａ$_{15}$の影響を受けたデータとなりベース鍵Ａと変換鍵Ｂとの間にはフルディフュージョン性が保証される。さらに、（ａ）フルディフュージョン４ビット関数（Ｄｆ$_4$）、（ｂ）１６ビット置換関数（Ｂｐ$_{16}$）これらの関数が双方ともインボリューション性を有しているため、ベース鍵Ａと変換鍵Ｂとの間にはインボリューション性も保証される。

【０２２７】

図５１も、

（ａ）フルディフュージョン４ビット関数（Ｄｆ$_4$）

（ｂ）１６ビット置換関数（Ｂｐ$_{16}$）

これらの２つの関数から構成されるフルディフュージョン性を持つＧ関数を適用した鍵変換処理例（処理例５）を示す図である。

ベース鍵をＡとし、変換鍵Ｂとする。いずれも各要素４ビットの４×４ステートデータである。

この処理例５において適用する

（ａ）フルディフュージョン４ビット関数（Ｄｆ$_4$）

は、インボリューション性も有する関数である。

【０２２８】
図５１に示す例において、ベース鍵Ａから変換鍵Ｂを生成する処理は、以下の５つのステップによって構成される。
（Ｓ５１）ベース鍵Ａの１６個の４ビット要素各々に対してインボリューション性を有し、かつフルディフュージョン性を有する４ビット関数（Df_4）を適用して変換する。
（Ｓ５２）ステップＳ５１の変換処理によって生成されたデータ（４×４ステート）の各行１６ビットデータ各々に対して、１６ビット置換関数（Bp_{16}）を適用して変換する。
（Ｓ５３）ステップＳ５２の変換処理によって生成されたデータ（４×４ステート）の１６個の４ビット要素各々に対してインボリューション性を有し、かつフルディフュージョン性を有する４ビット関数（Df_4）を適用して変換する。
（Ｓ５４）ステップＳ５３の変換処理によって生成されたデータ（４×４ステート）の各列１６ビットデータ各々に対して、１６ビット置換関数（Bp_{16}）を適用して変換する。
（Ｓ５５）ステップＳ５４の変換処理によって生成されたデータ（４×４ステート）の１６個の４ビット要素各々に対してインボリューション性を有し、かつフルディフュージョン性を有する４ビット関数（Df_4）を適用して変換する。
【０２２９】
これらの処理によって、ベース鍵Ａから変換鍵Ｂを生成する。
変換鍵のＢの各要素b_0～b_{15}は、ベース鍵Ａの各要素a_0～a_{15}の影響を受けたデータとなりベース鍵Ａと変換鍵Ｂとの間にはフルディフュージョン性が保証される。さらに、（ａ）フルディフュージョン４ビット関数（Df_4）、（ｂ）１６ビット置換関数（Bp_{16}）これらの関数が双方ともインボリューション性を有しているため、ベース鍵Ａと変換鍵Ｂとの間にはインボリューション性も保証される。
【０２３０】
図４７～図５１まで５つの鍵変換関数Ｇの構成例を説明した。
これらの鍵変換関数は、ベース鍵Ｋから変換鍵を生成する場合に適用可能であり、また、先に図４２を参照して説明したベース鍵Ｋを分割して生成される分割鍵に対する変換処理に適用することも可能である。
さらに、２つの分割鍵に対して適用する鍵変換関数を異なる設定としてもよい。
【０２３１】
図５２に示す暗号処理装置は、鍵スケジュール部３８０と、暗号処理部３８５を有する。鍵スケジュール部３８０の鍵レジスタ３８１には、予め生成された秘密鍵Ｋが格納される。
秘密鍵Ｋは、鍵K_1と鍵K_2の連結データである。
例えば鍵K_1、K_2は６４ビット鍵であり、その連結データである秘密鍵Ｋは１２８ビットデータである。
図に示すＧ１、Ｇ２は、鍵変換部である。
これらは、少なくともフルディフュージョン性を有する。
あるいはフルディフュージョン性と、インボリューション性、これら２つの特性を有する。
【０２３２】
鍵変換関数Ｇ１，Ｇ２の組み合わせとしては、例えば以下のような設定すが可能である。
（ａ）Ｇ１，Ｇ２ともフルディフュージョン性を有するがインボリューション性を有さない。
（ｂ）Ｇ１，Ｇ２ともフルディフュージョン性と、インボリューション性を有する。
（ｃ）Ｇ１，Ｇ２ともフルディフュージョン性を有し、Ｇ１とＧ２が逆関数の設定、すなわち、$G2 = G1^{-1}$の関係にある。
鍵変換関数Ｇ１，Ｇ２の組み合わせとしては上記のような様々な設定が可能である。

【０２３３】
［６．定数入力による安全性向上を実現する構成について］
次に、ラウンド演算を繰り返し実行する暗号処理部に対して定数（Ｃｏｎｓｔａｎｔ）を入力して、変換対象データあるいはラウンド鍵と、定数との演算を実行し、拡散性能を高めた暗号処理装置について説明する。
【０２３４】
［６－１．定数入力による安全性向上を実現した従来構成とその問題点について］
ラウンド関数によるラウンド演算を繰り返し実行する構成において、ラウンド毎の変換処理の同一性を排除するため、ラウンド毎に異なる定数を作用させる構成については、従来から提案されている。
このような定数入力処理は、スライドアタック、リフレクションアタックと呼ばれる攻撃に対する耐性を高めるために有効な手法であると言われている。
【０２３５】
まず、従来型の定数入力構成の概略と問題点について説明する。
従来型の定数入力構成例としては、例えば図５３に示すような構成がある。
図５３に示すラウンド演算実行部４０１ａ～ｄは暗号処理部における排他的論理和部（ラウンド鍵演算部）、非線形変換部、線形変換部を含むラウンド関数実行部である。
この各ラウンド演算部４０１ａ～ｄに、定数１（ＣＯＮ１）～定数４（ＣＯＮ４）を順次、入力する。
なお、入力された定数ＣＯＮは、各ラウンド演算部における変換データ、あるいはラウンド鍵との排他的論理和演算が実行される。
このように、各ラウンドに様々な定数による演算を実行することでラウンド演算間の同一性を排除して、様々な攻撃に対する耐性を高めることができる。
【０２３６】
次に、このような定数入力構成における問題点について説明する。
暗号処理装置において、ラウンド関数の設定を工夫することで、暗号処理と復号処理を同じ装置で実行可能となる。
具体的には、図５４に示すように、暗号処理装置の暗号処理部に適用する複数の変換関数の構成を中心から左右に分割したとき、左半分と右半分が逆関数となる関係とすることで、暗号処理と復号処理を同じ装置で実行可能となる。
これは、インボリューション性を有する暗号処理装置と呼ばれる。
【０２３７】
なお、図５４に示す例では、変換関数Ｅ，４１１と、変換関数E^{-1}，４１３は逆関数の関係にある。中心の線形変換部Ｍは、入力Ａに対して、出力Ｂを出力し、入力Ｂに対して出力Ａを出力する。
平文Ｐに対して、変換関数Ｅ，４１１、線形変換部４１２、変換関数E^{-1}，４１３をこの順番で適用して暗号文Ｃが得られる。
また、暗号文Ｃに対して、同じ順番で各変換部を適用。すなわち、変換関数Ｅ，４１１、線形変換部４１２、変換関数E^{-1}，４１３をこの順番で適用することで、元の平文Ｐが得られる。
【０２３８】
このような暗号処理装置を、インボリューション性を持つ暗号処理装置という。なお、インボリューション性を持つ暗号処理装置のなかには、ラウンド関数の実行シーケンスが順方向、逆方向でいずれも同一シーケレンスであるのみならず、各ラウンドにおいて適用するラウンド鍵の入力順も順方向、逆方向とも同一となるようなものも存在する。
例えば、先に図４３を参照して説明した鍵入力シーケンスが暗号処理装置のインボリューション性を実現する１つの鍵入力シーケンスである。
【０２３９】
しかし、このようなインボリューション性を持つ暗号処理装置の一つの問題として安全性の問題がある。

この問題点について、図５５を参照して説明する。

【０２４０】

図５５（ａ）は、インボリューション性を持つ暗号処理装置に、定数を入力しない場合の各変換部のデータ入出力値の関係を説明する図である。

平文Ｐの一部の構成データ＝Ｙとする。

データＹに対する変換関数Ｅ，４１１の変換結果をＸとする。

【０２４１】

線形変換部４１２は、変換関数Ｅ，４１１からの出力値に対する線形変換を実行するが、その一部の構成データ（ビット）の値は変化しないでそのまま出力される場合がある。なお、線形変換において入出力値が同じ値となる点を不動点とよび、多くの暗号処理装置で適用される線形変換処理には、いくつかの不動点が存在する。

【０２４２】

図５５（ａ）に示す例では、線形変換部４１２に対する入力値Ｘが、線形変換部４１２の不動点の作用によって、そのまま線形変換部４１２の出力Ｘとなったものとする。

この場合、値Ｘは、変換関数E^{-1}，４１３に入力される。変換関数E^{-1}，４１３は、変換関数Ｅ，４１１の逆関数であるため、入力値Ｘは元の値Ｙに戻されることになる。

すなわち、暗号文Ｃを構成する一部の出力値Ｙは、入力平文Ｐの構成値Ｙと同じ値になってしまう。すなわち暗号処理装置全体においても入出力値が変わらない不動点が発生することになる。

このような性質は、様々な攻撃に対する脆弱性をもたらす性質であり、暗号処理装置としての安全性を損なう好ましくない性質である。

【０２４３】

図５５（ｂ）は、図５５（ａ）と同様インボリューション性を持つ暗号処理装置であるが、変換関数Ｅ，４１１において定数１（ＣＯＮ１）を入力した演算を行い、また、変換関数E^{-1}，４１３に定数２（ＣＯＮ２）を入力した演算を行う構成としている。

これらの定数を入力した場合の各変換部のデータ入出力値の関係を示している。

平文Ｐの一部の構成データ＝Ｙとする。

データＹに対する変換関数Ｅ，４１１の変換結果をＸとする。

【０２４４】

この例では、線形変換部４１２に対する入力値Ｘが、線形変換部４１２による線形変換処理によってＸ＋ΔＡに変換されたものとする。

この場合、値Ｘ＋ΔＡが、変換関数E^{-1}，４１３に入力される。変換関数E^{-1}，４１３は、変換関数Ｅ，４１１の逆関数であるが、定数２（ＣＯＮ２）を入力した演算を実行する構成であり、定数１（ＣＯＮ１）を入力した演算を実行する変換関数Ｅ，４１１の完全な逆関数とはならない。

しかし、定数の選択のしかたによっては、図に示すように、変換関数E^{-1}，４１３に対する入力Ｘ＋ΔＡに対応する出力値がＹ＋ΔＢのような設定となる場合がある。

【０２４５】

すなわち、

線形変換部の入出力値の対応が、Ｘと、Ｘ＋ΔＡ、

暗号処理装置の入出力値の対応が、Ｙと、Ｙ＋ΔＢ、

このように、入力データに対して特定の差分データを追加した関係性が発生する場合がある。

このような入出力データの関係性も、やはり様々な攻撃に対する脆弱性をもたらす性質であり、暗号処理装置としての安全性を損なう好ましくない性質である。

【０２４６】

［６－２．安全性の高い定数入力構成を持つ暗号処理装置の構成について］

次に、上記のような従来構成の問題点を解決した安全性の高い定数入力構成を持つ暗号処理装置の構成について説明する。

【０２４７】

図５６以下を参照して本実施例にかかる暗号処理装置の構成例について説明する。
図５６は、本実施例に係る暗号処理部に対する定数入力構成例を説明する図である。
図５６（ａ）には、図５４を参照して説明したと同様、インボリューション性を有するデータ変換部からなる暗号処理部を示している。
すなわち、暗号処理部は、
変換関数Ｅ，４３１、
線形変換部４３２、
変換関数Ｅ$^{-1}$，４３３、
これらのデータ変換部を有し、変換関数Ｅ$^{-1}$，４３３、は、変換関数Ｅ４３１の逆関数である。

【０２４８】
本実施例において、定数（ＣＯＮ）４３５は、変換関数Ｅ$^{-1}$，４３３に入力する。
なお、変換関数Ｅ$^{-1}$，４３３は、複数のラウンド関数によって構成され、定数（ＣＯＮ）は、１つ以上のラウンド関数部に対して入力する構成とする。
なお、ここに示す実施例では、定数（ＣＯＮ）の入力部は、変換関数Ｅ$^{-1}$，４３３に入力する設定としているが、変換関数Ｅ，４３１側に入力する設定としてもよい。

【０２４９】
すなわち、暗号処理部は、変換関数Ｅ，４３１と、変換関数Ｅ，４３１の逆関数で変換関数Ｅ$^{-1}$４３３をシーケンシャルに実行するインボリューション性を有し、変換関数Ｅ、または、逆関数Ｅ$^{-1}$のいずれか一方のみにおいて、１回以上の定数を適用したラウンド演算を実行する構成とする。

【０２５０】
図５６（ｂ）は、定数（ＣＯＮ）４３５の入力構成の具体例を示している。定数（ＣＯＮ）４３５は、暗号処理部の線形変換部４３７の前段の排他的論理和部４３６に入力され。排他的論理和部４３６に対する入力データＡと、排他的論理和演算を実行する。
なお、排他的論理和部４３６は、ラウンド鍵Ｋｒとの排他的論理和演算を行うラウンド鍵演算部であり、排他的論理和部４３６では、図に示す前段のラウンド演算部からの出力であるデータＡと、ラウンド鍵Ｋｒと、定数ＣＯＮとの排他的論理和演算が実行されることになる。
すなわち、排他的論理和部４３６では、以下の演算実行結果としてのＢが算出され、後段のラウンド演算部の線形変換処理部４３７に出力される。
Ｂ＝Ａ（＋）Ｋｒ（＋）ＣＯＮ
なお、上記式において、（＋）は排他的論理和演算を示すものとする。

【０２５１】
この構成において、入力する定数（ＣＯＮ）４３５の条件として、以下の条件を設定する。
条件：排他的論理和部４３７に隣接する線形変換部、図に示す例では線形変換部４３７における線形変換処理における入出力値の差分が減少しない値に設定する。
上記条件は、具体的には、定数ＣＯＮと、線形変換部４３７において適用する線形変換行列との行列演算の結果として得られる全ての要素が非ゼロ、すなわちゼロでない値となることである。

【０２５２】
図５７を参照してこの条件について説明する。
図５７には、排他的論理和部４３６に入力する定数ＣＯＮ４３５を構成する４×４マトリックスと、線形変換部４３７において起用する線形変換行列Ｍを示している。
定数ＣＯＮ４３５は、各要素（con_0～con_{15}）が４ビットデータである４×４ステートであり、６４ビットデータである。
また、線形変換行列Ｍは、４×４の行列データである。
すなわち、以下に示す線形変換行列である。

【０２５３】

【数１２】

$$\begin{pmatrix} m_0 & m_4 & m_8 & m_{12} \\ m_1 & m_5 & m_9 & m_{13} \\ m_2 & m_6 & m_{10} & m_{14} \\ m_3 & m_7 & m_{11} & m_{15} \end{pmatrix}$$

【０２５４】

定数ＣＯＮの条件は、定数ＣＯＮと、線形変換部４３７において適用する線形変換行列との行列演算の結果として得られる全ての要素が非ゼロ、すなわちゼロでない値となることである。

すなわち、以下の行列演算によって得られる値が、すべて非ゼロとなることである。

【０２５５】

【数１３】

$$\begin{pmatrix} m_0 & m_4 & m_8 & m_{12} \\ m_1 & m_5 & m_9 & m_{13} \\ m_2 & m_6 & m_{10} & m_{14} \\ m_3 & m_7 & m_{11} & m_{15} \end{pmatrix} \cdot \begin{pmatrix} con_0 \\ con_1 \\ con_2 \\ con_3 \end{pmatrix}, \ldots, \begin{pmatrix} m_0 & m_4 & m_8 & m_{12} \\ m_1 & m_5 & m_9 & m_{13} \\ m_2 & m_6 & m_{10} & m_{14} \\ m_3 & m_7 & m_{11} & m_{15} \end{pmatrix} \cdot \begin{pmatrix} con_{12} \\ con_{13} \\ con_{14} \\ con_{15} \end{pmatrix}$$

【０２５６】

上記行列演算式によって算出される１６個の値がすべてゼロでない、すなわち非ゼロとなるように、定数ＣＯＮを設定する。

このような設定により、定数ＣＯＮを入力する排他的論理和部に隣接する線形変換部、図５７に示す例では線形変換部４３７における線形変換処理の入出力値の差分の減少を防止でき、結果として、最小差分アクティブＳボックスの数を所定数以上に維持することが可能となる。

【０２５７】

具体的な定数ＣＯＮの設定例について、図５８を参照して説明する。

図５８に示す例は、定数ＣＯＮ４３５を入力する排他的論理和部４３６に隣接する線形変換部４３７を、先に図２３～図３０を参照して説明した線形変換部Ｐ１とした設定である。すなわち、以下に示す行列を適用した列拡散演算（ＭｉｘＣｏｌｕｍｎ）を行う設定とした例である。

【０２５８】

【数１４】

$$M_0 = \begin{pmatrix} 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{pmatrix}$$

【０２５９】

線形変換部４３７は、上記の行列M_0を適用した列拡散演算（ＭｉｘＣｏｌｕｍｎ）、すなわち、

ＭＣ［M_0］

を実行する。

【０２６０】

このとき、定数ＣＯＮの条件は、この線形変換部４３７において適用する線形変換行列M_0と、定数ＣＯＮとの行列演算の結果として得られる全ての要素が非ゼロ、すなわちゼロでない値となることである。

このような定数ＣＯＮの例が、図５８に示す定数ＣＯＮであり、以下の要素構成を持つ４×４ステートとなる。

【０２６１】

【数１５】

con_0	con_4	con_8	con_{12}
con_1	con_5	con_9	con_{13}
con_2	con_6	con_{10}	con_{14}
con_3	con_7	con_{11}	con_{15}

＝

1	2	4	8
2	1	8	4
4	8	1	2
8	4	2	1

【０２６２】

上記設定を持つ４×４ステートの定数を行列M_0を適用した列拡散演算（ＭｉｘＣｏｌｕｍｎ）：Ｍｃ［M_0］を実行する線形変換部に隣接する排他的論理和部に入力して排他的論理和演算を実行する。この構成によって、線形変換部の線形変換による差分減少が防止される。この結果、最小差分アクティブＳボックスの数の減少を防止することが可能となり、各種の攻撃に対する耐性を高めた安全性の高い暗号処理構成が実現される。

【０２６３】

図５９は、上記の定数条件を満足する定数ＣＯＮの入力構成を持つ暗号処理装置の暗号処理部に対するラウンド鍵と定数ＣＯＮの入力構成と演算構成例を示す図である。

図５９において、平文Ｐは、左下から入力され、暗号文Ｃは右下から出力される。

図５９に示す暗号処理装置の暗号処理部４５１（Ｅ）は、図５６（ａ）に示す変換関数Ｅ，４３１に相当する。

また、線形変換部（Ｐ２）４５２は、図５６（ａ）に示す線形変換部４３２に相当する。

また、暗号処理部４５３（Ｅ$^{-1}$）は、図５６（ａ）に示す変換関数Ｅ$^{-1}$，４３３に相当する。

図５９に示す暗号処理装置構成は、インボリューション性を持つ暗号処理装置である。

【０２６４】

図５９に示すラウンド鍵入力例は、先に図４３を参照して説明した構成例に対応する。すなわち、暗号処理部に構成された排他的論理和部に対するラウンド鍵の入力順は、以下の通りである。

鍵Ｋ$_1$、
鍵Ｋ$_2$、
変換鍵Ｋｄ$_1$、
変換鍵Ｋｄ$_2$、
排他的論理和演算鍵Ｋ$_1$（＋）Ｋ$_2$、
排他的論理和演算変換鍵Ｋｄ$_1$（＋）Ｋｄ$_2$、
排他的論理和演算鍵Ｋ$_1$（＋）Ｋ$_2$、
排他的論理和演算鍵Ｋ$_1$（＋）Ｋ$_2$、
排他的論理和演算変換鍵Ｋｄ$_1$（＋）Ｋｄ$_2$、
排他的論理和演算鍵Ｋ$_1$（＋）Ｋ$_2$、
変換鍵Ｋｄ$_2$、
変換鍵Ｋｄ$_1$、
鍵Ｋ$_2$、
鍵Ｋ$_1$、

【０２６５】

なお、図４３に示す設定では、ラウンド演算部Ｒ$_6$に対して、排他的論理和演算鍵Ｋ$_1$（＋）Ｋ$_2$を入力する設定としているが、図５９に示す例では、線形変換部４５２の前後において、排他的論理和演算鍵Ｋ$_1$（＋）Ｋ$_2$、を繰り返し入力する設定である。

図５９に示す構成は、図４３に示すラウンド演算部Ｒ$_6$を、
線形変換部４５２と、線形変換部４５２前後の２つの、
排他的論理和演算鍵Ｋ$_1$（＋）Ｋ$_2$との排他的論理和部、
非線形変換部、
これらの変換部によって構成されるラウンド演算部とする設定した場合の構成に対応する。

【０２６６】

鍵スケジュール部のラウンド鍵供給部は、上記の順番で６種類の鍵を出力する。この鍵の入力シーケンスは、逆の順番も同一シーケンスとなる。

これは、平文Ｐから暗号文Ｃを生成する暗号処理における鍵入力順と、暗号文Ｃから平文Ｐを生成する復号処理における鍵入力順を同じ設定とすることが可能であることを意味する。すなわち、インボリューション性を持つ鍵入力シーケンスであり、暗号処理および復号処理に適用するハードウェアやプログラムの共通化が可能であり、装置の軽量化（小型化）に寄与する設定である。

【０２６７】

図５９に示す構成において、定数ＣＯＮは、暗号処理部４５３（Ｅ$^{-1}$）の排他的論理和部４６１、排他的論理和部４６３、排他的論理和部４６５、これらの各排他的論理和部に入力する。

【０２６８】

これらの定数ＣＯＮは、例えば、図５８を参照して説明した４×４ステートの定数ＣＯＮである。

また、これら３つの排他的論理和部４６１，４６３，４６５に隣接する線形変換部４６２，４６４，４６６は、いずれも前述の行列Ｍ$_0$を適用した列拡散演算（ＭｉｘＣｏｌｕｍｎ）、すなわち、
ＭＣ［Ｍ$_0$］

を実行する。

【０２６９】

この図５９に示す暗号処理部は、平文Ｐから暗号文Ｃを生成するシーケンスと、逆のシーケンスを実行することで、暗号文Ｃから平文Ｐを生成することも可能なインボリューション性を持つ構成であり、同一のハードウェア、あるいは同一プログラムを適用して暗号化処理と復号処理を行なうことが可能となる。

また、鍵入力シーケンスも暗号化と復号処理において同一のシーケンスとなるため、鍵スケジュール部の鍵供給処理も同一のハードウェア、あるいは同一プログラムを適用した処理として実行可能となる。

【０２７０】

上述した定数入力構成によって、最小差分アクティブＳボックスの数の減少を防止することが可能となり、各種の攻撃に対する耐性を高めた安全性の高い暗号処理構成が実現される。

なお、一般的な暗号処理におけるアクティブＳボックスに基づく評価処理と、本開示の定数入力構成におけるアクティブＳボックスに基づく評価処理とはやや異なっており、この点について、図６０、図６１を参照して説明する。

【０２７１】

前述したように共通鍵ブロック暗号に設定される非線形変換部には、ｓビット単位の非線形変換を実行するＳボックス（Ｓ－ｂｏｘ）が用いられる。

差分攻撃に対する耐性を図る指標として，差分の接続関係を表現した差分パスに含まれる差分アクティブＳボックスの最小数、すなわち、最小差分アクティブＳボックス数がある。

【０２７２】

一般的なブロック暗号において、非線形変換はＳボックスによる処理の部分のみである。図６０（Ａ），（Ｂ）に示すように、ブロック暗号を実行するブロック暗号装置に対して、特定の差分ΔＸを持つデータＰ１，Ｐ２を個別に入力して暗号処理結果Ｃ１，Ｃ２を得る。

【０２７３】

この２つの暗号処理（Ａ），（Ｂ）において、差分値が入力されるＳボックスをアクティブＳボックスと定義する。差分値が入力されるＳボックスを特定することで、解析容易性が高まる。すなわち攻撃に対する耐性が弱まることになる。

一般的には、この図６０に示すように所定の差分ΔＸを有する２つの入力Ｐ１，Ｐ２を設定した場合に発生する差分値が入力されるＳボックスの数をアクティブＳボックスと定義し、この数をカウントすることで安全性評価を行う。

【０２７４】

図６１は、上述した図５６（ａ）に示すインボリューション性を有するデータ変換部からなる暗号処理部を示している。

すなわち、暗号処理部は、

変換関数Ｅ，４３１、

線形変換部４３２、

変換関数E^{-1}，４３３、

これらのデータ変換部を有し、変換関数E^{-1}，４３３、は、変換関数Ｅ４３１の逆関数である。

【０２７５】

定数（ＣＯＮ）４３５は、変換関数E^{-1}，４３３に入力する。

なお、変換関数E^{-1}，４３３は、複数のラウンド関数によって構成され、定数（ＣＯＮ）は、１つ以上のラウンド関数部に対して入力する。

【０２７６】

暗号処理部は、変換関数Ｅ，４３１と、変換関数Ｅ，４３１の逆関数で変換関数E^{-1} ４３３をシーケンシャルに実行するインボリューション性を有し、変換関数Ｅ、または、

逆関数Ｅ$^{-1}$のいずれか一方のみにおいて、１回以上の定数を適用したラウンド演算を実行する。

【０２７７】

この構成において、変換関数Ｅ４３１に対して、線形変換部４３２側から入力Ｓ$_1$を入力して、出力Ｔを得る。

一方、変換関数Ｅ$^{-1}$，４３３に対して、同様に線形変換部４３２側から、上記の入力値Ｓ$_1$に対して差分値ΔＸを付加した入力Ｓ$_2$（＝Ｓ$_1$（＋）ΔＸ）を入力して、出力Ｔ（＋）ΔＹを得る。

【０２７８】

この２つの関数、すなわち、変換関数Ｅ４３１と、変換関数Ｅ４３１の逆関数である変換関数Ｅ$^{-1}$，４３３に対して、図６１に示すように、逆方向に差分ΔＸを有する入力値Ｓ１，Ｓ２を入力したときに、各関数において対応位置にあるＳボックス中、差分が入力されるＳボックスをアクティブＳボックスとする。

【０２７９】

図６２は、前述の図５９に示す暗号処理構成におけるアクティブＳボックスの数の算出構成を説明する図である。

図６２に示す暗号処理装置の暗号処理部４５１（Ｅ）の線形変換部４５２側から入力値Ｓ１を入力し、暗号処理部４５１（Ｅ）を適用した暗号処理を実行する。

一方、暗号処理部４５１（Ｅ）の逆関数である暗号処理部４５３（Ｅ$^{-1}$）の線形変換部４５２側から入力値Ｓ１に対して差分ΔＸを設定した入力値Ｓ２を入力し、暗号処理部４５３（Ｅ$^{-1}$）を適用した暗号処理を実行する。

この２つの暗号処理において各暗号処理部（Ｅ），（Ｅ$^{-1}$）において対応位置にあるＳボックス中、差分が入力されるＳボックスをアクティブＳボックスとする。

【０２８０】

なお、アクティブＳボックスの数が減少すると、例えばスライド攻撃やリフレクション攻撃と呼ばれる攻撃に対する耐性が弱くなり、アクティブＳボックスの数を一定以上に維持することで、これらの攻撃に対する耐性が高めることが可能となり、安全性を向上させることができる。上述した定数入力構成によって、アクティブＳボックスの数の減少を防止することが可能となり、各種の攻撃に対する耐性を高めた安全性の高い暗号処理構成が実現される。

【０２８１】

［６－３．定数挿入位置のバリエーションについて］

上述した実施例では、データ変換関数Ｅと、データ変換関数Ｅの逆関数Ｅ$^{-1}$をシーケンシャルに実行するインボリューション性を有する暗号処理装置に対して、関数Ｅ、または逆関数Ｅ$^{-1}$のいずれか一方のみに定数ＣＯＮを入力して定数を適用したラウンド演算を実行する構成について説明した。

このような設定により、定数ＣＯＮを入力する排他的論理和部に隣接する線形変換部、図５７に示す例では線形変換部４３７における線形変換処理の入出力値の差分の減少を防止でき、結果として、最小差分アクティブＳボックスの数を所定数以上に維持することが可能となる。

【０２８２】

この定数入力構成は、関数Ｅ、または逆関数Ｅ$^{-1}$のいずれか一方のみに限定されず、関数Ｅ、および逆関数Ｅ$^{-1}$の双方において１回以上の定数を適用したラウンド演算を実行する構成としてもよい。ただし、定数適用位置は、関数Ｅ、および逆関数Ｅ$^{-1}$の対応位置ではない、対応位置からずれた位置（非対応位置）とする。

【０２８３】

この定数入力構成を持つ暗号処理装置の構成例を図６３に示す。

図６３において、平文Ｐは、左下から入力され、暗号文Ｃは右下から出力される。

図６３に示す暗号処理装置の暗号処理部４５１（Ｅ）は、図５６（ａ）に示す変換関数Ｅ，４３１に相当する。

また、線形変換部（Ｐ２）４５２は、図５６（ａ）に示す線形変換部４３２に相当する。

また、暗号処理部４５３（Ｅ$^{-1}$）は、図５６（ａ）に示す変換関数Ｅ$^{-1}$，４３３に相当する。

図６３に示す暗号処理部の構成は、インボリューション性を持つ暗号処理部である。

【０２８４】

図６３に示すラウンド鍵入力例は、先に図４３を参照して説明した構成例に対応する。すなわち、暗号処理部に構成された排他的論理和部に対するラウンド鍵の入力順は、以下の通りである。

鍵Ｋ$_1$、
鍵Ｋ$_2$、
変換鍵Ｋｄ$_1$、
変換鍵Ｋｄ$_2$、
排他的論理和演算鍵Ｋ$_1$（＋）Ｋ$_2$、
排他的論理和演算変換鍵Ｋｄ$_1$（＋）Ｋｄ$_2$、
排他的論理和演算鍵Ｋ$_1$（＋）Ｋ$_2$、
排他的論理和演算鍵Ｋ$_1$（＋）Ｋ$_2$、
排他的論理和演算変換鍵Ｋｄ$_1$（＋）Ｋｄ$_2$、
排他的論理和演算鍵Ｋ$_1$（＋）Ｋ$_2$、
変換鍵Ｋｄ$_2$、
変換鍵Ｋｄ$_1$、
鍵Ｋ$_2$、
鍵Ｋ$_1$、

【０２８５】

なお、図４３に示す設定では、ラウンド演算部Ｒ$_6$に対して、排他的論理和演算鍵Ｋ$_1$（＋）Ｋ$_2$を入力する設定としているが、図６３に示す例では、線形変換部４５２の前後において、排他的論理和演算鍵Ｋ$_1$（＋）Ｋ$_2$、を繰り返し入力する設定である。

図６３に示す構成は、図４３に示すラウンド演算部Ｒ$_6$を、
線形変換部４５２と、線形変換部４５２前後の２つの、
排他的論理和演算鍵Ｋ$_1$（＋）Ｋ$_2$との排他的論理和部、
非線形変換部、
これらの変換部によって構成されるラウンド演算部とする設定した場合の構成に対応する。

【０２８６】

鍵スケジュール部のラウンド鍵供給部は、上記の順番で６種類の鍵を出力する。この鍵の入力シーケンスは、逆の順番も同一シーケンスとなる。

これは、平文Ｐから暗号文Ｃを生成する暗号処理における鍵入力順と、暗号文Ｃから平文Ｐを生成する復号処理における鍵入力順を同じ設定とすることが可能であることを意味する。すなわち、インボリューション性を持つ鍵入力シーケンスであり、暗号処理および復号処理に適用するハードウェアやプログラムの共通化が可能であり、装置の軽量化（小型化）に寄与する設定である。

【０２８７】

図６３に示す構成において、定数ＣＯＮは、
暗号処理部４５１（Ｅ）の排他的論理和部４７１、排他的論理和部４７２、これらの各排他的論理和部に入力する。
さらに、暗号処理部４５３（Ｅ$^{-1}$）の排他的論理和部４７３にも入力する。

【０２８８】

定数ＣＯＮは、例えば、図５８を参照して説明した４×４ステートの定数ＣＯＮである。

また、これら３つの排他的論理和部４７１，４７２，４７３に隣接する線形変換部４８

１，４８２，４８３は、いずれも前述の行列M_0を適用した列拡散演算（ＭｉｘＣｏｌｕｍｎ）、すなわち、
ＭＣ［M_0］
を実行する。
【０２８９】
このように、定数入力構成は、関数Ｅ、または逆関数E^{-1}のいずれか一方のみに限定されず、関数Ｅ、および逆関数E^{-1}の双方において１回以上の定数を適用したラウンド演算を実行する構成としてもよい。ただし、定数適用位置は、関数Ｅ、および逆関数E^{-1}の対応位置ではない、ずれた位置（非対応位置）とする。
【０２９０】
この図６３に示す暗号処理部は、平文Ｐから暗号文Ｃを生成するシーケンスと、逆のシーケンスを実行することで、暗号文Ｃから平文Ｐを生成することも可能なインボリューション性を持つ構成であり、同一のハードウェア、あるいは同一プログラムを適用して暗号化処理と復号処理を行なうことが可能となる。
また、鍵入力シーケンスも暗号化と復号処理において同一のシーケンスとなるため、鍵スケジュール部の鍵供給処理も同一のハードウェア、あるいは同一プログラムを適用した処理として実行可能となる。
【０２９１】
［７．非線形変換部に適用するＳボックス（Ｓ－ｂｏｘ）の具体的構成例について］
次に、非線形変換部に適用するＳボックス（Ｓ－ｂｏｘ）の具体的構成例について説明する。
【０２９２】
例えば図１９に示す暗号処理装置１００において、暗号処理部１２０のインボリューション性、すなわち、平文Ｐから暗号文Ｃを生成して出力するハードウェア、あるいはプログラムと、暗号文Ｃから平文Ｐを生成して出力するハードウェアあるいはプログラムを同一とするためには、暗号処理部１２０に構成される非線形変換部１２２についてもインボリューション性が要求される。
【０２９３】
以下、暗号処理部１２０に構成される非線形変換部１２２のインボリューション性を有する構成例について説明する。
先に、図２２を参照して説明したように、図１９に示す暗号処理装置１００の暗号処理部１２０内の非線形変換部１２２は、例えば図２２（１）に示すように、複数のＳボックス（Ｓ－ｂｏｘ）を有する構成である。
各Ｓボックスは例えば、４ビット入出力構成を持つ非線形変換部であり、１６個のＳボックスによる並列処理によって４×１６＝６４ビットの非線形変換処理を実行する。
【０２９４】
この４ビット入出力のＳボックス（Ｓ－ｂｏｘ）がインボリューション性を持つ構成であることが必要である。
すなわち４ビット入力値に対して、ある４ビット出力値が得られた場合、その４ビット出力値を同一のＳボックスに入力した場合、元の４ビット入力値が得られる構成であることが必要である。
【０２９５】
なお、インボリューション性を有する関数ｆ（ｘ）は、すべての入力値ｘに対して、
ｆ（ｆ（ｘ））＝ｘ
上記を満たす関数である。
暗号処理部１２０に構成される非線形変換部１２２は、このインボリューション性を有する関数ｆ（ｘ）であることが要求される。
【０２９６】
図６４以下を参照してインボリューション性を持つ４ビット入出力Ｓボックスの構成例について説明する。

【０２９７】
図６４（１）は、先に説明した図２２（１）と同様、非線形変換部の構成例を示す図である。
すなわち、図１９に示す暗号処理装置１００の暗号処理部１２０に構成される非線形変換部１２２の構成例である。非線形変換部１２２は、非線形変換処理を実行するＳボックス（Ｓ－ｂｏｘ）を複数配置した構成を有する、
各Ｓボックスは４ビットデータの非線形変換を実行する。
【０２９８】
図６４（２）は、非線形変換部内に構成される１つのＳボックス（Ｓ－ｂｏｘ）の構成を示している。Ｓボックス（Ｓ－ｂｏｘ）は、
非線形変換層１，５２１、
線形変換層５２２、
非線形変換層２，５２３、
これら３つの層に区分することができる。
なお、非線形変換層２，５２３は、非線形変換層１，５２１の逆関数となっている。
【０２９９】
図６５に、具体的なＳボックス（Ｓ－ｂｏｘ）の回路構成例を示す。
図６５に示すように、
非線形変換層１，５２１は、２つの排他的論理和演算部（ＸＯＲ）と、２つの基本演算子によって構成される。
なお、図６５に示す例では、基本演算子としてＮＯＲ回路を設定した例を示しているが、基本演算子は、ＡＮＤ回路、ＯＲ回路、ＮＡＮＤ回路、いずれかの２入力１出力の演算を行う基本演算子に置き換え可能である。
また、２つの基本演算子は、同一の基本演算子の組み合わせとしても、異なる基本演算子の組み合わせとしてもよい。
【０３００】
線形変換層５２２は、入力４ビットの入れ替え処理を行なう線形変換層であり、基本的にインボリューション性を有する。
非線形変換層２，５２３は、非線形変換層１，５２１の逆関数によって構成される。
【０３０１】
これらの３つの層によって構成されるＳボックス回路は、インボリューション性を有する非線形変換回路となる。
図６５の下部に、図６５に示すＳボックスに対する入力値（ｉｎ）と出力値（ｏｕｔ）との対応関係データを示す。
なお、入出力値はいずれも４ビットデータであり、０００００～１１１１のデータである。図６５に示す表は、０００００～１１１１を１０進法で示す０～１５の入力値と出力値との対応表として示している。
この表から理解されるように、任意の入力値Ｘから得られる出力値Ｙを、入力値Ｙとして得られる出力値は元の入力値Ｘとなる。
【０３０２】
すなわち、図６５に示す４ビット入出力Ｓボックスは、インボリューション性を有する持つ非線形変換回路である。
【０３０３】
図６６に、このＳボックス（Ｓ－ｂｏｘ）によるデータ変換式を示す。
Ｓボックス（Ｓ－ｂｏｘ）に対する４ビット入力を、
a_{in}，b_{in}，c_{in}，d_{in}とし、
Ｓボックス（Ｓ－ｂｏｘ）からの４ビット出力を、
a_{out}，b_{out}，c_{out}，d_{out}、
とする。
Ｓボックス（Ｓ－ｂｏｘ）によるデータ変換式は、以下の通りである。

【０３０４】
【数１６】

$$a_{out} = c_{in} \oplus\sim\left(\left(\mathrm{d}_{\mathrm{in}} \oplus\sim\left(\mathrm{a}_{\mathrm{in}} \mid b_{in}\right)\right) \mid \left(\mathrm{a}_{\mathrm{in}} \oplus\sim\left(\mathrm{b}_{\mathrm{in}} \mid c_{in}\right)\right)\right)$$
$$b_{out} = d_{in} \oplus\sim\left(\mathrm{a}_{\mathrm{in}} \mid b_{in}\right)$$
$$c_{out} = a_{in} \oplus\sim\left(\mathrm{b}_{\mathrm{in}} \mid c_{in}\right)$$
$$d_{out} = b_{in} \oplus\sim\left(\mathrm{b}_{\mathrm{out}} \mid a_{out}\right)$$
$$= b_{in} \oplus\sim\left(\left(d_{in} \oplus\sim\left(\mathrm{a}_{\mathrm{in}} \mid b_{in}\right)\right) \mid \left(c_{in} \oplus\sim\left(\left(\mathrm{d}_{\mathrm{in}} \oplus\sim\left(\mathrm{a}_{\mathrm{in}} \mid b_{in}\right)\right) \mid \left(\mathrm{a}_{\mathrm{in}} \oplus\sim\left(\mathrm{b}_{\mathrm{in}} \mid c_{in}\right)\right)\right)\right)\right)$$

【０３０５】
なお、上記式において、～（ｘ｜ｙ）は、（）内の値の否定（ＮＯＴ）を示す。具体的には、ＮＯＲ回路に対する入力値がｘとｙである場合のＮＯＲ回路の出力値を示す。
【０３０６】
上記演算式によって表現されるデータ変換を実行するＳボックスは、インボリューション性を有する。
また、この図６６に示すＳボックス回路は、差分確率、線形確率がいずれも2^{-2}であり、十分な安全性を有する。
図６６に示すＳボックスは、４つの排他的論理和演算子（ＸＯＲ）と４つのＮＯＲ回路から構成され、ハードウェア回路上に必要なゲート数は１３ゲートとなる。
なお、必要なゲート数は、排他的論理和演算子（ＸＯＲ）＝２．２５ゲート、ＮＯＲ回路＝１ゲートとして算出している。
【０３０７】
例えば図５４を参照して説明した暗号処理部構成、すなわち、
変換関数Ｅ４１１、
線形変換部４１２、
変換関数E^{-1}４１３、
これらの構成を持つ暗号処理部内の変換関数Ｅ４１１と、変換関数E^{-1}４１３内の非線形変換部に図６４～図６６に示すＳボックスを利用した構成とすることで、暗号処理部全体のインボリューション性が実現される。
【０３０８】
なお、図６５、図６６を参照して説明したＳボックスは、
非線形変換層１、
線形変換層、
非線形変換層２、
これらの３層構成となっているが、この３層構成を持つＳボックスの他の例について以下説明する。
【０３０９】
なお、上記３層構成中の線形変換層の必要条件として、インボリューション性を有し、かつ、入出力ビットが不変となる置換部を有していないビット置換を実行する構成であることが必要である。
この線形変換層の条件について、図６７を参照して説明する。
【０３１０】
図６７には４ビット入出力Ｓボックスにおける線形変換層の設定例を示している。
図６５、図６６を参照して説明したＳボックスと同様、
非線形変換層１、
線形変換層、
非線形変換層２、
これらの３層構成を持つ４ビット入出力Ｓボックスにおいて線形変換層は、例えば、図

６７（１）に示す設定のいずれかとする。

【０３１１】

線形変換層に対する入力４ビットをＸ＝（ｘ０，ｘ１，ｘ２，ｘ３）、出力４ヒビットＹ＝（ｙ０，ｙ１，ｙ２，ｙ３）としたとき、

インボリューション性を有し、かつ、入出力ビットが同一とならない設定、すなわち、

ｙｉ≠ｘｉ

ただしｉ＝０，１，２，３、

上記式が成立することが線形変換層の条件となる。

図６７（２）に示す線形変換構成は、上記条件を満たさず、不適合となる。

【０３１２】

４ビット置換を実行する線形変換層の置換関数Ｐ４の条件を、式で示すと以下のように示すことができる。

（ａ）Ｐ４（Ｐ４（Ｘ））＝Ｘ

（ｂ）　ｙｉ≠ｘｉ　　ただしｉ＝０，１，２，３、

上記（ａ）は置換関数Ｐ４がインボリューション性を有することを示す条件式である。

（ｂ）は、入出力ビットが同一とならないことを示す条件式である。

線形変換層は、上記条件を満たす置換処理を行なう構成であることが必要である。

【０３１３】

４ビット置換を行うＳボックスの例として、図６５、図６６を参照して構成と異なる構成を持つ例について、図６８以下を参照して説明する。

【０３１４】

図６８に示すＳボックスは、

非線形変換層１，５３１、

線形変換層５３２、

非線形変換層２，５３３、

これらの３層構成を持つ４ビット入出力Ｓボックスである。

【０３１５】

非線形変換層１，５３１は、２つの排他的論理和演算部（ＸＯＲ）と、ＮＡＮＤ回路とＮＯＲ回路によって構成されている。

線形変換層５３２は、入力４ビットの入れ替え処理を行なう線形変換層であり、インボリューション性を有する。

非線形変換層２，５３３は、非線形変換層１，５３１の逆関数によって構成される。

【０３１６】

これらの３つの層によって構成されるＳボックス回路は、インボリューション性を有する非線形変換回路となる。

図６８の下部に、図６８に示すＳボックスに対する入力値（ｉｎ）と出力値（ｏｕｔ）との対応関係データを示す。

なお、入出力値はいずれも４ビットデータであり、００００～１１１１のデータである。図６８に示す表は、００００～１１１１を１０進法で示す０～１５の入力値と出力値との対応表として示している。

この表から理解されるように、任意の入力値Ｘから得られる出力値Ｙを、入力値Ｙとして得られる出力値は元の入力値Ｘとなる。

【０３１７】

すなわち、図６８に示す４ビット入出力Ｓボックスは、インボリューション性を有する持つ非線形変換回路である。

【０３１８】

図６９に、このＳボックス（Ｓ－ｂｏｘ）によるデータ変換式を示す。

Ｓボックス（Ｓ－ｂｏｘ）に対する４ビット入力を、

a_{in}，b_{in}，c_{in}，d_{in}とし、

Ｓボックス（Ｓ－ｂｏｘ）からの４ビット出力を、

a_{out}，b_{out}，c_{out}，d_{out}、
とする。
Ｓボックス（Ｓ－ｂｏｘ）によるデータ変換式は、以下の通りである。
【０３１９】
【数１７】

$$a_{out} = d_{in} \oplus\!\!\!\sim (b_{out} \mid d_{out}) = d_{in} \oplus\!\!\!\sim ((c_{in} \oplus\!\!\!\sim (a_{in} \,\&\, b_{in})) \mid (a_{in} \oplus\!\!\!\sim (b_{in} \mid d_{in})))$$
$$b_{out} = c_{in} \oplus\!\!\!\sim (a_{in} \,\&\, b_{in})$$
$$c_{out} = b_{in} \oplus (a_{out} \,\&\, b_{out})$$
$$= b_{in} \oplus ((d_{in} \oplus\!\!\!\sim ((c_{in} \oplus\!\!\!\sim (a_{in} \,\&\, b_{in})) \mid (a_{in} \oplus\!\!\!\sim (b_{in} \mid d_{in})))) \,\&\, (c_{in} \oplus\!\!\!\sim (a_{in} \,\&\, b_{in})))$$
$$d_{out} = a_{in} \oplus\!\!\!\sim (b_{in} \mid d_{in})$$

【０３２０】
なお、上記式において、
～（ｘ｜ｙ）は、ＮＯＲ回路に対する入力値がｘとｙである場合のＮＯＲ回路の出力値を示す。
～（ｘ＆ｙ）は、ＮＡＮＤ回路に対する入力値がｘとｙである場合のＮＡＮＤ回路の出力値を示す。
【０３２１】
上記演算式によって表現されるデータ変換を実行するＳボックスは、インボリューション性を有する。
また、この図６９に示すＳボックス回路も、差分確率、線形確率がいずれも2^{-2}であり、十分な安全性を有する。
図６９に示すＳボックスは、４つの排他的論理和演算子（ＸＯＲ）と２つのＮＯＲ回路と２つのＮＡＮＤ回路から構成される
【０３２２】
例えば図５４を参照して説明した暗号処理部構成、すなわち、
変換関数Ｅ４１１、
線形変換部４１２、
変換関数E^{-1}４１３、
これらの構成を持つ暗号処理部内の変換関数Ｅ４１１と、変換関数E^{-1}４１３内の非線形変換部に図６９に示すＳボックスを利用した構成とすることで、暗号処理部全体のインボリューション性が実現される。
【０３２３】
次に、図７０を参照してさらに異なる構成を持つ４ビット入出力Ｓボックスの例について説明する。
図７０に示すＳボックスは、
非線形変換層１，５４１、
線形変換層５４２、
非線形変換層２，５４３、
これらの３層構成を持つ４ビット入出力Ｓボックスである。
【０３２４】
非線形変換層１，５４１は、２つの排他的論理和演算部（ＸＯＲ）と、２つのＯＲ回路によって構成されている。
線形変換層５４２は、入力４ビットの入れ替え処理を行なう線形変換層であり、インボリューション性を有する。
非線形変換層２，５４３は、非線形変換層１，５４１の逆関数によって構成される。
【０３２５】

これらの３つの層によって構成されるＳボックス回路は、インボリューション性を有する非線形変換回路となる。

図７０の下部に、図６８に示すＳボックスに対する入力値（ｉｎ）と出力値（ｏｕｔ）との対応関係データを示す。

なお、入出力値はいずれも４ビットデータであり、０００００～１１１１のデータである。図７０に示す表は、０００００～１１１１を１０進法で示す０～１５の入力値と出力値との対応表として示している。

この表から理解されるように、任意の入力値Ｘから得られる出力値Ｙを、入力値Ｙとして得られる出力値は元の入力値Ｘとなる。

【０３２６】

すなわち、図７０に示す４ビット入出力Ｓボックスは、インボリューション性を有する持つ非線形変換回路である。

【０３２７】

図７１に、このＳボックス（Ｓ－ｂｏｘ）によるデータ変換式を示す。

Ｓボックス（Ｓ－ｂｏｘ）に対する４ビット入力を、

a_{in}，b_{in}，c_{in}，d_{in}とし、

Ｓボックス（Ｓ－ｂｏｘ）からの４ビット出力を、

a_{out}，b_{out}，c_{out}，d_{out}、

とする。

Ｓボックス（Ｓ－ｂｏｘ）によるデータ変換式は、以下の通りである。

【０３２８】

【数１８】

$$a_{out} = c_{in} \oplus \left(\left(\mathrm{d}_{\mathrm{in}} \oplus \left(\mathrm{a}_{\mathrm{in}} \mid b_{in} \right) \right) \mid \left(\mathrm{a}_{\mathrm{in}} \oplus \left(\mathrm{b}_{\mathrm{in}} \mid c_{in} \right) \right) \right)$$

$$b_{out} = d_{in} \oplus \left(\mathrm{a}_{\mathrm{in}} \mid b_{in} \right)$$

$$c_{out} = a_{in} \oplus \left(\mathrm{b}_{\mathrm{in}} \mid c_{in} \right)$$

$$d_{out} = b_{in} \oplus \left(\mathrm{b}_{\mathrm{out}} \mid a_{out} \right)$$

$$= b_{in} \oplus \left(\left(d_{in} \oplus \left(\mathrm{a}_{\mathrm{in}} \mid b_{in} \right) \right) \mid \left(c_{in} \oplus \left(\left(\mathrm{d}_{\mathrm{in}} \oplus \left(\mathrm{a}_{\mathrm{in}} \mid b_{in} \right) \right) \mid \left(\mathrm{a}_{\mathrm{in}} \oplus \left(\mathrm{b}_{\mathrm{in}} \mid c_{in} \right) \right) \right) \right) \right)$$

【０３２９】

なお、上記式において、

（ｘ｜ｙ）は、ＯＲ回路に対する入力値がｘとｙである場合のＯＲ回路の出力値を示す。

【０３３０】

上記演算式によって表現されるデータ変換を実行するＳボックスは、インボリューション性を有する。

また、この図７１に示すＳボックス回路も、差分確率、線形確率がいずれも2^{-2}であり、十分な安全性を有する。

図７１に示すＳボックスは、４つの排他的論理和演算子（ＸＯＲ）と４つのＯＲ回路から構成される

【０３３１】

例えば図５４を参照して説明した暗号処理部構成、すなわち、

変換関数Ｅ４１１、

線形変換部４１２、

変換関数E^{-1}４１３、

これらの構成を持つ暗号処理部内の変換関数Ｅ４１１と、変換関数E^{-1}４１３内の非

線形変換部に図７１に示すＳボックスを利用した構成とすることで、暗号処理部全体のインボリューション性が実現される。

【０３３２】

［８．暗号処理装置の具体例について］

次に、上述した説明に従った各構成、すなわち、以下の各構成を全て有する暗号処理装置の全体構成例について説明する。

（１）安全性を高めた共通鍵暗号処理の全体構成（図１９～図２２）

（２）複数の異なる線形変換行列を適用した線形変換を実行する構成（図２３～図３１）

（３）ベース鍵と変換鍵を使用して生成したラウンド鍵を提供し、インボリューション性、フルディフュージョン性を実現する鍵スケジュール部の構成（図３２～図５２）

（４）定数入力構成（図５３～図６３）

（５）インボリューション性を持つＳボックスを適用した非線形変換部の構成（図６４～図７１）

【０３３３】

図７２に示す暗号処理装置７００は、上記の各構成を全て備えた暗号処理装置の一例を示す図である。

暗号処理装置７００は、鍵スケジュール部７２０と、暗号処理部７５０を有する。

暗号処理部７５０は、排他的論理和部７５１、非線形変換部７５２、線形変換部７５３の各データ変換部を有し、これらの処理を繰り返し実行する構成を有する。

一方、鍵スケジュール部７２０は、暗号処理部７５０に構成された排他的論理和部の各々に対してラウンド鍵ＲＫｎを出力して、変換対象データとの排他的論理和演算を実行させる。

【０３３４】

鍵スケジュール部７２０は、ラウンド鍵供給部７２１と定数供給部（定数レジスタ）７２５を有する。

また、ラウンド鍵供給部７２１は、秘密鍵Ｋを格納した鍵レジスタ７２２と、鍵変換部７２３を有する。

【０３３５】

図７２に示す暗号処理装置７００の暗号処理部７５０は、例えば６４ビットの平文Ｐを入力して、６４ビットの暗号文Ｃを出力する。また、この同じ暗号処理部７５０を適用して、暗号文Ｃを入力して平文Ｐを出力することが可能である。

変換データが６４ビットである場合、各ラウンド鍵ＲＫｎも６４ビットである。

なお、これらの各６４ビットデータは、いずれも各要素４ビットの１６要素からなる４×４ステートである。

【０３３６】

なお、暗号処理部７５０は、平文Ｐを入力データとしてラウンド演算を繰り返して出力データとしての暗号文Ｃを出力可能であるとともに、暗号文Ｃを入力データとして、ラウンド演算の実行シーケンスを逆順に設定したデータ変換処理により出力データとして前記平文Ｐを生成可能なインボリューション性を有する構成である。

【０３３７】

平文Ｐから暗号文Ｃを生成する場合は、図に示す暗号処理部７５０の上段から下段に向けて各変換処理を実行する。

一方、暗号文Ｃから平文Ｐを生成する場合は、図に示す暗号処理部７５０の下段から上段に向けて各変換処理を実行する。

【０３３８】

また、鍵スケジュール部７２０のラウンド鍵供給部７２１は、平文Ｐから暗号文Ｃを生成する場合の鍵供給シーケンスと、暗号文Ｃから平文Ｐを生成する場合の鍵供給シーケンスが一致するインボリューション性を有する鍵供給処理を行なう構成である。なお、鍵スケジュール部７２０は、暗号処理部７５０に対する鍵供給処理に際して、供給鍵の一部に

定数による演算を施し、演算結果である鍵データを暗号処理部７５０に出力する。

【０３３９】

このように、図７２に示す暗号処理部７５０の構成は、先に図５９を参照して説明した構成と同様、変換関数Ｅ、線形変換関数、変換関数$Ｅ^{-1}$のシーケンスで各変換関数が設定されており、インボリューション性を有する構成である。

【０３４０】

暗号処理部７５０には、先に項目［４．暗号処理部の線形変換部の構成と処理について］において、図２３～図３１を参照して説明したように、３種類の異なる線形変換処理を実行する線形変換処理部が設定されている。

すなわち、

線形変換部Ｐ１、

線形変換部Ｐ２、

線形変換部Ｐ３、

これらの３つの異なる線形変換部を有し、暗号処理において、ラウンド毎に実行する線形変換処理を変更する。すなわち、連続ラウンドでは同じ線形変換処理を行なわない設定としている。

【０３４１】

線形変換部Ｐ１は、４×４ステートの入力データの各列の要素に対して、各列単位で、１つの行列$Ｍ_0$を適用した行列演算を行う。

これは、先に、図２４、図２５を参照して説明した列拡散演算（ＭｉｘＣｏｌｕｍ）である。

すなわち、線形変換処理部Ｐ１は、

$ＭＣ［Ｍ_0］$

上記式によって示される列拡散演算（ＭＣ）を実行する。

なお、$ＭＣ［Ｍ_0］$は、ステートの各列に対して、同一の行列$Ｍ_0$を適用した行列演算を示す式であり、ステートの各列に対して適用する行列を個別に示した式、

$ＭＣ［Ｍ_0，Ｍ_0，Ｍ_0，Ｍ_0］$

上記式と同じ意味である。

【０３４２】

線形変換部Ｐ２は、先に図２４、図２６等を参照して説明したように、４×４ステートの入力データの各行の要素に対して、各行単位で異なる行列を適用した行列演算を行う。上位の第１行から第４行に対して、以下の行列を適用した行列演算を実行する。

第１行：適用行列$Ｍ_0$、

第２行：適用行列$Ｍ_1$、

第３行：適用行列$Ｍ_2$、

第４行：適用行列$Ｍ_3$、

すなわち、線形変換処理部Ｐ２は、

$ＭＲ［Ｍ_0，Ｍ_1，Ｍ_2，Ｍ_3］$

上記式によって示される行拡散演算（ＭｉｘＲｏｗ）を実行する。

【０３４３】

線形変換処理部Ｐ３も、線形変換処理部Ｐ２と同様、図２４（２）に示すように、４×４ステートの入力データの各行の要素に対して、各行単位で異なる行列を適用した行列演算を行う。線形変換処理Ｐ３は、線形変換処理Ｐ２とは異なり、上位の第１行から第４行に対して、以下の行列を適用した行列演算を実行する。

第１行：適用行列$Ｍ_2$、

第２行：適用行列$Ｍ_0$、

第３行：適用行列$Ｍ_1$、

第４行：適用行列$Ｍ_3$、

これは、先に、図２７を参照して説明した行拡散演算（ＭｉｘＲｏｗ）である。

すなわち、線形変換処理部Ｐ３は、

ＭＲ［M_2，M_0，M_1，M_3］
上記式によって示される行拡散演算（ＭｉｘＲｏｗ）を実行する。
【０３４４】
これらの複数の異なる線形変換処理を組み合わせて利用し、ラウンド毎に実行する線形変換処理を切り換えることで、先に図３０を参照して説明したように、アクティブＳボックスの数を増加させることが可能となり、より安全性の高い暗号処理（暗号化処理および復号処理）が実現される。
【０３４５】
なお、図７２に示す暗号処理部７５０は、以下の各データ変換部を順次適用したデータ変換処理を実行する。
ラウンド鍵RK_1との排他的論理和演算を行う排他的論理和部、
非線形変換部Ｓ、
線形変換部Ｐ１、
ラウンド鍵RK_2との排他的論理和演算を行う排他的論理和部、
非線形変換部Ｓ、
線形変換部Ｐ２、
ラウンド鍵RK_3との排他的論理和演算を行う排他的論理和部、
非線形変換部Ｓ、
線形変換部Ｐ１、
ラウンド鍵RK_4との排他的論理和演算を行う排他的論理和部、
非線形変換部Ｓ、
線形変換部Ｐ３、
ラウンド鍵RK_5との排他的論理和演算を行う排他的論理和部、
非線形変換部Ｓ、
線形変換部Ｐ１、
ラウンド鍵RK_6との排他的論理和演算を行う排他的論理和部、
【０３４６】
非線形変換部Ｓ、
ラウンド鍵RK_7との排他的論理和演算を行う排他的論理和部、
線形変換部Ｐ２、
ラウンド鍵RK_7との排他的論理和演算を行う排他的論理和部、
非線形変換部Ｓ、
【０３４７】
ラウンド鍵RK_8との排他的論理和演算を行う排他的論理和部、
線形変換部Ｐ１、
非線形変換部Ｓ、
ラウンド鍵RK_9との排他的論理和演算を行う排他的論理和部、
線形変換部Ｐ３、
非線形変換部Ｓ、
ラウンド鍵RK_{10}との排他的論理和演算を行う排他的論理和部、
線形変換部Ｐ１、
非線形変換部Ｓ、
ラウンド鍵RK_{11}との排他的論理和演算を行う排他的論理和部、
線形変換部Ｐ２、
非線形変換部Ｓ、
ラウンド鍵RK_{12}との排他的論理和演算を行う排他的論理和部、
線形変換部Ｐ１、
非線形変換部Ｓ、
ラウンド鍵RK_{13}との排他的論理和演算を行う排他的論理和部、
【０３４８】

このラウンド演算実行構成は、先に図５９を参照して説明したと同様の構成であり、
変換関数Ｅ、
線形変換部、
変換関数$Ｅ^{-1}$、
これらのシーケンスに設定され、インボリューション性を有する。
なお、各非線形変換部は、先に図６４～図６６を参照して説明したインボリューション性を有するＳボックスによって構成されている。

【０３４９】

鍵スケジュール部７２０のラウンド鍵供給部７２１は、鍵レジスタ７２２と鍵変換部７２３を有する。このラウンド鍵供給部７２１の実行する処理は、先に項目［５．鍵スケジュール部の構成と処理について］において図３２～図４３を参照して説明した処理である。
すなわち、ベース鍵と変換鍵を使用して生成したラウンド鍵を暗号処理部７５０に提供する。このラウンド鍵供給構成では、インボリューション性、フルディフュージョン性を実現している。

【０３５０】

鍵レジスタに格納された秘密鍵Ｋは６４ビットのベース鍵$Ｋ_{1}$と$Ｋ_{2}$との連結データである１２８ビット鍵データである。
鍵変換部７２３は、ベース鍵$Ｋ_{1}$に基づく変換鍵$Ｋｄ_{1}$を生成し、ベース鍵$Ｋ_{2}$に基づく変換処理によって変換鍵$Ｋｄ_{2}$を生成する。
この変換処理を、変換関数Ｇ、および逆関数$Ｇ^{-1}$を用いて式で示すと以下の通りとなる。
$Ｋｄ_{1}＝Ｇ（Ｋ_{1}）$
$Ｋ_{1}＝Ｇ^{-1}（Ｋｄ_{1}）$
なお、
ここで、$Ｇ＝Ｇ^{-1}$
が成立する。
すなわち、鍵変換部７２３において鍵変換に適用するデータ変換関数Ｇは、インボリューション性、すなわち、図３３に示すように、順方向関数Ｇと逆方向関数$Ｇ^{-1}$とが同じ関数であるという性質を持つ。

【０３５１】

なお、この鍵変換処理には、先に図３４等を参照して説明したように、アダマール（Ｈａｄａｍａｒｄ）ＭＤＳ行列$Ｍ_{D}$を適用して実行する。
具体的には、図３４に示すステップＳ１の列拡散演算と、ステップＳ２の行拡散演算を実行する。
ステップＳ１の列拡散演算は、以下の演算式によって示される。
$ＭＣ［Ｍ_{D}］＝ＭＣ［Ｍ_{D}，Ｍ_{D}，Ｍ_{D}，Ｍ_{D}］$
また、ステップＳ２の行拡散演算は、以下の演算式によって示される。
$ＭＲ［Ｍ_{D}］＝ＭＲ［Ｍ_{D}，Ｍ_{D}，Ｍ_{D}，Ｍ_{D}］$

【０３５２】

すなわち、図３４のステップＳ１の列拡散演算は、４ビット要素からなる４×４のステート表現データの４つの全ての列に対して、同一のアダマール（Ｈａｄａｍａｒｄ）ＭＤＳ行列$Ｍ_{D}$を適用した行列演算を実行する。
また、ステップＳ２の行拡散演算は、４ビット要素からなる４×４のステート表現データの４つの全ての行に対して、同一のアダマール（Ｈａｄａｍａｒｄ）ＭＤＳ行列$Ｍ_{D}$を適用した行列演算を実行する。

【０３５３】

列拡散演算$ＭＣ［Ｍ_{D}］$と、行拡散演算$ＭＲ［Ｍ_{D}］$の連続処理からなる関数Ｇは、インボリューション性を持ち、順方向関数Ｇと、逆方向関数$Ｇ^{-1}$が同一であるため、２回繰り返すことで、元の値が算出される。

【０３５４】
さらに、鍵変換部７２３において実行する行列M_Dを適用した列拡散演算ＭＣ［M_D］と、行列M_Dを適用した行拡散演算ＭＲ［M_D］により、入出力ステート全要素間のデータ拡散（ｄｉｆｆｕｓｉｏｎ）、すなわち「全拡散（ｆｕｌｌ　ｄｉｆｆｕｓｉｏｎ）変換」が行われる。
このフルディフュージョン性を持つラウンド鍵を暗号処理部に入力して変換対象データとの排他的論理和を実行することで、変換データの拡散性能も向上し、より安全性の高い暗号処理が実現される。
【０３５５】
なお、ラウンド鍵供給部７２１の実行するラウンド鍵供給構成は、
（１）全拡散（ｆｕｌｌ　ｄｉｆｆｕｓｉｏｎ）変換を実現するフルディフュージョン性、
（２）順方向関数Ｇと、逆方向関数G^{-1}が同一であるインボリューション性、
これら２つの性質を持つ。
これらの２つの特性により、先に説明したように、以下の効果がもたらされる。
【０３５６】
全拡散（ｆｕｌｌ　ｄｉｆｆｕｓｉｏｎ）変換を実現するフルディフュージョン性に基づいて、変換対象データに対する鍵データの構成情報の拡散が少ないラウンド数で実現され、ラウンド関数（Ｒ）の処理に依存することなくより大きな拡散性能が保証される。
すなわち、少ないラウンド数で攻撃に対する耐性の高い安全な暗号処理が実現される。結果として、高速処理や軽量化が実現される。
【０３５７】
また、インボリューション性に基づく効果としては、１つの鍵変換部を繰り返し利用する構成が可能であり、アンロールド実装、ラウンド実装のいずれにおいてもハードウェアの小型化が実現される。
【０３５８】
なお、図７２に示す例では、ラウンド鍵供給部７２１は、以下の順番で、鍵の出力を行う。
鍵K_1、
鍵K_2、
変換鍵Kd_1、
変換鍵Kd_2、
排他的論理和演算鍵K_1（＋）K_2、
排他的論理和演算変換鍵Kd_1（＋）Kd_2、
排他的論理和演算鍵K_1（＋）K_2、
排他的論理和演算変換鍵Kd_1（＋）Kd_2、
排他的論理和演算鍵K_1（＋）K_2、
変換鍵Kd_2、
変換鍵Kd_1、
鍵K_2、
鍵K_1、
この順番で、６種類の鍵を出力する。
【０３５９】
なお、暗号処理部７５０に対して入力されるラウンド鍵K_1～K_{13}は、上記の各鍵をそのまま、あるいは、定数ＣＯＮを作用させて生成される。
暗号処理部７５０の中心位置にある線形変換部Ｐ２の前後において、ラウンド鍵K_7として、排他的論理和演算鍵K_1（＋）K_2、を繰り返し利用する。
また、ラウンド鍵RK_8，RK_{10}，RK_{12}は、定数供給部７２５から供給される定数ＣＯＮをラウンド鍵供給部７２１の供給する鍵に排他的論理和して生成される。
【０３６０】

上記鍵の入力シーケンスは、先に図５９を参照して説明したシーケンスと同様であり、逆の順番も同一シーケンスとなる。

これは、平文Ｐから暗号文Ｃを生成する暗号処理における鍵入力順と、暗号文Ｃから平文Ｐを生成する復号処理において、ラウンド鍵供給部７２１は、同じシーケンスで鍵生成、出力を行うことが可能であることを意味する。これは、暗号処理および復号処理に適用するハードウェアやプログラムを共通化可能であることを意味し、装置の軽量化（小型化）に寄与する設定である。

【０３６１】

鍵スケジュール部７２０に設定された定数供給部７２５は、先に項目［６．定数入力による安全性向上を実現する構成について］において、図５３～図５９を参照して説明した処理に従った定数供給処理を実行する。

図に示す例において、定数（ＣＯＮ）は、

ラウンド鍵RK_8、

ラウンド鍵RK_{10}

ラウンド鍵RK_{12}、

これらのラウンド鍵生成時にラウンド鍵供給部の生成する鍵データに対して排他的論理和演算がなされる。

【０３６２】

すなわち、

ラウンド鍵$RK_8 = Kd_1$（＋）Kd_2（＋）ＣＯＮ

ラウンド鍵$RK_{10} = Kd_2$（＋）ＣＯＮ

ラウンド鍵$RK_{12} = K_2$（＋）ＣＯＮ

なお、（＋）は排他的論理和演算を意味する。

【０３６３】

このような定数（ＣＯＮ）の入力処理の結果として、暗号処理部７５０に入力されるラウンド鍵RK_1～RK_{13}の設定は以下の通りとなる。

$RK_1 = K_1$、

$RK_2 = K_2$、

$RK_3 = Kd_1$、

$RK_4 = Kd_2$、

$RK_5 = K_1$（＋）K_2、

$RK_6 = Kd_1$（＋）Kd_2、

$RK_7 = K_1$（＋）K_2、

$RK_7 = K_1$（＋）K_2、

$RK_8 = Kd_1$（＋）Kd_2（＋）ＣＯＮ、

$RK_9 = K_1$（＋）K_2、

$RK_{10} = Kd_2$（＋）ＣＯＮ、

$RK_{11} = Kd_1$、

$RK_{12} = K_2$（＋）ＣＯＮ、

$RK_{13} = K_1$、

なお、（＋）は排他的論理和演算を意味する。

RK_7は、同じラウンド鍵を線形変換部（Ｐ２）の前後で２回入力する設定となっている。

【０３６４】

このように、定数（ＣＯＮ）はラウンド鍵の生成時にラウンド鍵供給部の生成する鍵に対して排他的論理和処理がなされる。

なお、定数をラウンド鍵とは別に、暗号処理部の排他的論理和部に入力して、変換データとの排他的論理和処理を行なってもよい。この場合も結果は同じとなる。

【０３６５】

なお、定数（ＣＯＮ）は、前述したように、定数ＣＯＮと、定数ＣＯＮを入力する暗号

処理部の排他的論理和部に隣接する線形変換部において適用する線形変換行列との行列演算の結果の全要素が非ゼロ、すなわちゼロでない値となる定数（ＣＯＮ）を利用する。

この構成によって、線形変換部の線形変換による差分減少が防止される。この結果、最小差分アクティブＳボックスの数の減少を防止することが可能となり、各種の攻撃に対する耐性を高めた安全性の高い暗号処理構成が実現される。

【０３６６】

さらに、暗号処理部７５０に設定される非線形変換部は、先に項目［７．非線形変換部に適用するＳボックス（Ｓ－ｂｏｘ）の具体的構成例について］において、図６４～図６６を参照して説明したインボリューション性を有する４ビット入出力ｋとボックス（Ｓ－ｂｏｘ）を複数設定した構成である。

先に説明たように、暗号処理部７５０は、変換関数Ｅと、線形変換部、変換関数E^{-1}を有する構成であり、変換関数Ｅと、変換関数E^{-}の非線形変換部に図６４～図６６に示すＳボックスを利用した構成とすることで、暗号処理部全体のインボリューション性が実現される。

【０３６７】

［９．暗号処理装置の実装例について］

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の実装例について説明する。

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は、暗号処理を実行する様々な情報処理装置に搭載可能である。具体的には、ＰＣ、ＴＶ、レコーダ、プレーヤ、通信機器、さらに、ＲＦＩＤ、スマートカード、センサネットワーク機器、デンチ／バッテリー認証モジュール、健康・医療機器、自立型ネットワーク機器等、例えばデータ処理や通信処理に伴う暗号処理を実行する様々な機器において利用可能である。

【０３６８】

本開示の暗号処理を実行する装置の一例としてのＩＣモジュール８００の構成例を図７３に示す。上述の処理は、例えばＰＣ、ＩＣカード、リーダライタ、スマートフォンやウェアラブルデバイス等の様々な情報処理装置において実行可能であり、図７３に示すＩＣモジュール８００は、これら様々な機器に構成することが可能である。

【０３６９】

図７３に示すＣＰＵ（Ｃｅｎｔｒａｌ　ｐｒｏｃｅｓｓｉｎｇ　Ｕｎｉｔ）８０１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８０２は、ＣＰＵ８０１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）、ＣＰＵ８０１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８０２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。

【０３７０】

暗号処理部８０３は、上記において説明した暗号処理構成を有しい、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。

【０３７１】

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、ＣＰＵ８０１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。

【０３７２】

乱数発生器８０４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。

【０３７３】

送受信部８０５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。

【０３７４】

なお、上述した実施例において説明した暗号処理装置は、入力データとしての平文を暗号化する暗号化処理に適用可能であるのみならず、入力データとしての暗号文を平文に復元する復号処理にも適用可能である。

暗号化処理、復号処理、双方の処理において、上述した実施例において説明した構成を適用することが可能である。

【０３７５】

図７４は、本開示に係る暗号処理を実行するスマートフォン９００の概略的な構成の一例を示すブロック図である。スマートフォン９００は、プロセッサ９０１、メモリ９０２、ストレージ９０３、外部接続インタフェース９０４、カメラ９０６、センサ９０７、マイクロフォン９０８、入力デバイス９０９、表示デバイス９１０、スピーカ９１１、無線通信インタフェース９１３、アンテナスイッチ９１４、アンテナ９１５、バス９１７、バッテリー９１８及び補助コントローラ９１９を備える。

【０３７６】

プロセッサ９０１は、例えばＣＰＵ（Ｃｅｎｔｒａｌ　Ｐｒｏｃｅｓｓｉｎｇ　Ｕｎｉｔ）又はＳｏＣ（Ｓｙｓｔｅｍ　ｏｎ　Ｃｈｉｐ）であってよく、スマートフォン９００のアプリケーションレイヤ及びその他のレイヤの機能を制御し、また、暗号処理を制御する。メモリ９０２は、ＲＡＭ（Ｒａｎｄｏｍ　Ａｃｃｅｓｓ　Ｍｅｍｏｒｙ）及びＲＯＭ（Ｒｅａｄ　Ｏｎｌｙ　Ｍｅｍｏｒｙ）を含み、プロセッサ９０１により実行されるプログラム及びデータを記憶する。また、メモリ９０２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。ストレージ９０３は、半導体メモリ又はハードディスクなどの記憶媒体を含み得る。外部接続インタフェース９０４は、メモリーカード又はＵＳＢ（Ｕｎｉｖｅｒｓａｌ　Ｓｅｒｉａｌ　Ｂｕｓ）デバイスなどの外付けデバイスをスマートフォン９００へ接続するためのインタフェースである。

【０３７７】

カメラ９０６は、例えば、ＣＣＤ（Ｃｈａｒｇｅ　Ｃｏｕｐｌｅｄ　Ｄｅｖｉｃｅ）又はＣＭＯＳ（Ｃｏｍｐｌｅｍｅｎｔａｒｙ　Ｍｅｔａｌ　Ｏｘｉｄｅ　Ｓｅｍｉｃｏｎｄｕｃｔｏｒ）などの撮像素子を有し、撮像画像を生成する。センサ９０７は、例えば、測位センサ、ジャイロセンサ、地磁気センサ及び加速度センサなどのセンサ群を含み得る。マイクロフォン９０８は、スマートフォン９００へ入力される音声を音声信号へ変換する。カメラ９０６で生成された画像や、センサ９０７で取得されたセンサデータ、マイクロフォン９０８で取得した音声信号などは、プロセッサ９０１により暗号化され無線通信インタフェース９１３を介して他の装置に送信されてもよい。入力デバイス９０９は、例えば、表示デバイス９１０の画面上へのタッチを検出するタッチセンサ、キーパッド、キーボード、ボタン又はスイッチなどを含み、ユーザからの操作又は情報入力を受け付ける。表示デバイス９１０は、液晶ディスプレイ（ＬＣＤ）又は有機発光ダイオード（ＯＬＥＤ）ディスプレイなどの画面を有し、スマートフォン９００の出力画像を表示する。スピーカ９１１は、スマートフォン９００から出力される音声信号を音声に変換する。

【０３７８】

無線通信インタフェース９１３は、無線通信を実行し、典型的には、ベースバンドプロセッサ、ＲＦ（Ｒａｄｉｏ　Ｆｒｅｑｕｅｎｃｙ）回路及びパワーアンプなどを含み得る。無線通信インタフェース９１３は、通信制御プログラムを記憶するメモリ、当該プログラムを実行するプロセッサ及び関連する回路を集積したワンチップのモジュールであってもよい。無線通信インタフェース９１３は、無線ＬＡＮ方式に加えて、近距離無線通信方式、近接無線通信方式又はセルラ通信方式などの他の種類の無線通信方式をサポートして

もよい。

【０３７９】

バス９１７は、プロセッサ９０１、メモリ９０２、ストレージ９０３、外部接続インタフェース９０４、カメラ９０６、センサ９０７、マイクロフォン９０８、入力デバイス９０９、表示デバイス９１０、スピーカ９１１、無線通信インタフェース９１３及び補助コントローラ９１９を互いに接続する。バッテリー９１８は、図中に破線で部分的に示した給電ラインを介して、図７４に示したスマートフォン９００の各ブロックへ電力を供給する。補助コントローラ９１９は、例えば、スリープモードにおいて、スマートフォン９００の必要最低限の機能を動作させる。

【０３８０】

なお、上述した実施例において説明したスマートフォンにおける暗号処理は、入力データとしての平文を暗号化する暗号化処理に適用可能であるのみならず、入力データとしての暗号文を平文に復元する復号処理にも適用可能である。

暗号化処理、復号処理、双方の処理において、上述した実施例において説明した構成を適用することが可能である。

また、図７４に示すスマートフォン９００に図７３に示すＩＣモジュール８００を搭載し、上述した実施例に従った暗号処理をＩＣモジュール８００において実行する構成としてもよい。

【０３８１】

［１０．本開示の構成のまとめ］

以上、特定の実施例を参照しながら、本開示の実施例について詳解してきた。しかしながら、本開示の要旨を逸脱しない範囲で当業者が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。本開示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。

【０３８２】

なお、本明細書において開示した技術は、以下のような構成をとることができる。

（１）　入力データに対するラウンド演算を実行して出力データを生成する暗号処理部と、

前記暗号処理部におけるラウンド演算において適用するラウンド鍵を前記暗号処理部に出力する鍵スケジュール部を有し、

前記暗号処理部は、

データ変換関数Ｅと、前記データ変換関数Ｅの逆関数E^{-1}をシーケンシャルに実行するインボリューション性を有し、

前記関数Ｅ、または、前記逆関数E^{-1}のいずれか一方において１回以上の定数を適用したラウンド演算を実行する暗号処理装置。

【０３８３】

（２）前記暗号処理部は、前記定数を適用したラウンド演算として、前記定数と変換対象データ、または前記定数とラウンド鍵との排他的論理和演算を実行する前記（１）に記載の暗号処理装置。

【０３８４】

（３）　前記ラウンド演算を実行するラウンド演算部は、線形変換処理を実行する線形変換処理部を有し、前記暗号処理部は、前記定数を、前記線形変換処理部と繋がる排他的論理和部に入力して、変換対象データ、またはラウンド鍵との排他的論理和演算を行う前記（１）または（２）に記載の暗号処理装置。

【０３８５】

（４）前記入力データ、および前記定数は、各要素が１ビット以上のｍ×ｎ個の要素からなるステートであり、前記線形変換部は、前記ステートに対して線形変換行列を適用した行列演算を実行する構成であり、前記定数は、定数を入力する排他的論理和部と繋がる線形変換処理部が線形変換処理に適用する線形変換行列と、該定数との行列演算結果であ

るステートの構成要素が全て非ゼロとなる条件を満たすステートである前記（３）に記載の暗号処理装置。

【０３８６】

（５）前記入力データ、および前記定数は、各要素が４ビットの４×４個の要素からなるステートであり、前記線形変換部は、前記ステートに対して線形変換行列を適用した行列演算を実行する構成であり、前記定数は、定数を入力する排他的論理和部の隣接位置の線形変換処理部が線形変換処理に適用する線形変換行列と、該定数との行列演算結果であるステートの構成要素が全て非ゼロとなる条件を満たす４×４ステートである前記（３）または（４）に記載の暗号処理装置。

【０３８７】

（６）前記線形変換部は、前記ステートの各列要素単位で行列を適用して線形変換を行う列拡散演算、または、前記ステートの各行要素単位で行列を適用して線形変換を行う行拡散演算のいずれかの行列演算を実行する前記（３）～（５）いずれかに記載の暗号処理装置。

【０３８８】

（７） 前記ラウンド演算を実行するラウンド演算部は、線形変換処理を実行する線形変換処理部を有し、前記暗号処理部は、１つおきのラウンドの線形変換処理部に繋がる排他的論理和部に前記定数を入力して、変換対象データ、またはラウンド鍵との排他的論理和を行う前記（１）～（６）いずれかに記載の暗号処理装置。

【０３８９】

（８）前記暗号処理部は、平文Ｐを入力データとして前記ラウンド演算を繰り返して出力データとしての暗号文Ｃを出力し、前記暗号文Ｃを入力データとして、前記ラウンド演算の実行シーケンスを逆順に設定したデータ変換処理により出力データとして前記平文Ｐを生成可能なインボリューション性を有する構成である前記（１）～（７）いずれかに記載の暗号処理装置。

【０３９０】

（９）前記鍵スケジュール部は、平文Ｐから暗号文Ｃを生成する場合の鍵供給シーケンスと、暗号文Ｃから平文Ｐを生成する場合の鍵供給シーケンスが一致するインボリューション性を有する鍵供給処理を行なう構成である前記（１）～（８）いずれかに記載の暗号処理装置。

【０３９１】

（１０）前記鍵スケジュール部は、前記暗号処理部に対する鍵供給処理に際して、供給鍵の一部に定数による演算を施し、演算結果である鍵データを前記暗号処理部に出力する前記（１）～（９）いずれかに記載の暗号処理装置。

【０３９２】

（１１）前記暗号処理部が繰り返し実行するラウンド演算は、線形変換部による線形変換処理を含む演算であり、前記線形変換部は、ラウンド遷移に応じて線形変換態様を変更する前記（１）～（１０）いずれかに記載の暗号処理装置。

【０３９３】

（１２）前記ラウンド演算は非線形変換処理を含み、前記非線形変換処理を実行するＳボックスは、入力値から得られる出力値を、再入力することで前記入力値が得られるインボリューション性を有する構成である前記（１）～（１１）いずれかに記載の暗号処理装置。

【０３９４】

（１３） 入力データに対するラウンド演算を実行して出力データを生成する暗号処理部と、

前記暗号処理部におけるラウンド演算において適用するラウンド鍵を前記暗号処理部に出力する鍵スケジュール部を有し、

前記暗号処理部は、

データ変換関数Ｅと、前記データ変換関数Ｅの逆関数E^{-1}をシーケンシャルに実行す

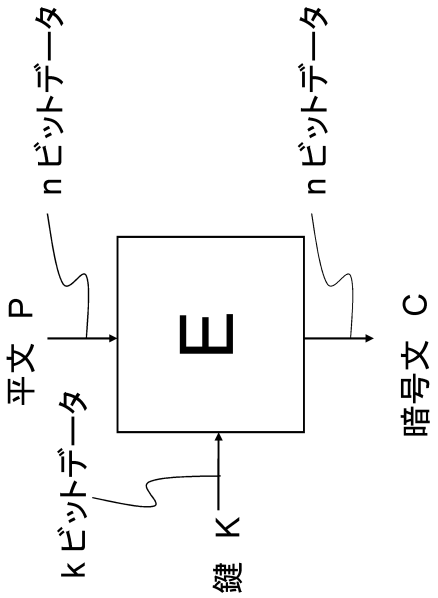
るインボリューション性を有し、
前記関数Ｅ、および前記逆関数Ｅ$^{-1}$の双方において１回以上の定数を適用したラウンド演算を実行する構成を有し、前記関数Ｅ、および前記逆関数Ｅ$^{-1}$の非対応位置に定数適用位置が設定されている暗号処理装置。
【０３９５】
（１４）前記暗号処理部は、前記定数を適用したラウンド演算として、前記定数と変換対象データ、または前記定数とラウンド鍵との排他的論理和演算を実行する前記（１３）に記載の暗号処理装置。
【０３９６】
（１５）前記ラウンド演算を実行するラウンド演算部は、線形変換処理を実行する線形変換処理部を有し、前記暗号処理部は、前記定数を、前記線形変換処理部に繋がる排他的論理和部に入力して、変換対象データ、またはラウンド鍵との排他的論理和を行う前記（１３）または（１４）に記載の暗号処理装置。
【０３９７】
（１６）前記入力データ、および前記定数は、各要素が１ビット以上のｍ×ｎ個の要素からなるステートであり、前記線形変換部は、前記ステートに対して線形変換行列を適用した行列演算を実行する構成であり、前記定数は、定数を入力する排他的論理和部に繋がる線形変換処理部が線形変換処理に適用する線形変換行列と、該定数との行列演算結果であるステートの構成要素が全て非ゼロとなる条件を満たすステートである前記（１５）に記載の暗号処理装置。
【０３９８】
（１７）　暗号処理装置において実行する暗号処理方法であり、
前記暗号処理装置は、
入力データに対するラウンド演算を実行して出力データを生成する暗号処理部と、
前記暗号処理部におけるラウンド演算において適用するラウンド鍵を前記暗号処理部に出力する鍵スケジュール部を有し、
前記暗号処理部は、
データ変換関数Ｅと、前記データ変換関数Ｅの逆関数Ｅ$^{-1}$をシーケンシャルに実行するインボリューション性を有し、
前記関数Ｅ、または、前記逆関数Ｅ$^{-1}$のいずれか一方において、１回以上の定数を適用したラウンド演算を実行する暗号処理方法。
【０３９９】
（１８）　暗号処理装置において実行する暗号処理方法であり、
前記暗号処理装置は、
入力データに対するラウンド演算を実行して出力データを生成する暗号処理部と、
前記暗号処理部におけるラウンド演算において適用するラウンド鍵を前記暗号処理部に出力する鍵スケジュール部を有し、
前記暗号処理部は、
データ変換関数Ｅと、前記データ変換関数Ｅの逆関数Ｅ$^{-1}$をシーケンシャルに実行するインボリューション性を有し、前記関数Ｅ、および前記逆関数Ｅ$^{-1}$の非対応位置に定数適用位置が設定された構成であり、
前記関数Ｅ、および前記逆関数Ｅ$^{-1}$の双方において１回以上の定数を適用したラウンド演算を実行する暗号処理方法。
【０４００】
（１９）　暗号処理装置において暗号処理を実行させるプログラムであり、
前記暗号処理装置は、
入力データに対するラウンド演算を実行して出力データを生成する暗号処理部と、
前記暗号処理部におけるラウンド演算において適用するラウンド鍵を前記暗号処理部に出力する鍵スケジュール部を有し、
前記暗号処理部は、

データ変換関数Ｅと、前記データ変換関数Ｅの逆関数Ｅ$^{-1}$をシーケンシャルに実行するインボリューション性を有し、
前記プログラムは、前記暗号処理部に、前記関数Ｅ、または、前記逆関数Ｅ$^{-1}$のいずれか一方において、１回以上の定数を適用したラウンド演算を実行させるプログラム。
【０４０１】
（２０）　暗号処理装置において暗号処理を実行させるプログラムであり、
前記暗号処理装置は、
入力データに対するラウンド演算を実行して出力データを生成する暗号処理部と、
前記暗号処理部におけるラウンド演算において適用するラウンド鍵を前記暗号処理部に出力する鍵スケジュール部を有し、
前記暗号処理部は、
データ変換関数Ｅと、前記データ変換関数Ｅの逆関数Ｅ$^{-1}$をシーケンシャルに実行するインボリューション性を有し、前記関数Ｅ、および前記逆関数Ｅ$^{-1}$の非対応位置に定数適用位置が設定された構成であり、
前記プログラムは、前記暗号処理部に前記関数Ｅ、および前記逆関数Ｅ$^{-1}$の双方において１回以上の定数を適用したラウンド演算を実行させるプログラム。
【０４０２】
また、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。例えば、プログラムは記録媒体に予め記録しておくことができる。記録媒体からコンピュータにインストールする他、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。
【０４０３】
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】
【０４０４】
上述したように、本開示の一実施例の構成によれば、各種の攻撃に対する耐性の高い安全性の優れた暗号処理が実現される。
具体的には、入力データに対するラウンド演算を繰り返して出力データを生成する暗号処理部と、暗号処理部におけるラウンド演算において適用するラウンド鍵を暗号処理部に出力する鍵スケジュール部を有し、暗号処理部は、データ変換関数Ｅと、前記データ変換関数Ｅの逆関数Ｅ$^{-1}$をシーケンシャルに実行するインボリューション性を有し、関数Ｅ、または逆関数Ｅ$^{-1}$のいずれか一方のみにおいて、１回以上の定数を適用したラウンド演算を実行する。定数は、定数を入力する排他的論理和部の隣接位置の線形変換処理部において適用する線形変換行列との行列演算結果であるステートの構成要素が全て非ゼロとなる条件を満たすステートとして構成される。
本構成により各種の攻撃に対する耐性を向上させた安全性の高い暗号処理構成が実現される。
【符号の説明】
【０４０５】
１００　暗号処理装置
１１０　鍵スケジュール部
１２０　暗号処理部

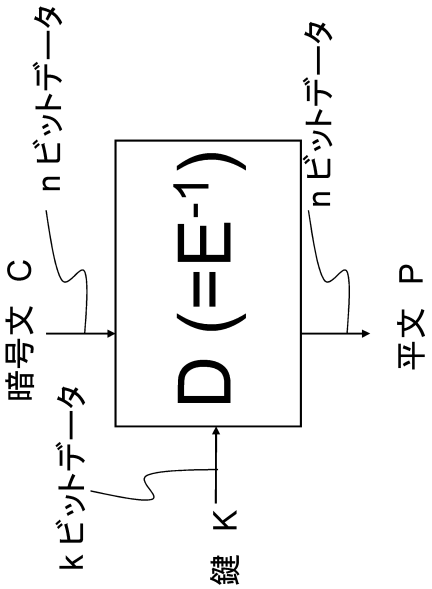
１２１　排他的論理和部
１２２　非線形変換部
１２３　線形変換部
２０１　線形変換部Ｐ１
２０２　線形変換部Ｐ２
２０３　線形変換部Ｐ３
３００　鍵スケジュール部
３０１　鍵供給部（鍵レジスタ）
３０２　鍵変換部
３２０　暗号処理部
３２１～３２７　排他的論理和部
３３１，３３３　排他的論理和部
３３２　非線形／線形変換部（Ｓ＆Ｐ）
３５０　暗号処理部
３５１　排他的論理和部
３５２　非線形／線形変換部（Ｓ＆Ｐ）
３６０　鍵スケジュール部
３６１，３６２　鍵レジスタ
３６３　鍵変換部
３７１　鍵レジスタ
３７２　鍵変換部
３８１　鍵レジスタ
３９１　鍵レジスタ
３９２，３９４　排他的論理和部
３９３　鍵変換部
４０１　ラウンド演算実行部
４０２　定数入力部
４１１　変換関数Ｅ
４１２　線形変換部
４１３　変換関数E^{-1}
４３１　変換関数Ｅ
４３２　線形変換部
４３３　変換関数E^{-1}
４３５　定数入力部
４３６　排他的論理和部
４３７　線形変換部
４５１　変換関数Ｅ
４５２　線形変換部
４５３　変換関数E^{-1}
５２１　非線形変換層１
５２２　線形変換層
５２３　非線形変換層
７００　暗号処理装置
７２０　鍵スケジュール部
７２１　ラウンド鍵供給部
７２２　鍵レジスタ
７２３　鍵変換部
７２５　定数供給部
７５０　暗号処理部
７５１　排他的論理和部

７５２　非線形変換部
７５３　線形変換部
８００　ＩＣモジュール
８０１　ＣＰＵ（Ｃｅｎｔｒａｌ　ｐｒｏｃｅｓｓｉｎｇ　Ｕｎｉｔ）
８０２　メモリ
８０３　暗号処理部
８０４　乱数生成部
８０５　送受信部
９００　スマートフォン
９０１　プロセッサ
９０２　メモリ
９０３　ストレージ
９０４　外部接続インタフェース
９０６　カメラ
９０７　センサ
９０８　マイクロフォン
９０９　入力デバイス
９１０　表示デバイス
９１１　スピーカ
９１３　無線通信インタフェース
９１４　アンテナスイッチ
９１５　アンテナ
９１７　バス
９１８　バッテリー
９１９　補助コントローラ

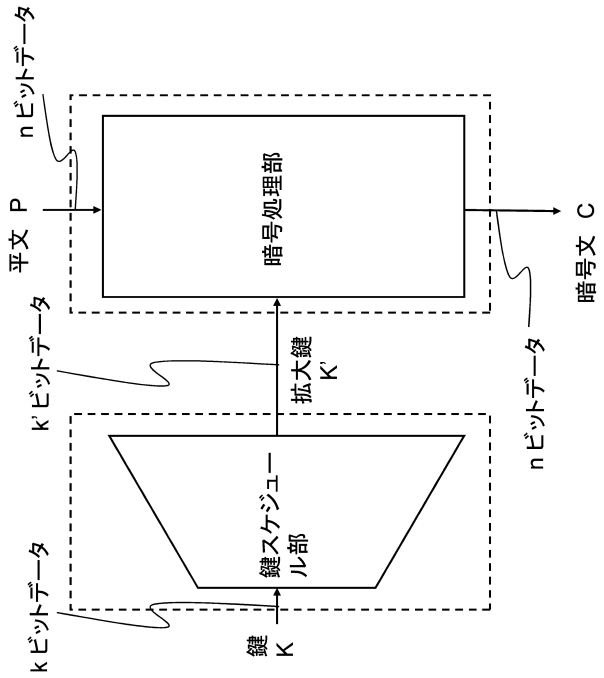
【図１】



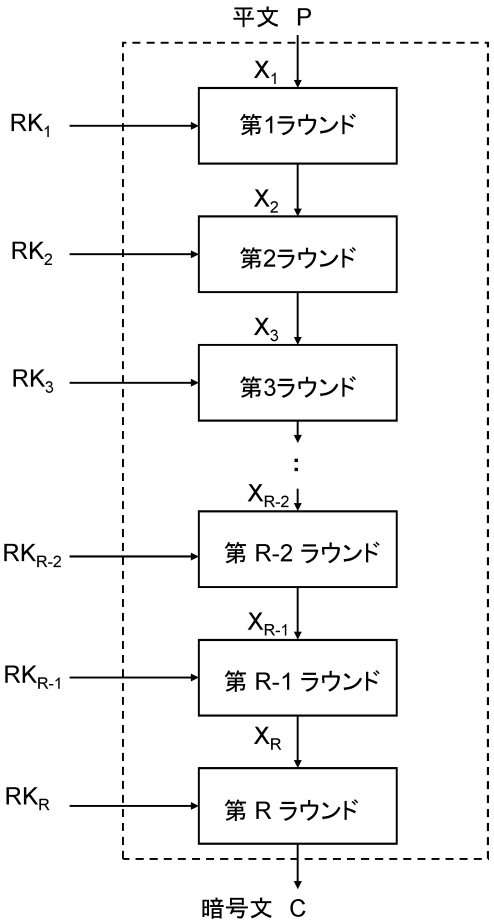
【図２】



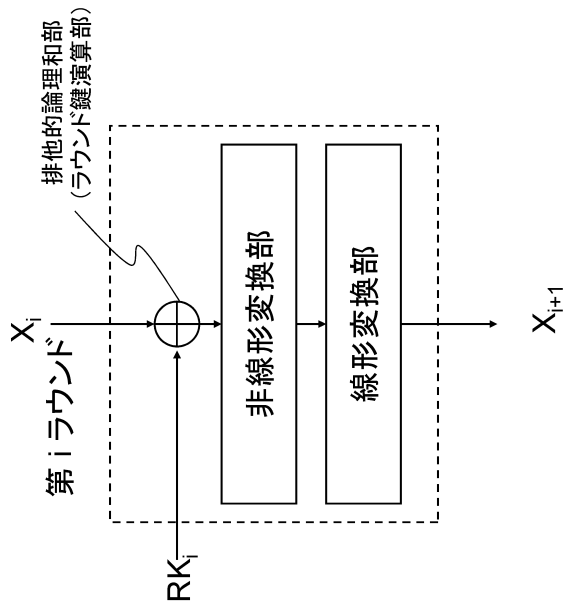
【図３】


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

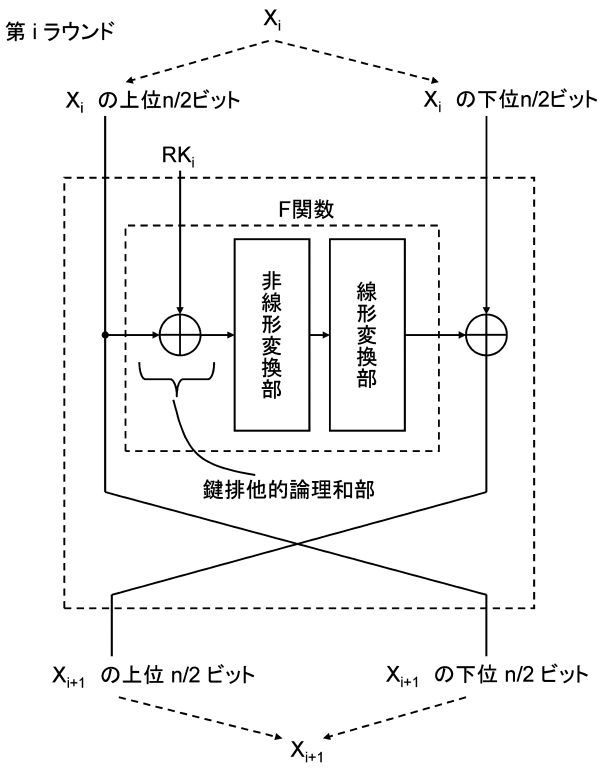
【図４】


平文 P
X1
RK1
第1ラウンド
X2
RK2
第2ラウンド
X3
RK3
第3ラウンド
XR-2
RKR-2
第 R-2 ラウンド
XR-1
RKR-1
第 R-1 ラウンド
XR
RKR
第 R ラウンド
暗号文 C

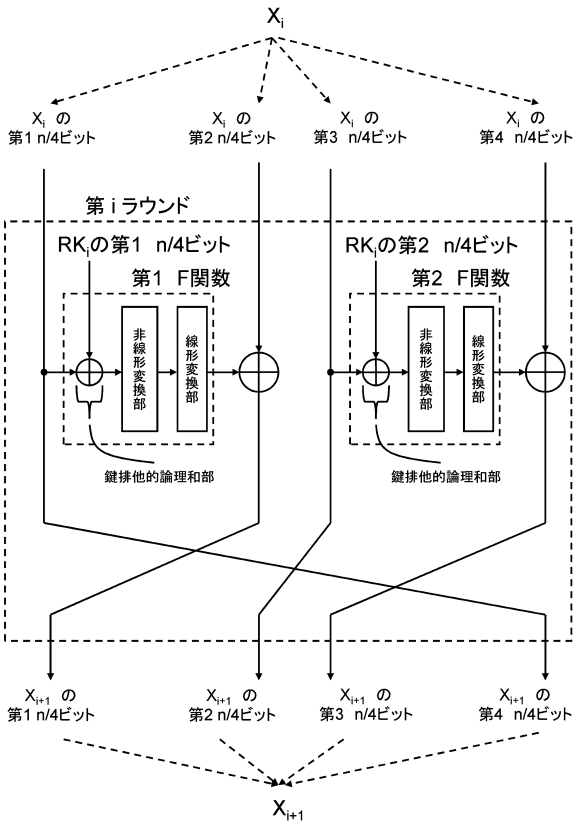
【図５】


排他的論理和部
（ラウンド鍵演算部）
Xi
第 i ラウンド
非線形変換部
線形変換部
Xi+1
RKi

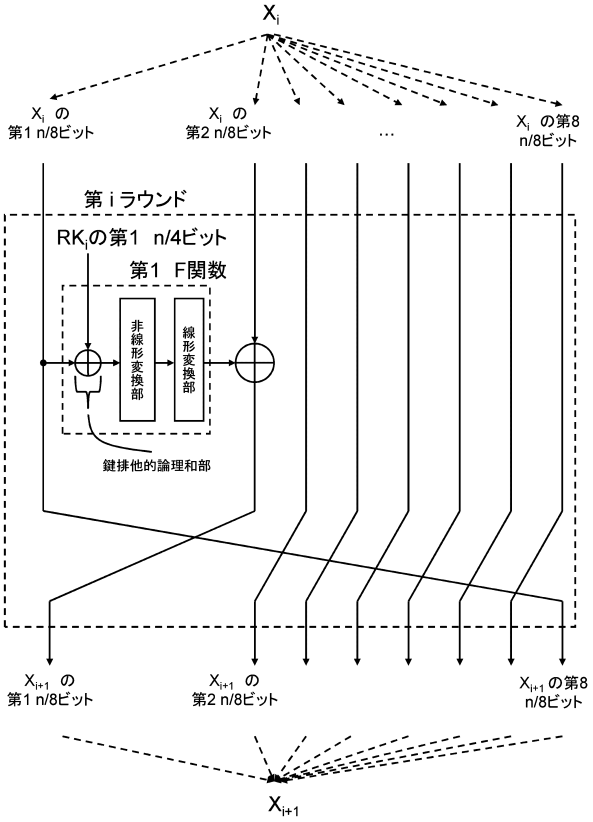
【図６】


第 i ラウンド
Xi
Xi の上位n/2ビット
Xi の下位n/2ビット
RKi
F関数
非線形変換部
線形変換部
鍵排他的論理和部
Xi+1 の上位 n/2 ビット
Xi+1 の下位 n/2 ビット
Xi+1

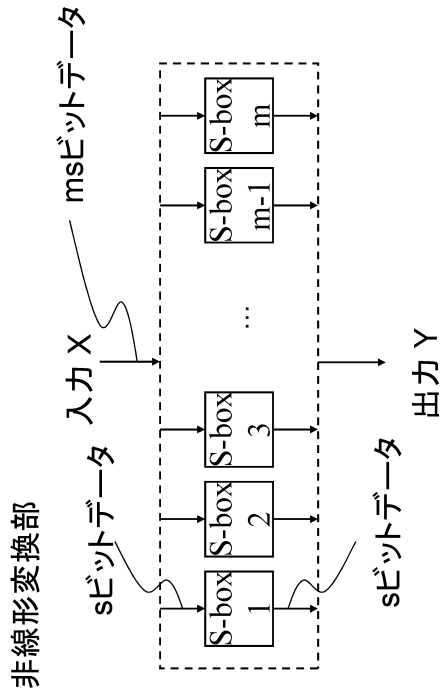
【図７】


Xi
Xi の 第1 n/4ビット
Xi の 第2 n/4ビット
Xi の 第3 n/4ビット
Xi の 第4 n/4ビット
第 i ラウンド
RKiの第1 n/4ビット
第1 F関数
RKiの第2 n/4ビット
第2 F関数
非線形変換部
線形変換部
鍵排他的論理和部
鍵排他的論理和部
Xi+1 の 第1 n/4ビット
Xi+1 の 第2 n/4ビット
Xi+1 の 第3 n/4ビット
Xi+1 の 第4 n/4ビット
Xi+1

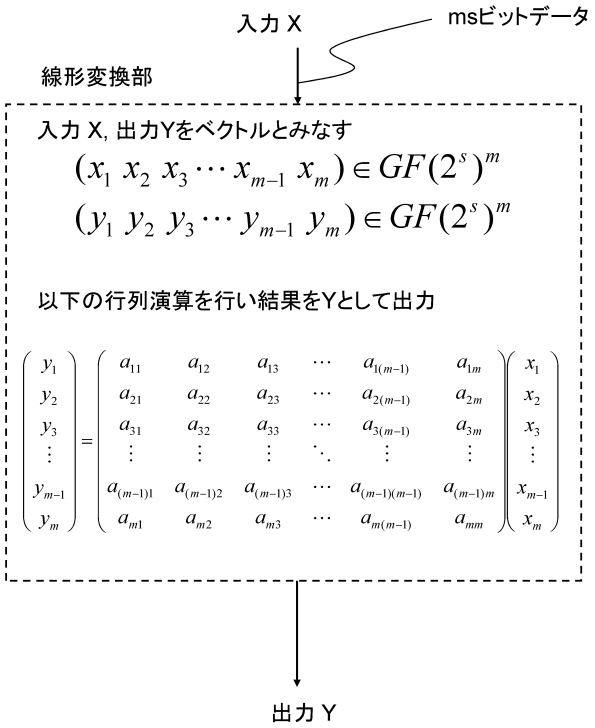
【図８】


Xi
Xi の 第1 n/8ビット
Xi の 第2 n/8ビット
…
Xi の第8 n/8ビット
第 i ラウンド
RKiの第1 n/4ビット
第1 F関数
非線形変換部
線形変換部
鍵排他的論理和部
Xi+1 の 第1 n/8ビット
Xi+1 の 第2 n/8ビット
Xi+1 の第8 n/8ビット
Xi+1

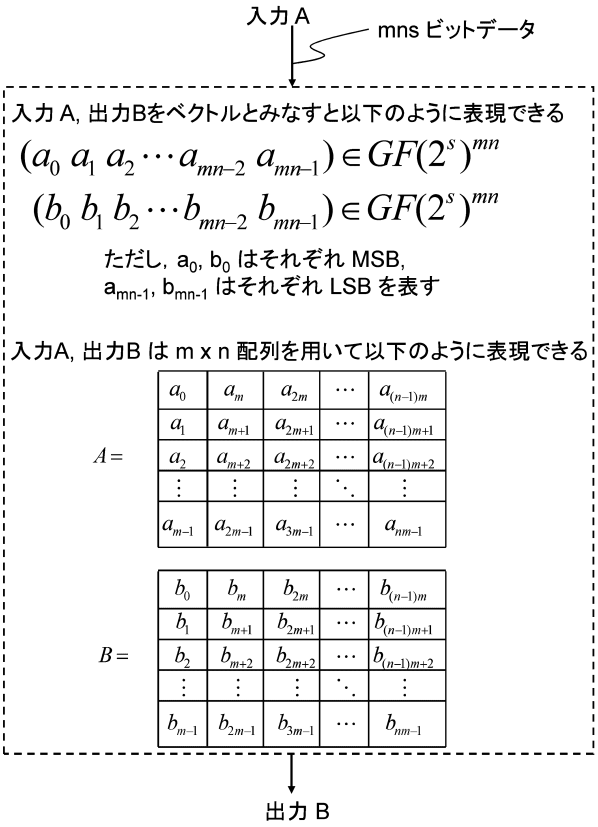
【図９】


msビットデータ
入力 X
非線形変換部
sビットデータ
S-box m
S-box m-1
S-box 3
S-box 2
S-box 1
出力 Y
sビットデータ

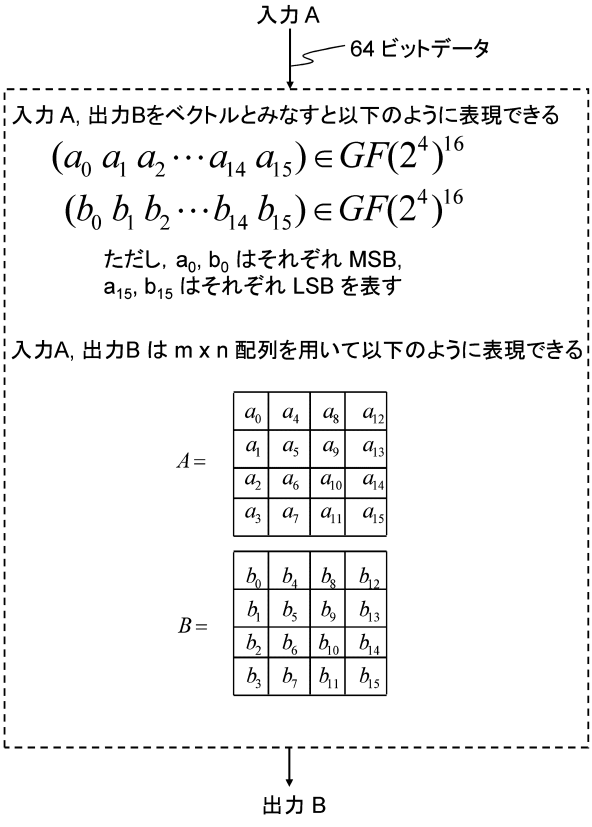
【図１０】


入力 X
msビットデータ
線形変換部
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
$\begin{pmatrix} y_1 \\ y_2 \\ y_3 \\ \vdots \\ y_{m-1} \\ y_m \end{pmatrix} = \begin{pmatrix} a_{11} & a_{12} & a_{13} & \cdots & a_{1(m-1)} & a_{1m} \\ a_{21} & a_{22} & a_{23} & \cdots & a_{2(m-1)} & a_{2m} \\ a_{31} & a_{32} & a_{33} & \cdots & a_{3(m-1)} & a_{3m} \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ a_{(m-1)1} & a_{(m-1)2} & a_{(m-1)3} & \cdots & a_{(m-1)(m-1)} & a_{(m-1)m} \\ a_{m1} & a_{m2} & a_{m3} & \cdots & a_{m(m-1)} & a_{mm} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_{m-1} \\ x_m \end{pmatrix}$
出力 Y

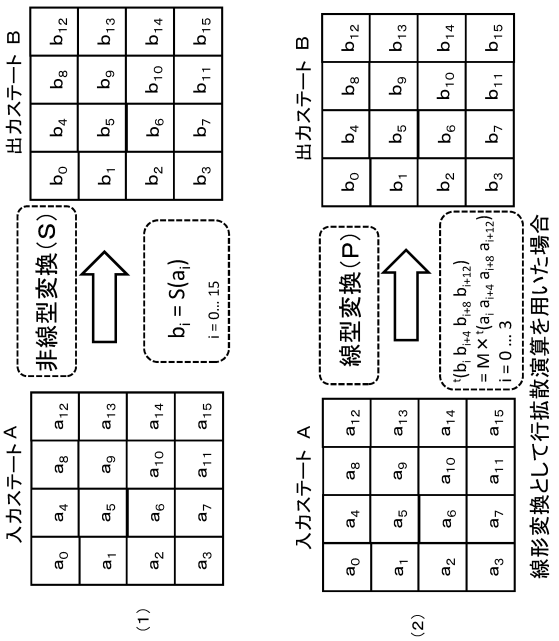
【図１１】



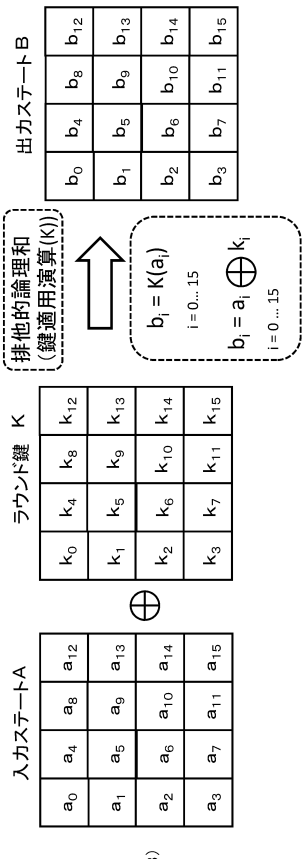
【図１２】



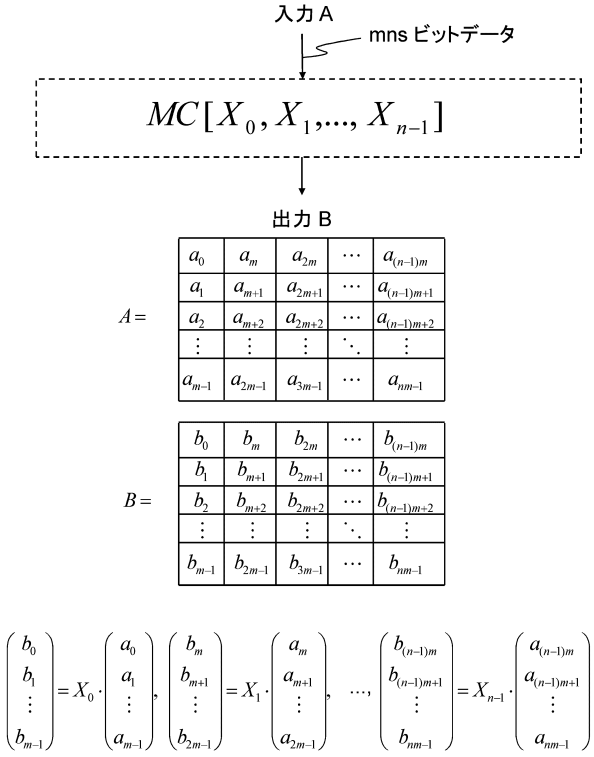
【図１３】



【図１４】

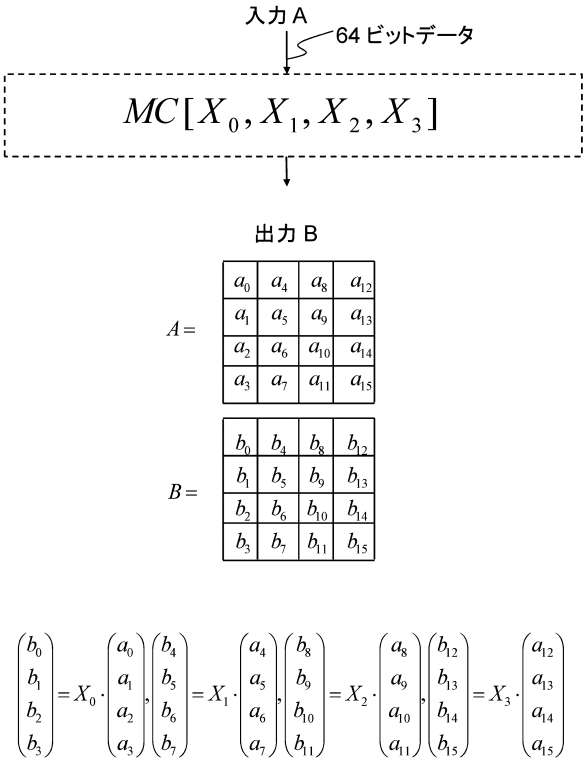


【図１５】



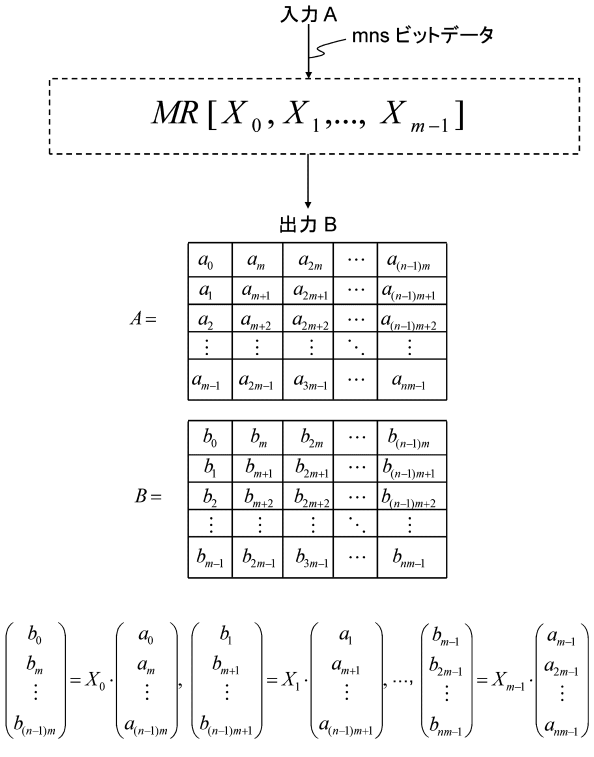
$$\begin{pmatrix} b_0 \\ b_1 \\ \vdots \\ b_{m-1} \end{pmatrix} = X_0 \cdot \begin{pmatrix} a_0 \\ a_1 \\ \vdots \\ a_{m-1} \end{pmatrix}, \begin{pmatrix} b_m \\ b_{m+1} \\ \vdots \\ b_{2m-1} \end{pmatrix} = X_1 \cdot \begin{pmatrix} a_m \\ a_{m+1} \\ \vdots \\ a_{2m-1} \end{pmatrix}, \quad \cdots, \begin{pmatrix} b_{(n-1)m} \\ b_{(n-1)m+1} \\ \vdots \\ b_{nm-1} \end{pmatrix} = X_{n-1} \cdot \begin{pmatrix} a_{(n-1)m} \\ a_{(n-1)m+1} \\ \vdots \\ a_{nm-1} \end{pmatrix}$$

【図１６】



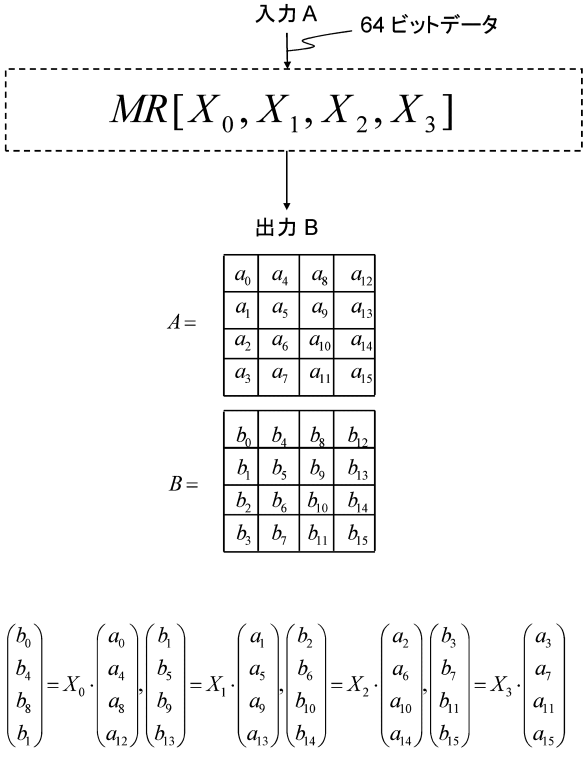
$$\begin{pmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \end{pmatrix} = X_0 \cdot \begin{pmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \end{pmatrix}, \begin{pmatrix} b_4 \\ b_5 \\ b_6 \\ b_7 \end{pmatrix} = X_1 \cdot \begin{pmatrix} a_4 \\ a_5 \\ a_6 \\ a_7 \end{pmatrix}, \begin{pmatrix} b_8 \\ b_9 \\ b_{10} \\ b_{11} \end{pmatrix} = X_2 \cdot \begin{pmatrix} a_8 \\ a_9 \\ a_{10} \\ a_{11} \end{pmatrix}, \begin{pmatrix} b_{12} \\ b_{13} \\ b_{14} \\ b_{15} \end{pmatrix} = X_3 \cdot \begin{pmatrix} a_{12} \\ a_{13} \\ a_{14} \\ a_{15} \end{pmatrix}$$

【図１７】



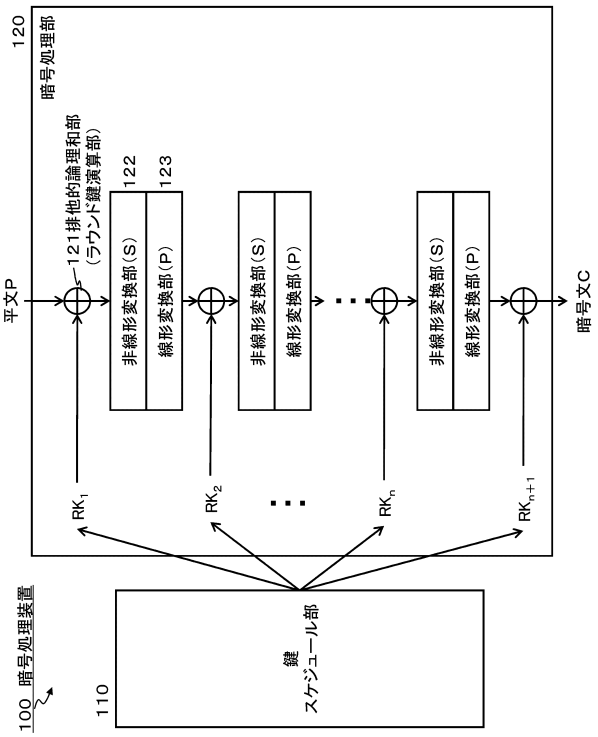
$$\begin{pmatrix} b_0 \\ b_m \\ \vdots \\ b_{(n-1)m} \end{pmatrix} = X_0 \cdot \begin{pmatrix} a_0 \\ a_m \\ \vdots \\ a_{(n-1)m} \end{pmatrix}, \begin{pmatrix} b_1 \\ b_{m+1} \\ \vdots \\ b_{(n-1)m+1} \end{pmatrix} = X_1 \cdot \begin{pmatrix} a_1 \\ a_{m+1} \\ \vdots \\ a_{(n-1)m+1} \end{pmatrix}, \cdots, \begin{pmatrix} b_{m-1} \\ b_{2m-1} \\ \vdots \\ b_{nm-1} \end{pmatrix} = X_{m-1} \cdot \begin{pmatrix} a_{m-1} \\ a_{2m-1} \\ \vdots \\ a_{nm-1} \end{pmatrix}$$

【図１８】

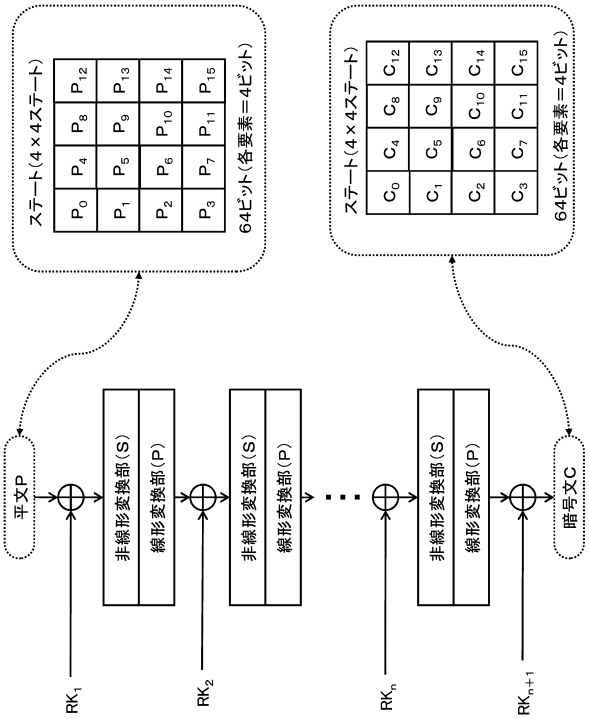


$$\begin{pmatrix} b_0 \\ b_4 \\ b_8 \\ b_1 \end{pmatrix} = X_0 \cdot \begin{pmatrix} a_0 \\ a_4 \\ a_8 \\ a_{12} \end{pmatrix}, \begin{pmatrix} b_1 \\ b_5 \\ b_9 \\ b_{13} \end{pmatrix} = X_1 \cdot \begin{pmatrix} a_1 \\ a_5 \\ a_9 \\ a_{13} \end{pmatrix}, \begin{pmatrix} b_2 \\ b_6 \\ b_{10} \\ b_{14} \end{pmatrix} = X_2 \cdot \begin{pmatrix} a_2 \\ a_6 \\ a_{10} \\ a_{14} \end{pmatrix}, \begin{pmatrix} b_3 \\ b_7 \\ b_{11} \\ b_{15} \end{pmatrix} = X_3 \cdot \begin{pmatrix} a_3 \\ a_7 \\ a_{11} \\ a_{15} \end{pmatrix}$$

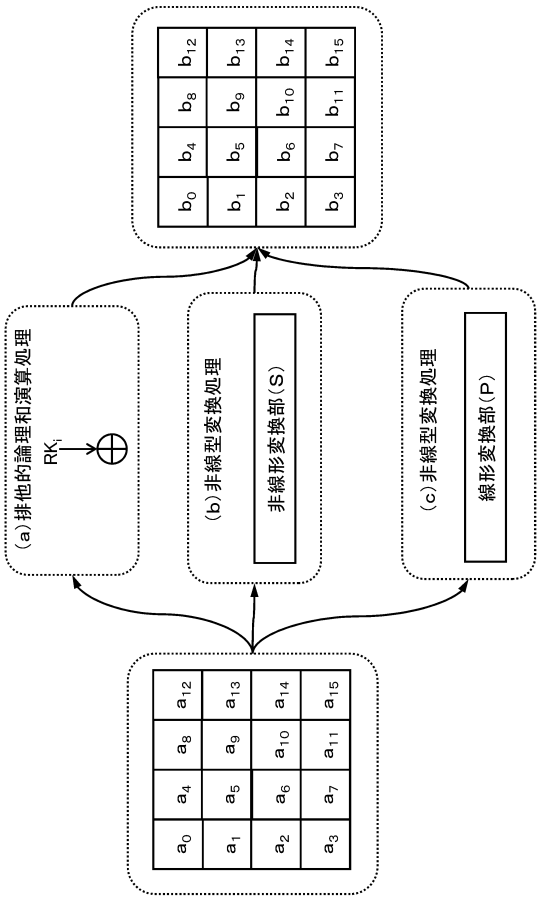
【図１９】



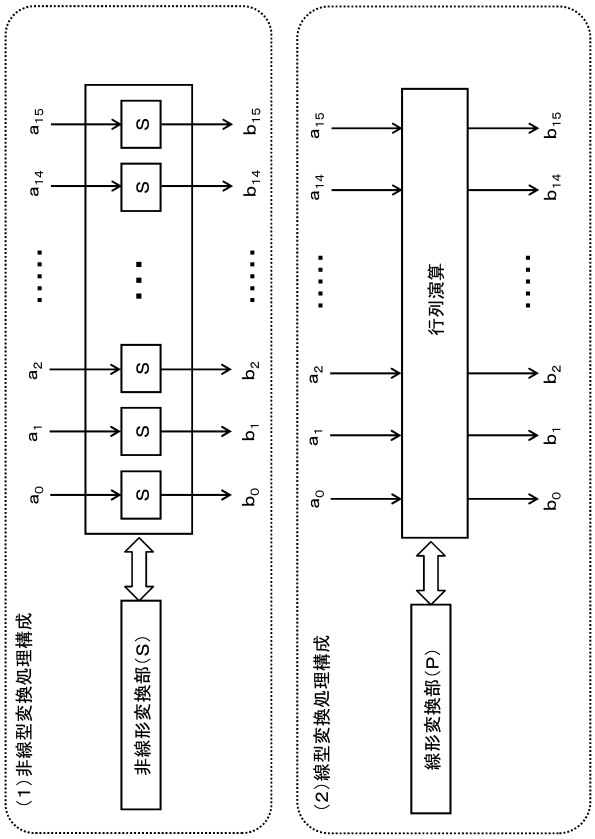
【図２０】



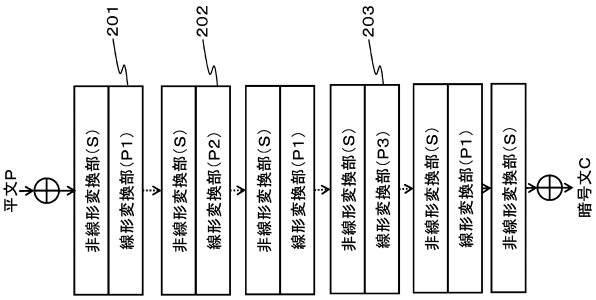
【図２１】



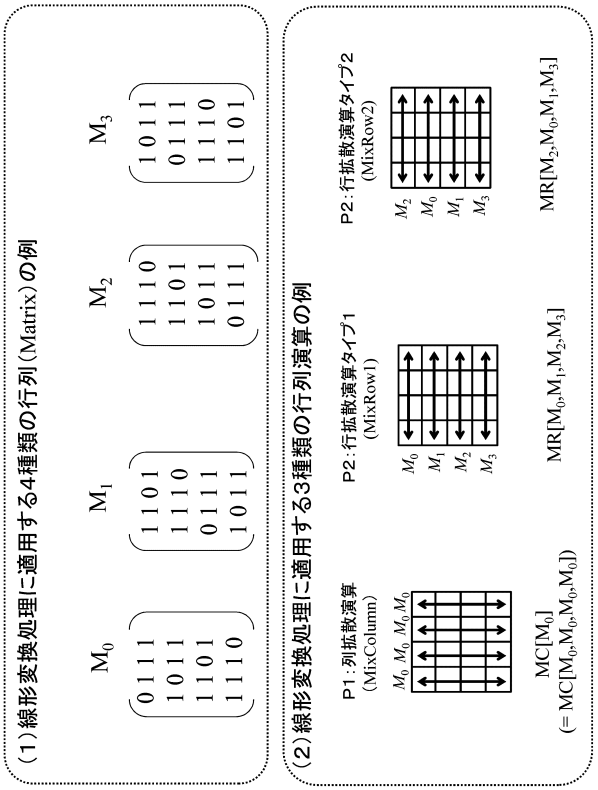
【図２２】



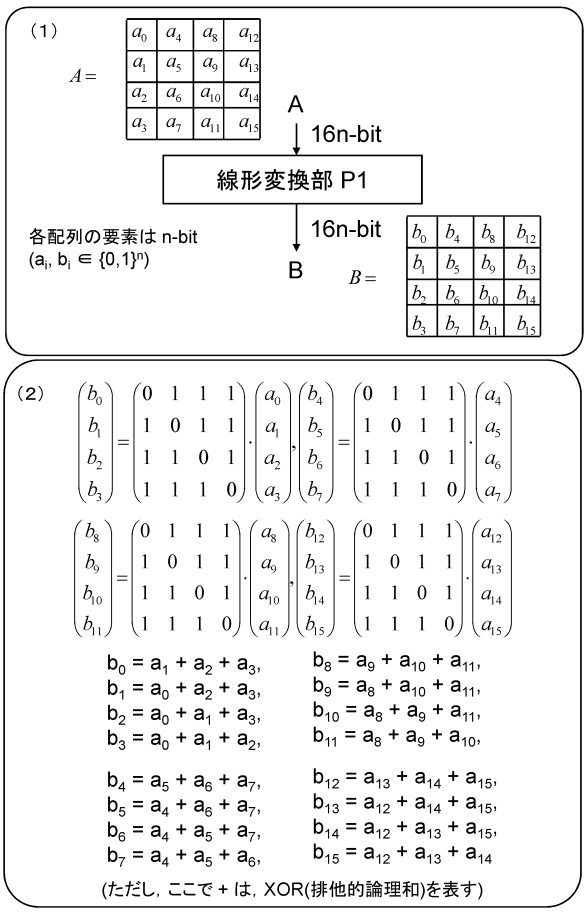
【図２３】



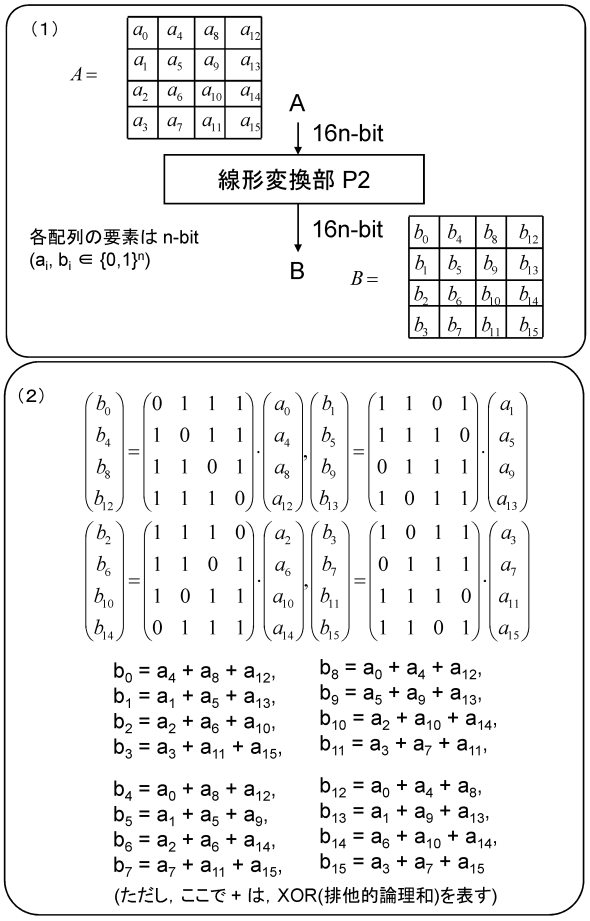
【図２４】



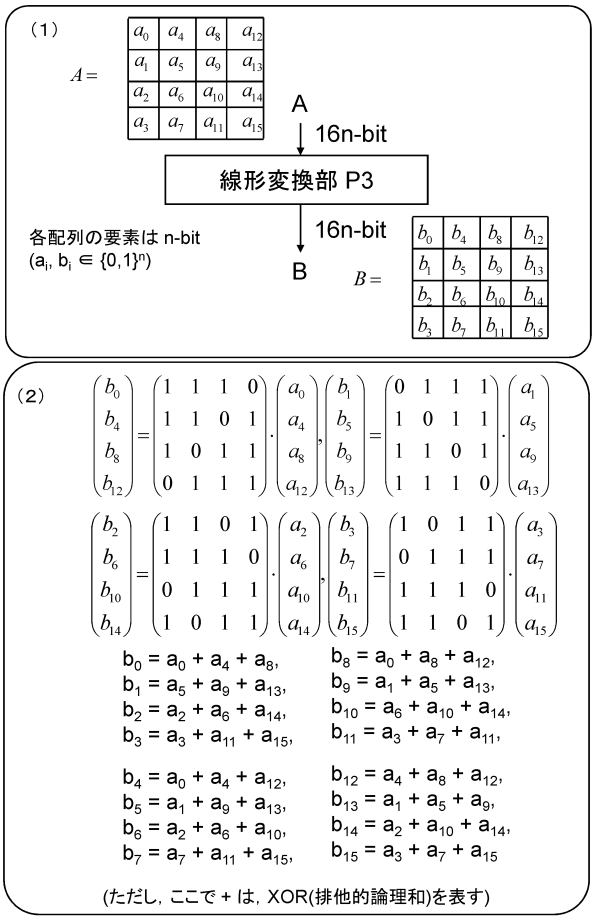
【図２５】



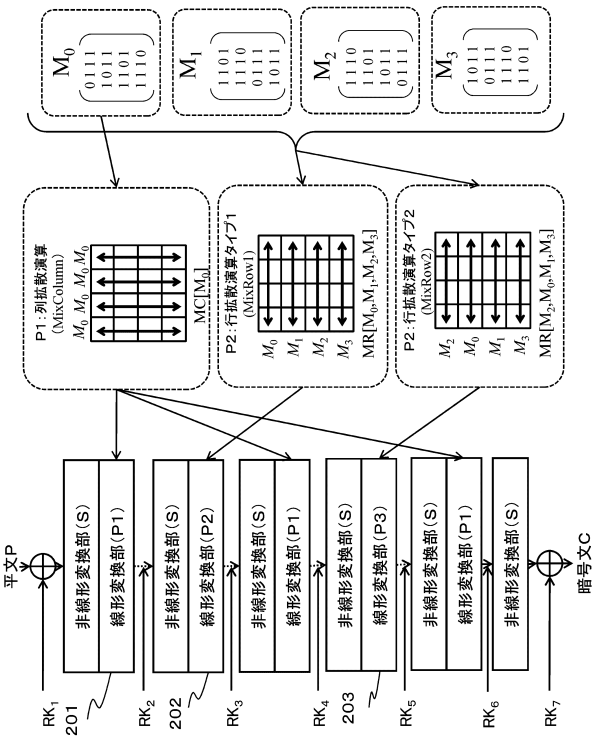
【図２６】



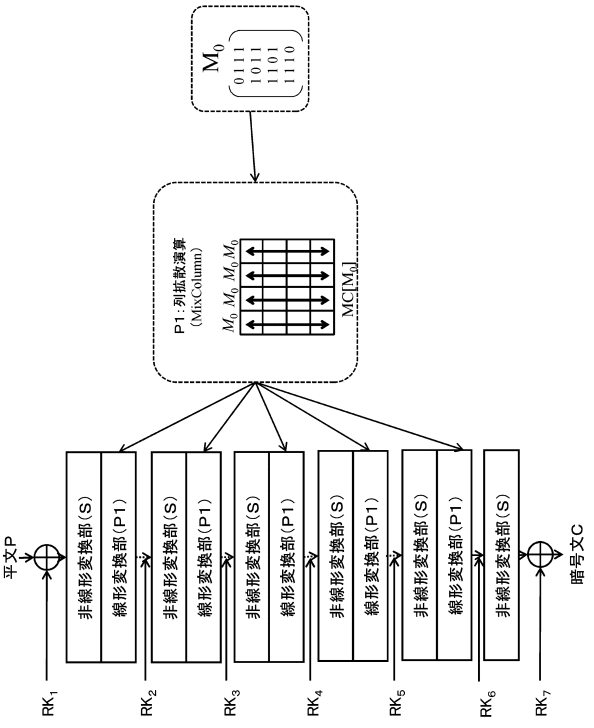
【図２７】



【図２８】



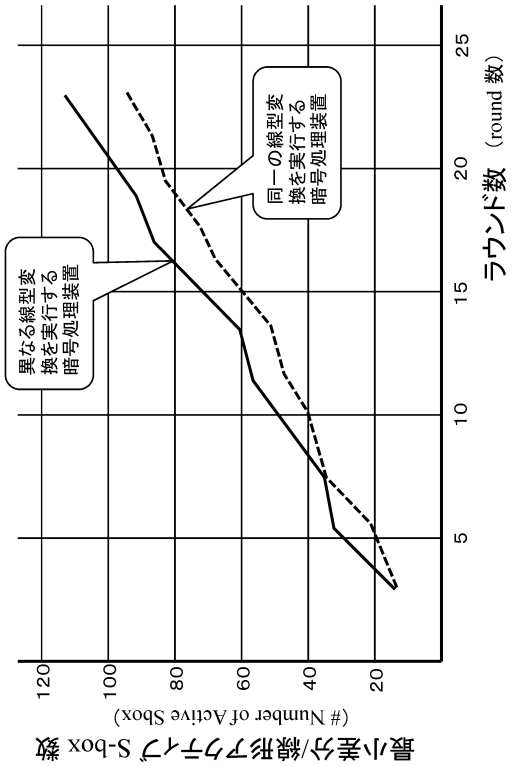
【図２９】



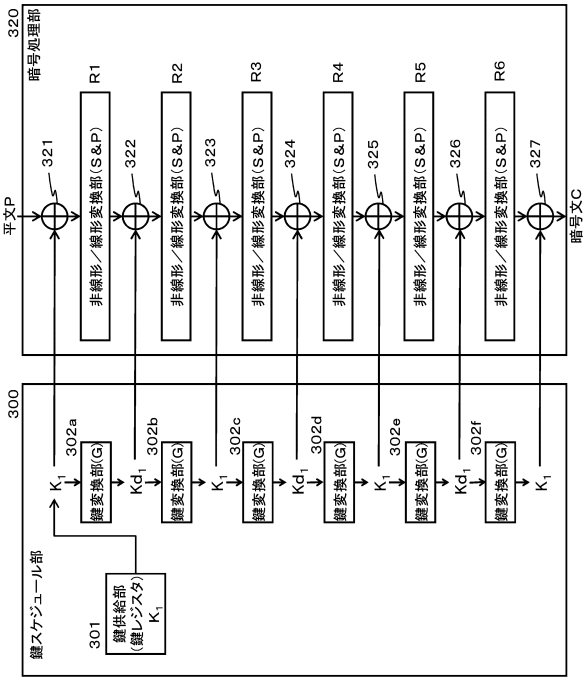
【図３０】

ラウンド数	従来型の同一の線形変換を実行する暗号処理構成の最小差分／線形 active S-box 数	異なる複数の線形変換を実行する暗号処理構成の最小差分／線形active S-box 数
4	16	16
6	20	28
8	32	32
10	36	44
12	48	56
14	52	60
16	64	72
18	68	84
20	80	88
22	84	100
24	96	112

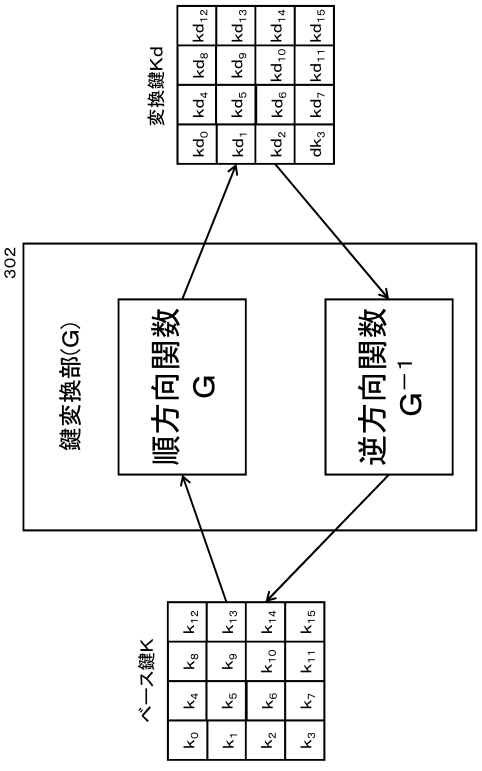
【図３１】



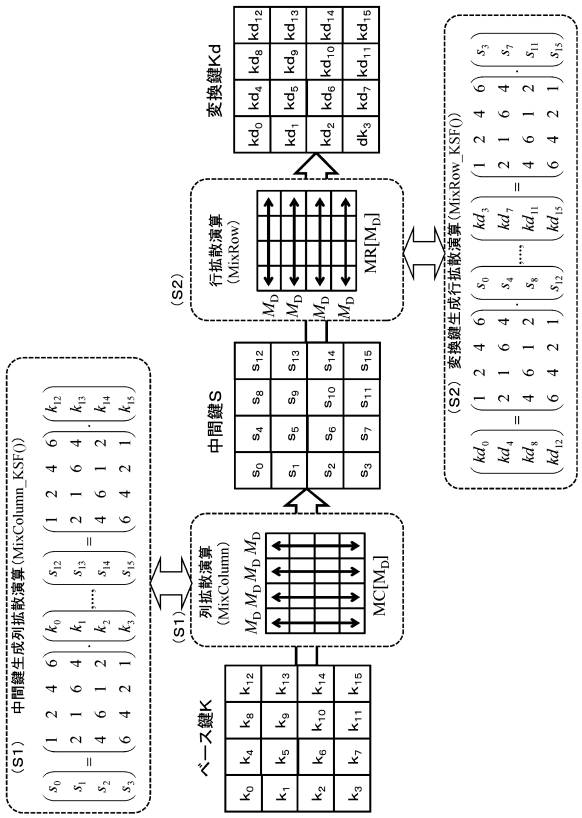
【図３２】



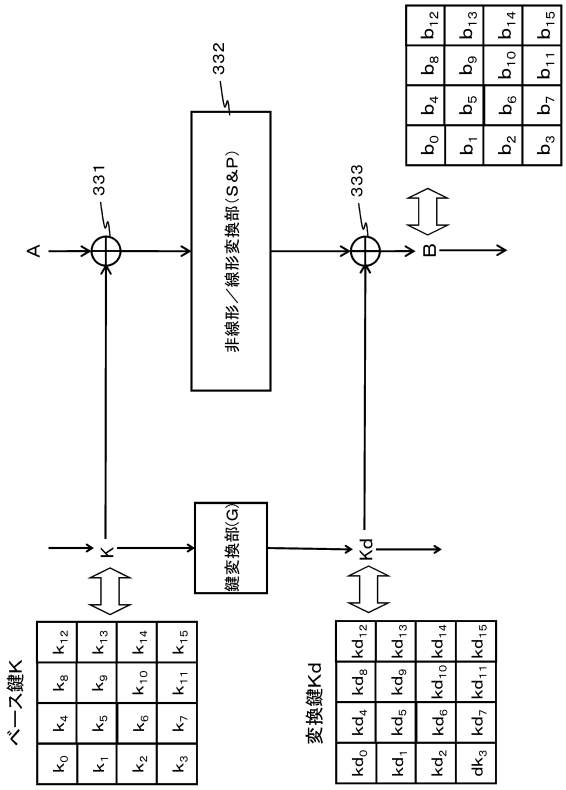
【図３３】



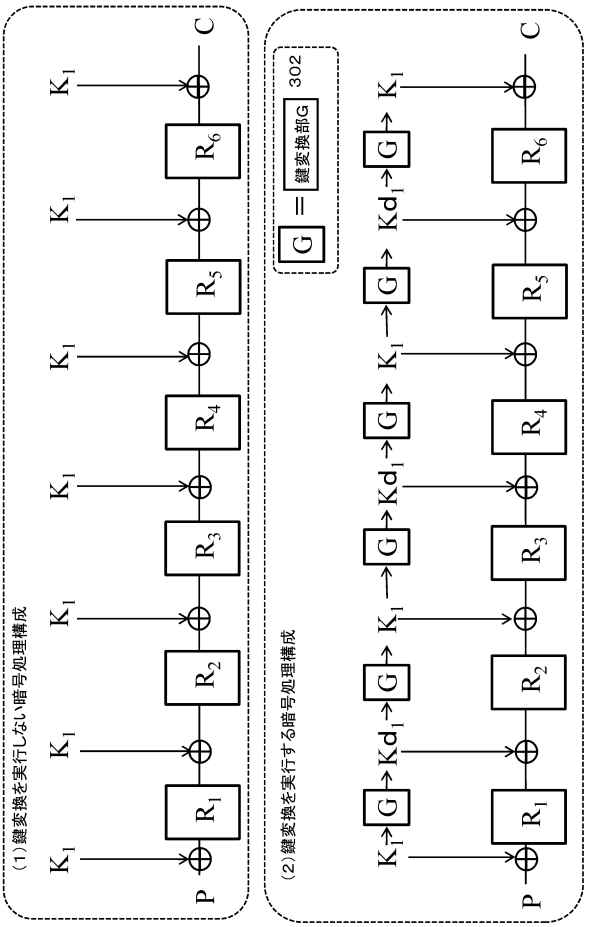
【図３４】



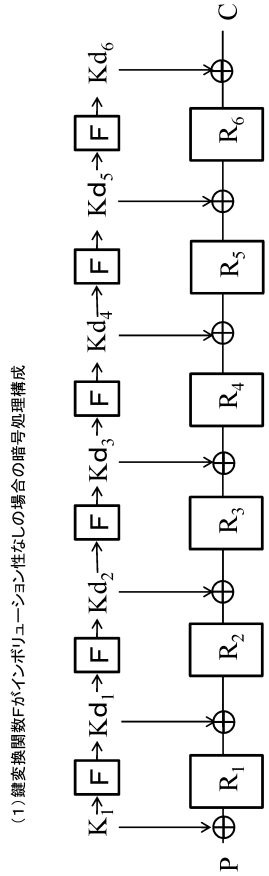
【図３５】



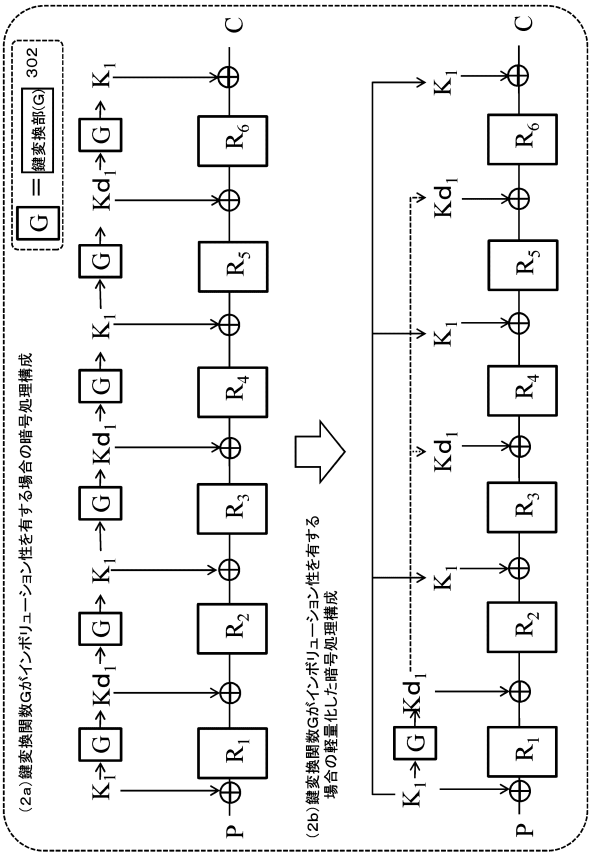
【図３６】



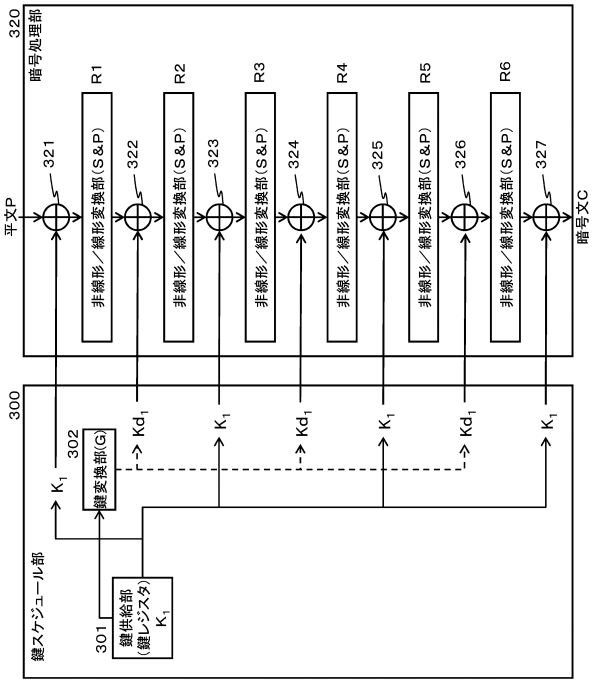
【図３７】



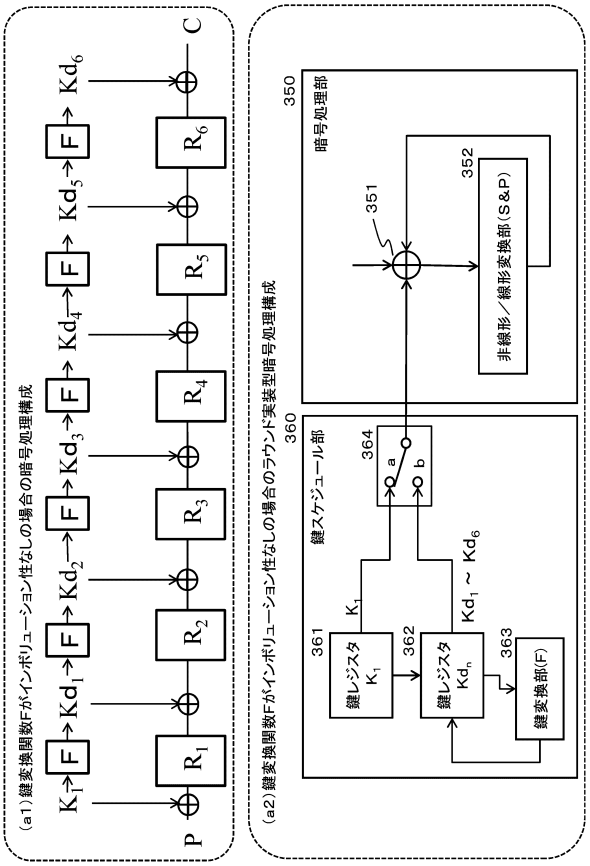
【図３８】



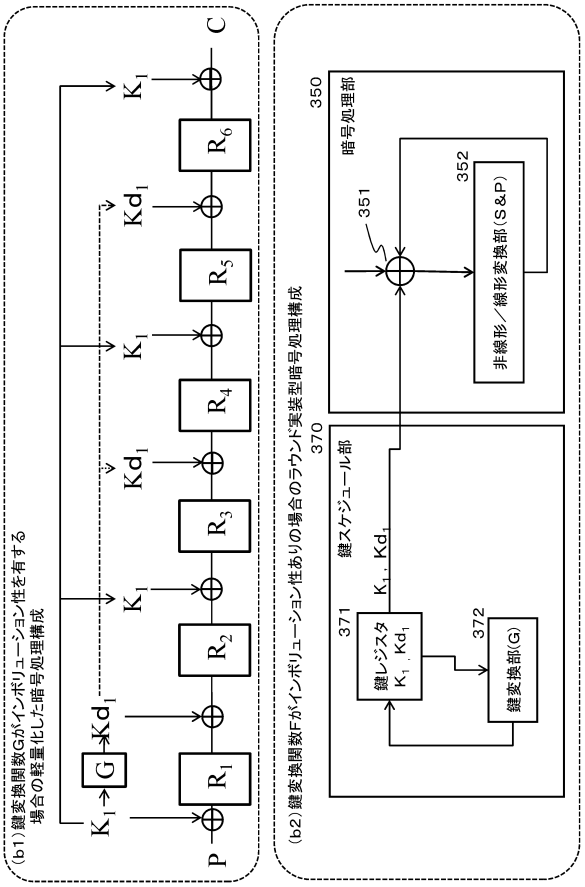
【図３９】



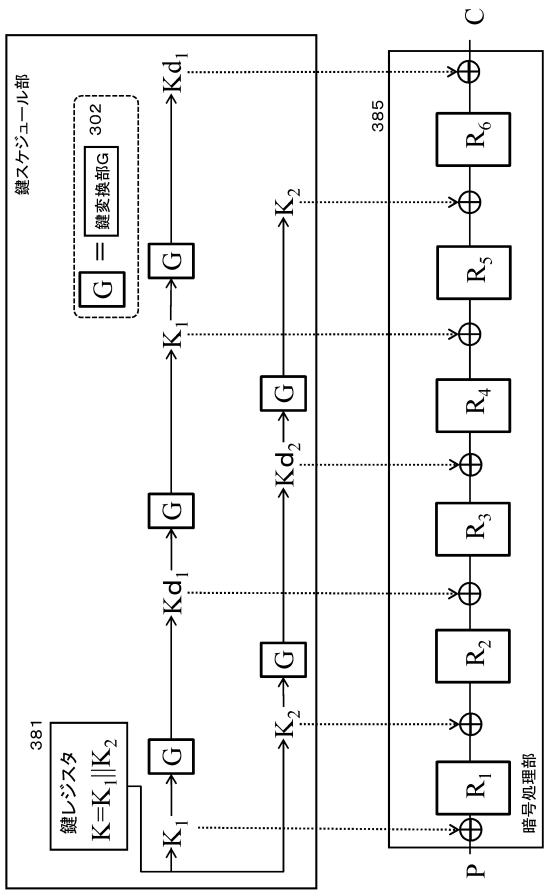
【図４０】



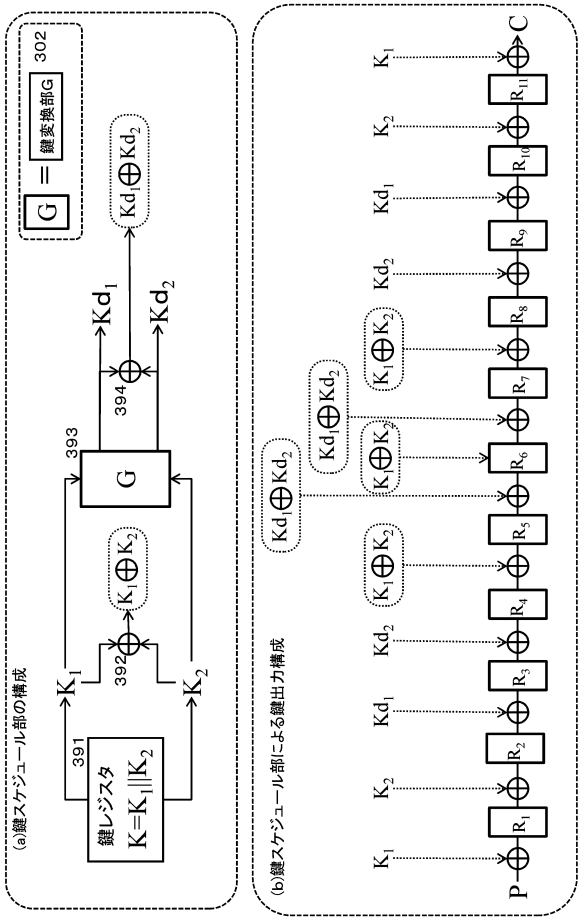
【図４１】



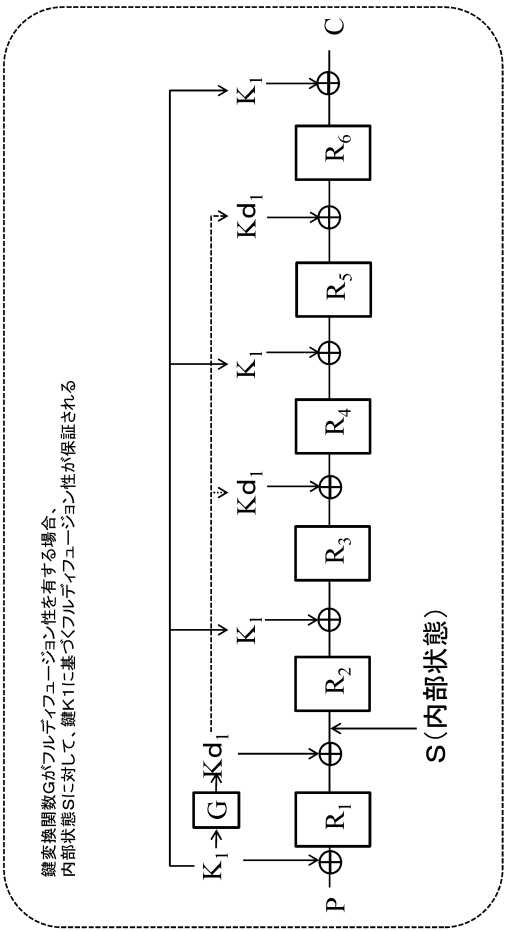
【図４２】



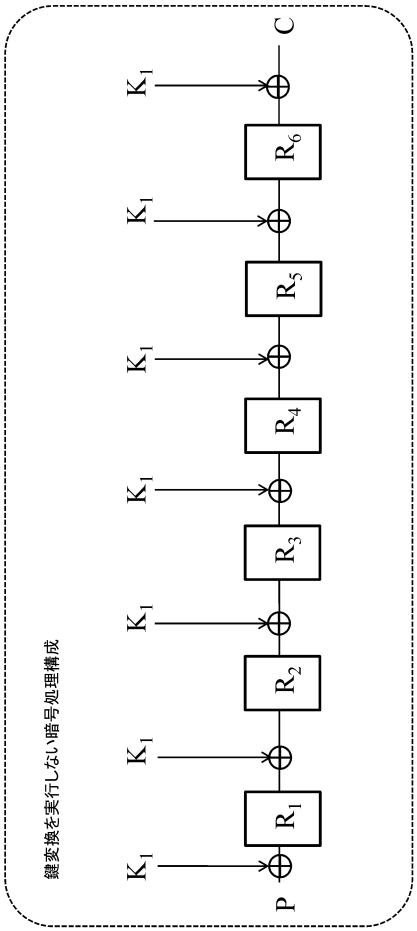
【図４３】



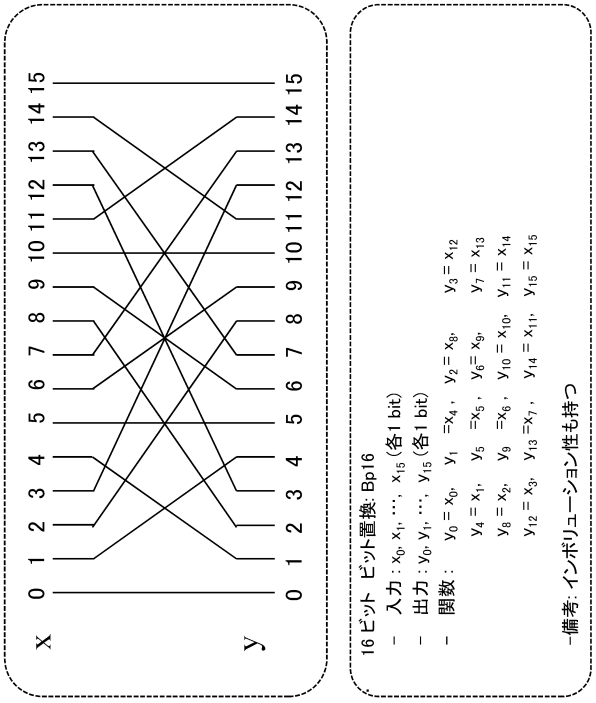
【図４４】



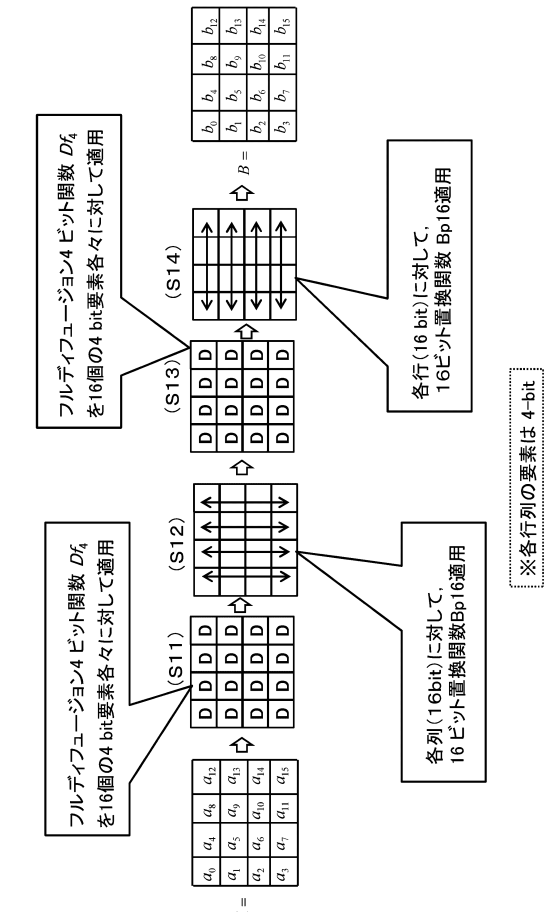
【図４５】



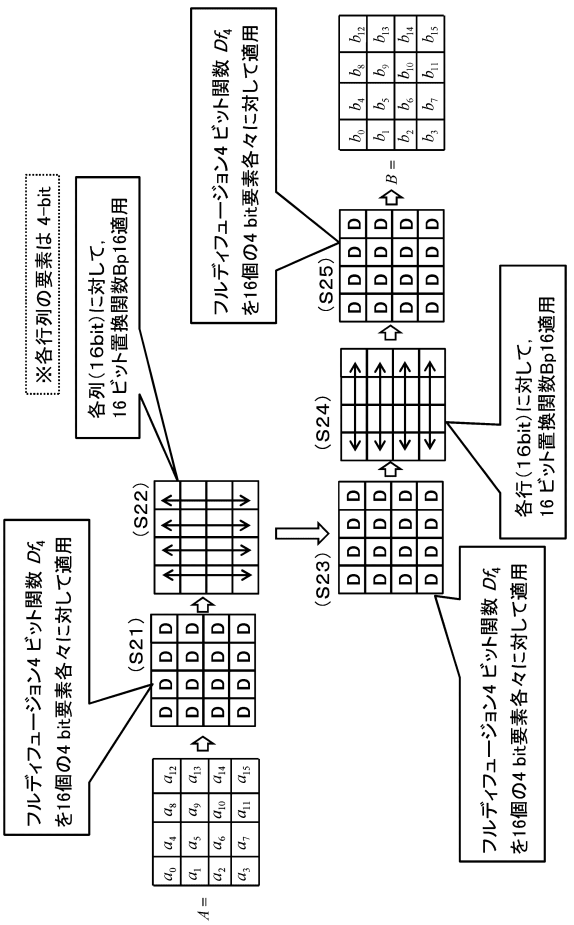
【図４６】



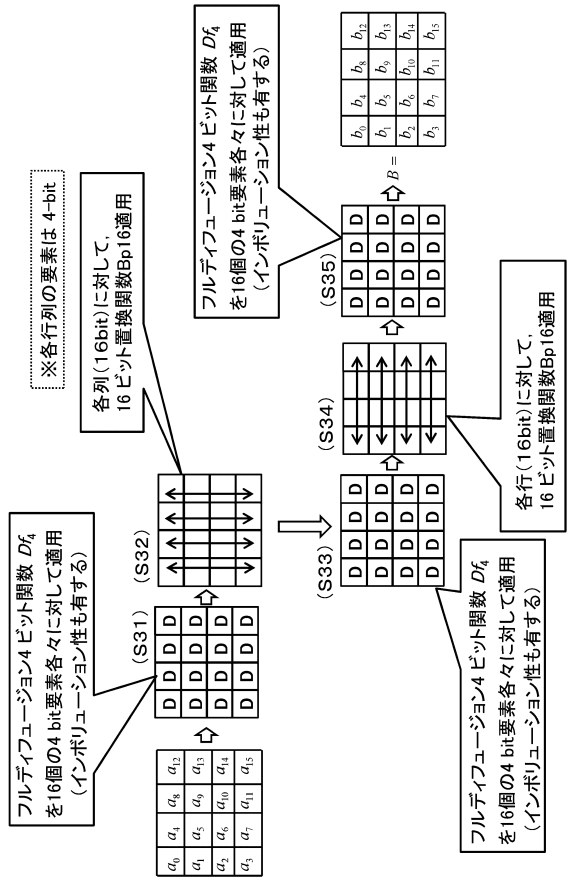
【図４７】



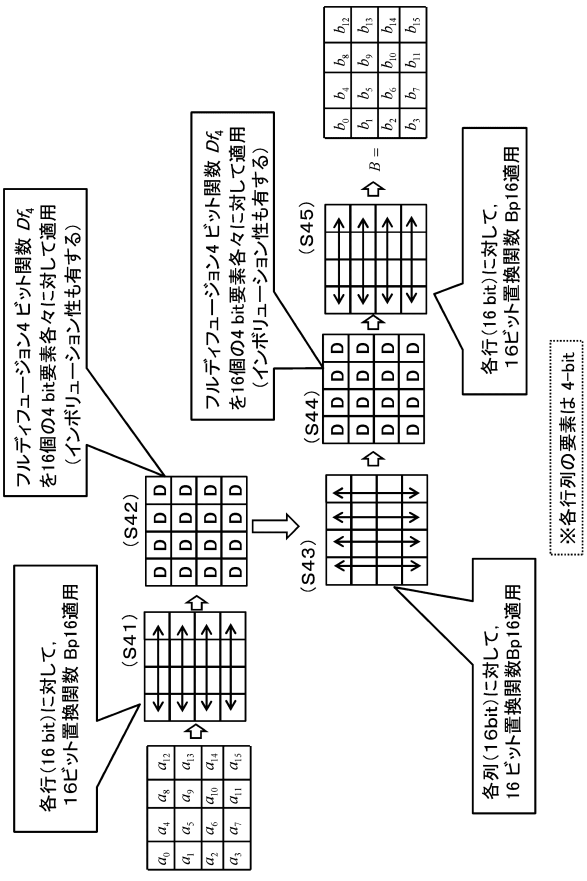
【図４８】



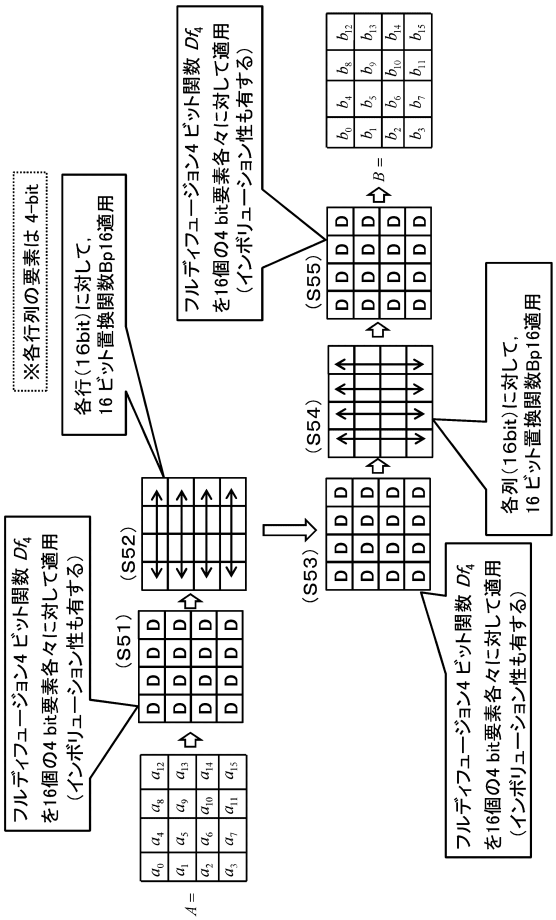
【図４９】



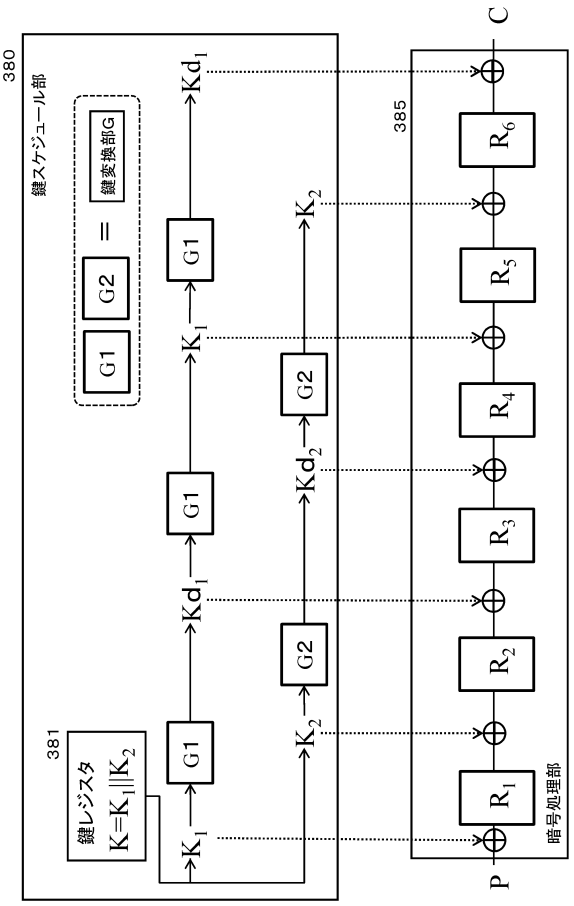
【図５０】



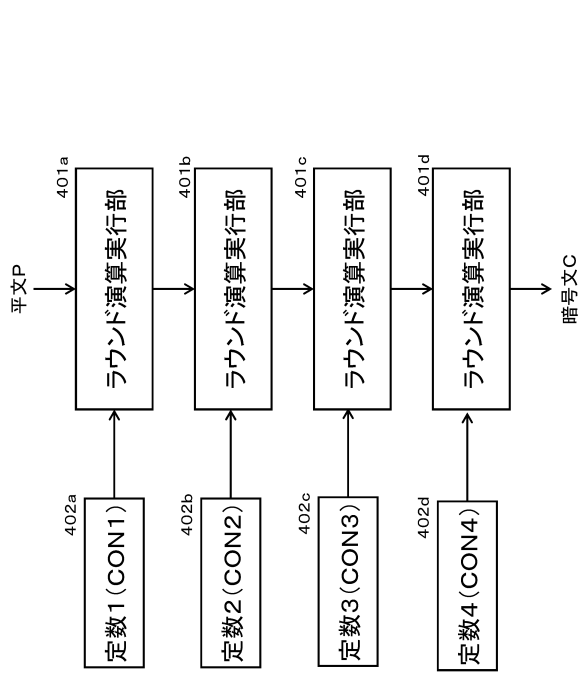
【図５１】



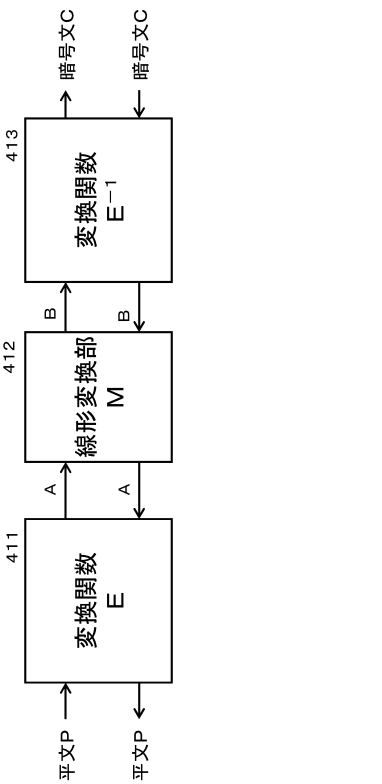
【図５２】



【図５３】



【図５４】



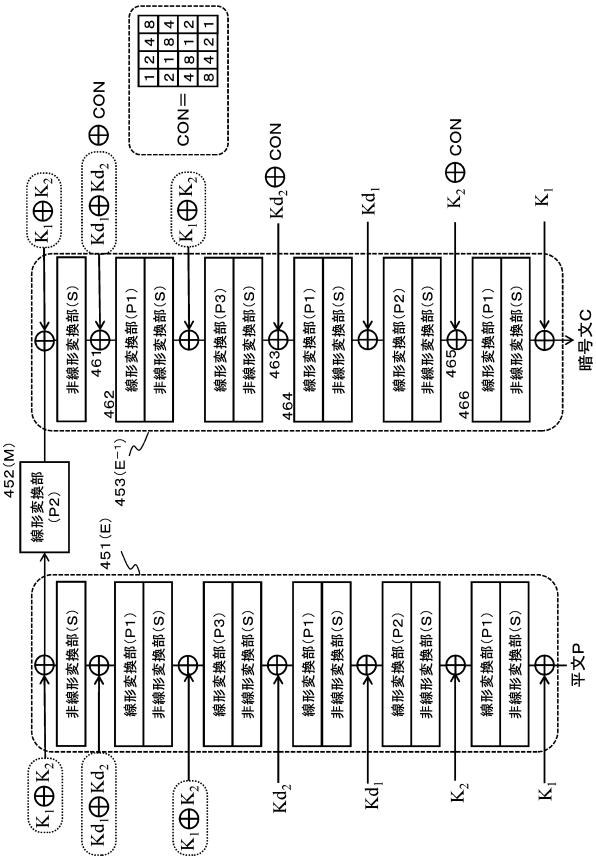
【図５５】

【図５６】

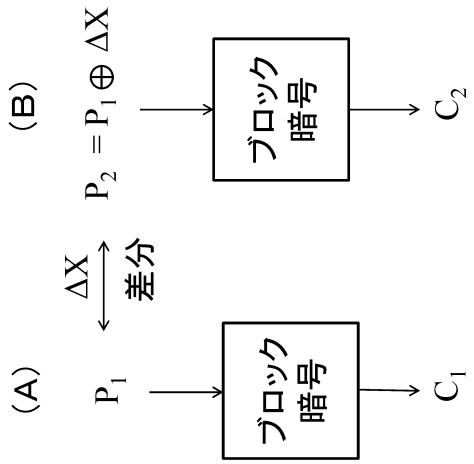
【図５７】

【図５８】

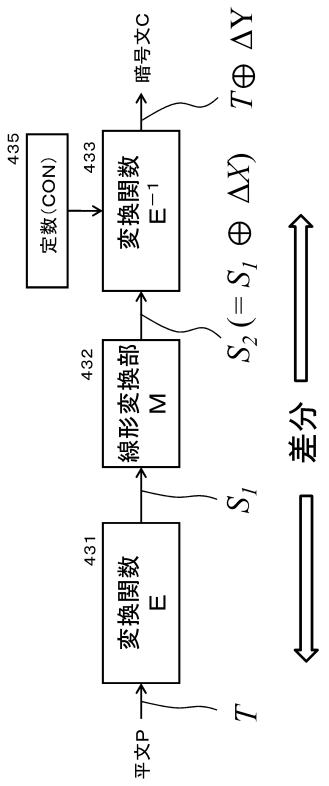
【図５９】



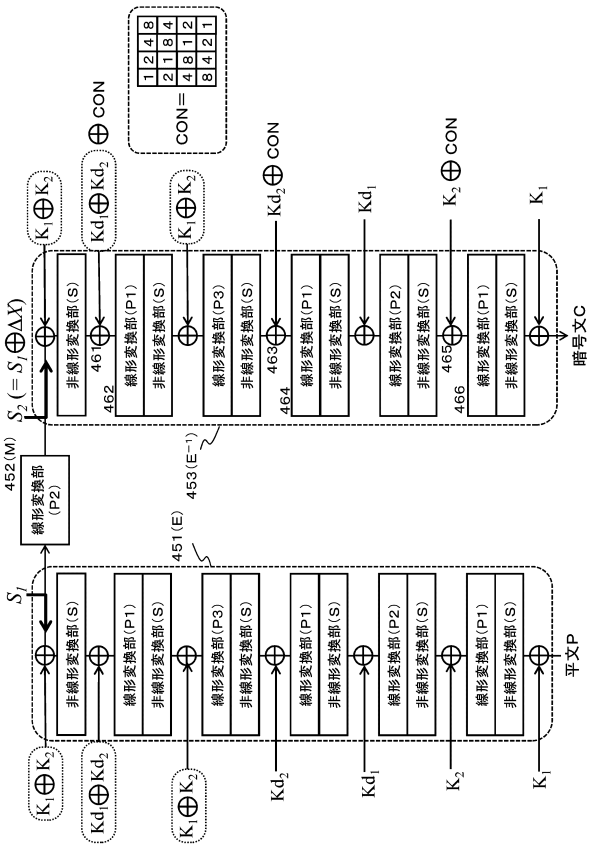
【図６０】



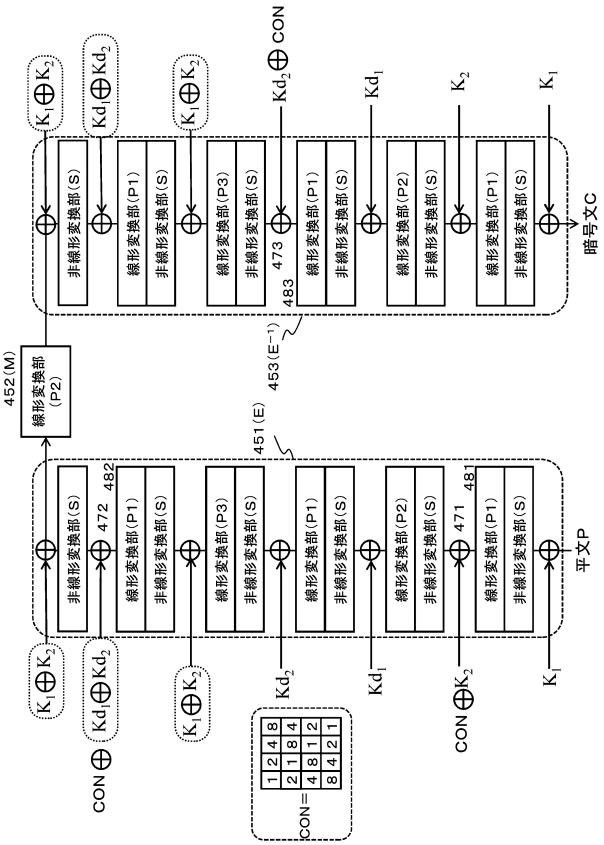
【図６１】



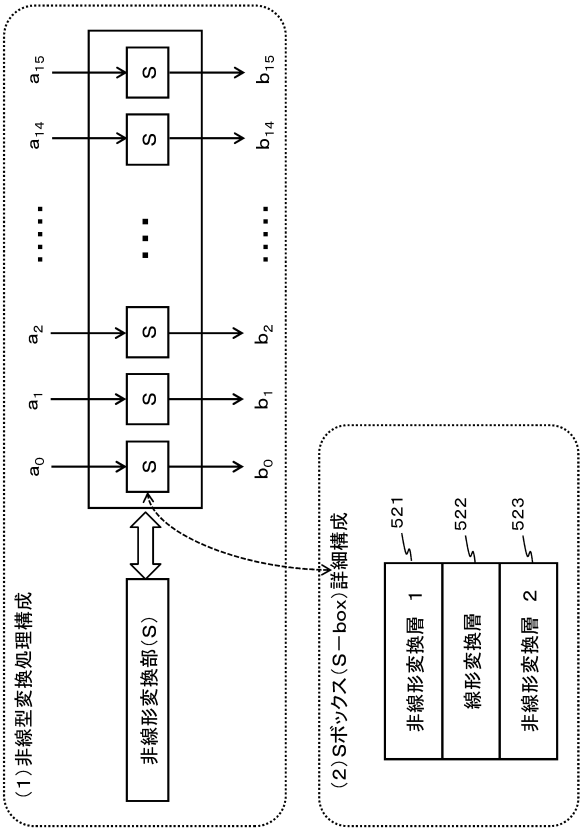
【図６２】



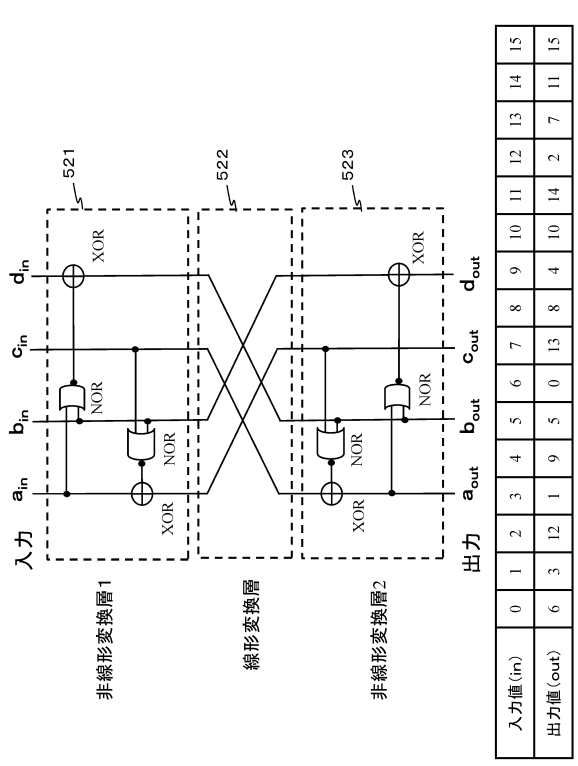
【図６３】



【図６４】

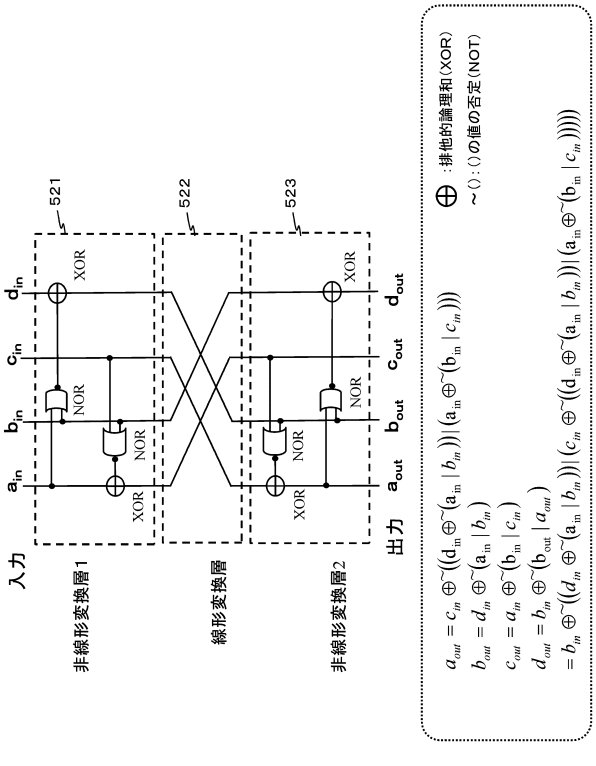


【図６５】

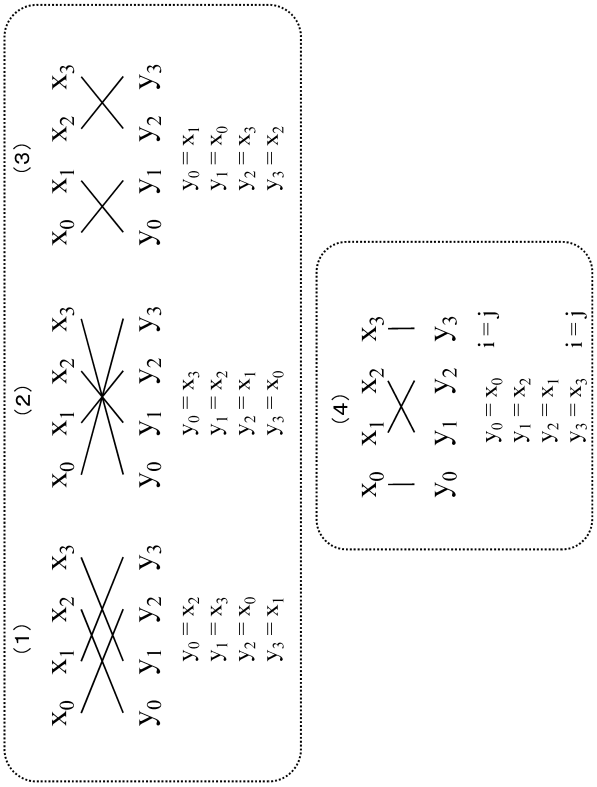


入力値(in)	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
出力値(out)	6	3	12	1	9	5	0	13	8	4	10	14	2	7	11	15

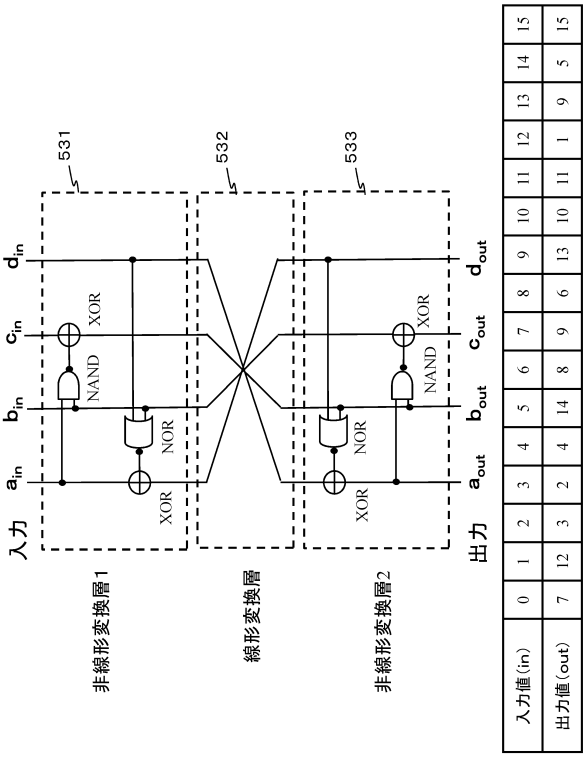
【図６６】



【図６７】

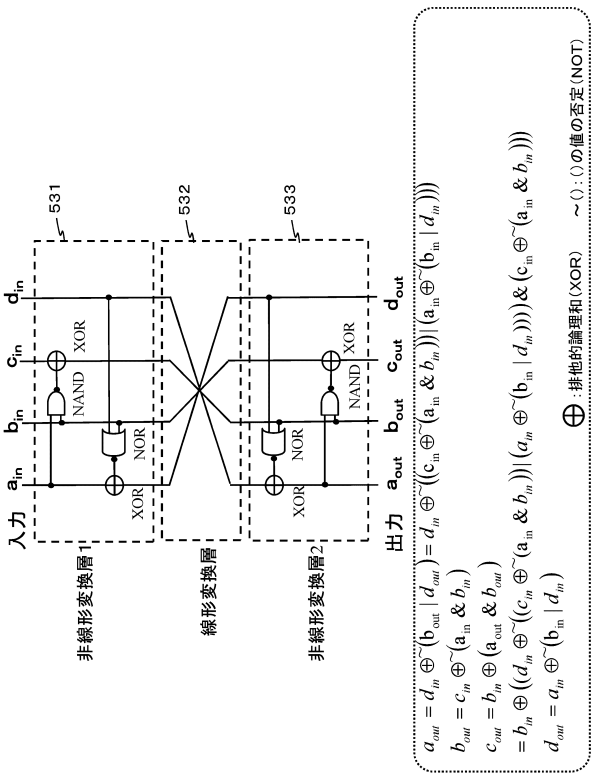


【図６８】

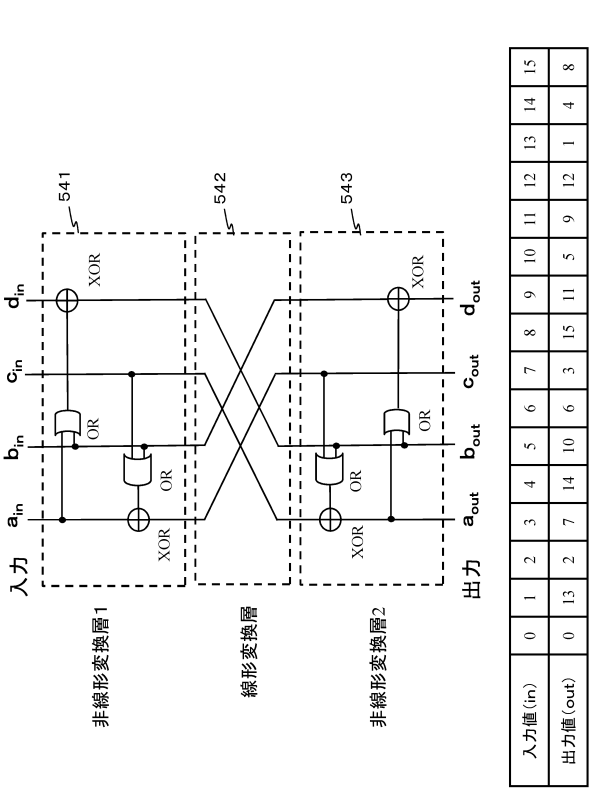


入力値(in)	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
出力値(out)	7	12	3	2	4	14	8	9	6	13	10	11	1	9	5	15

【図６９】

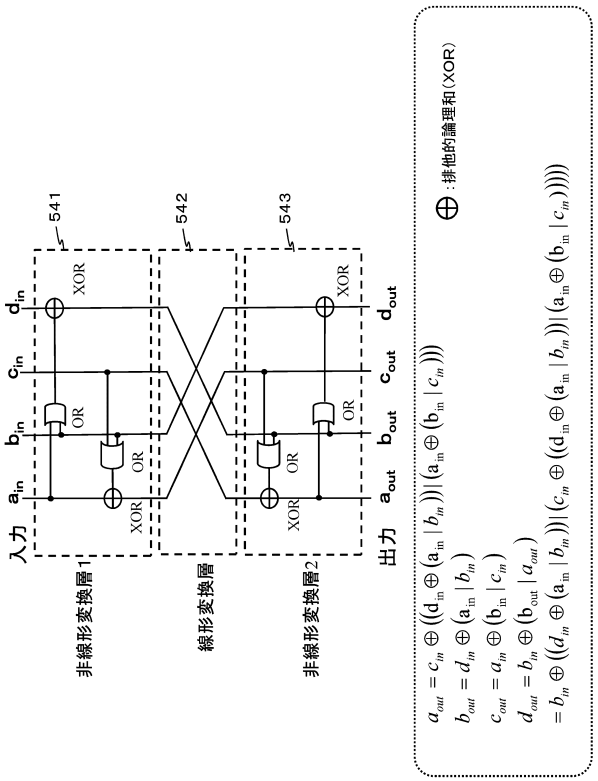


【図７０】

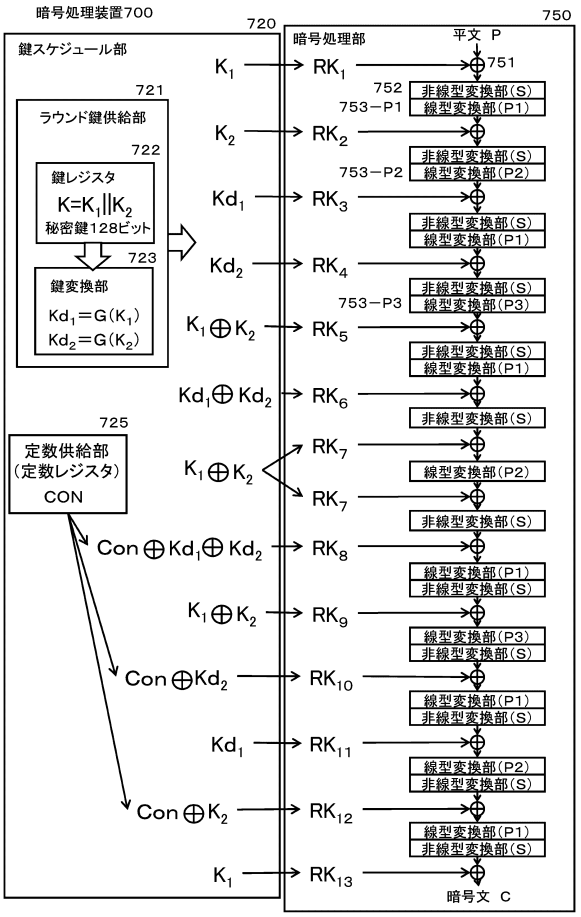


入力値(in)	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
出力値(out)	0	13	2	7	14	10	6	3	15	11	5	9	12	1	4	8

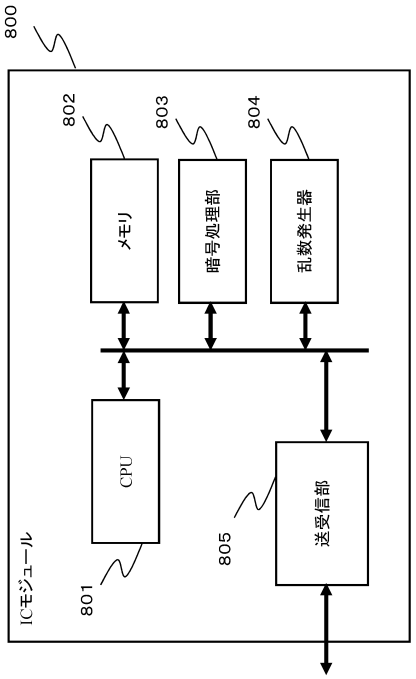
【図７１】



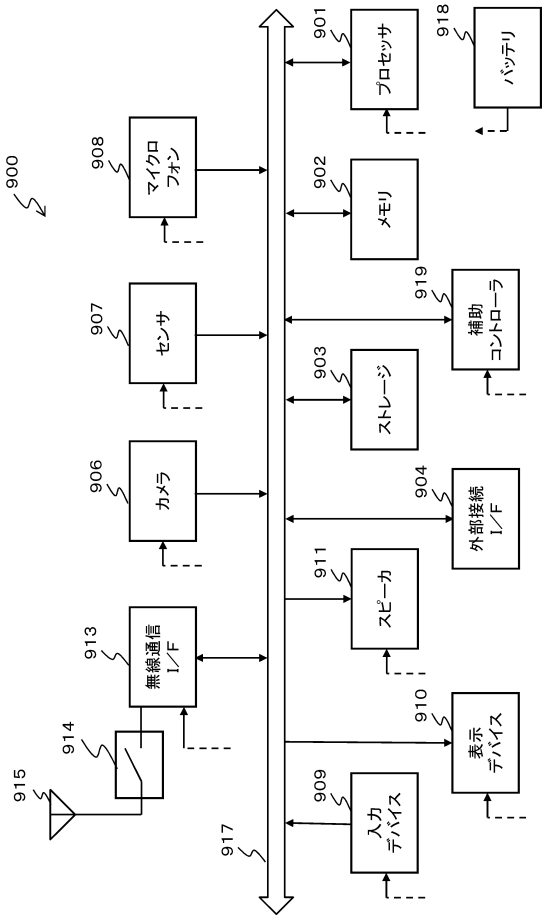
【図７２】



【図７３】



【図７４】



フロントページの続き

(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　五十部　孝典
東京都港区港南１丁目７番１号　ソニー株式会社内

審査官　中里　裕正

(56)参考文献　BORGHOFF, J. et al., PRINCE - A Low-latency Block Cipher for Pervasive Computing Applications, Cryptology ePrint Archive, [online], ２０１２年　9月１３日, Report 2012/529, [2018年7月31日検索], ＵＲＬ, https://eprint.iacr.org/2012/529/20120913:093817
WANG, C. and HEYS, H. M., AN ULTRA COMPACT BLOCK CIPHER FOR SERIALIZED ARCHITECTURE IMPLEMENTATIONS, 2009 Canadian Conference on Electrical and Computer Engineering, ２００９年, p.1085-1090
BARRETO, P. S. L. M. and RIJMEN, V., The ANUBIS Block Cipher, First Open NESSIE Workshop, [online], ２０００年１１月, [2018年7月31日検索], ＵＲＬ, https://www.cosic.esat.kuleuven.be/nessie/workshop/submissions/anubis.zip

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