

(12) 特許協力条約に基づいて公開された国際出願

(19) 世界知的所有権機関
国際事務局

(43) 国際公開日
2015年10月1日(01.10.2015)

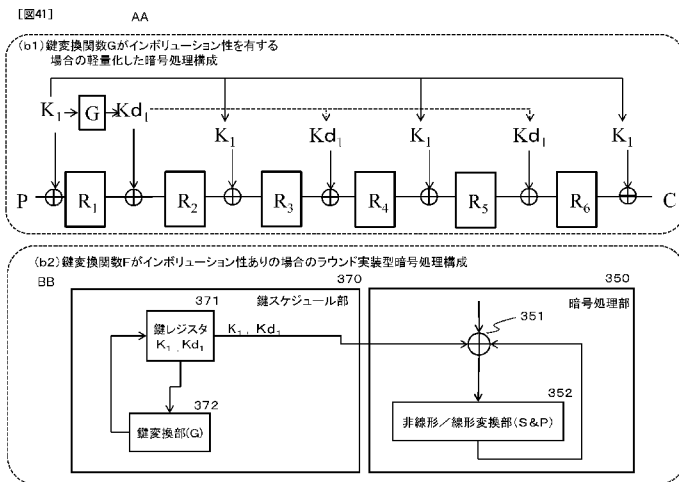


(10) 国際公開番号
WO 2015/146430 A1

- (51) 国際特許分類:
G09C 1/00 (2006.01) H04L 9/06 (2006.01)
 - (21) 国際出願番号: PCT/JP2015/055280
 - (22) 国際出願日: 2015年2月24日(24.02.2015)
 - (25) 国際出願の言語: 日本語
 - (26) 国際公開の言語: 日本語
 - (30) 優先権データ:
特願 2014-068291 2014年3月28日(28.03.2014) JP
 - (71) 出願人: ソニー株式会社(SONY CORPORATION) [JP/JP]; 〒1080075 東京都港区港南1丁目7番1号 Tokyo (JP).
 - (72) 発明者: 渋谷 香士(SHIBUTANI, Kyoji); 〒1080075 東京都港区港南1丁目7番1号 ソニー株式会社内 Tokyo (JP). 五十部 孝典(ISOBE, Takanori); 〒1080075 東京都港区港南1丁目7番1号 ソニー株式会社内 Tokyo (JP).
 - (74) 代理人: 宮田 正昭, 外(MIYATA, Masaaki et al.); 〒1040032 東京都中央区八丁堀三丁目25番9号 KSKビル西館8階 特許業務法人 大同特許事務所 Tokyo (JP).
 - (81) 指定国 (表示のない限り、全ての種類の国内保護が可能): AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BN, BR, BW, BY, BZ, CA, CH, CL, CN, CO, CR, CU, CZ, DE, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IR, IS, KE, KG, KN, KP, KR, KZ, LA, LC, LK, LR, LS, LU, LY, MA, MD, ME, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PA, PE, PG, PH, PL, PT, QA, RO, RS, RU, RW, SA, SC, SD, SE, SG, SK, SL, SM, ST, SV, SY, TH, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, ZA, ZM, ZW.
 - (84) 指定国 (表示のない限り、全ての種類の広域保護が可能): ARIPO (BW, GH, GM, KE, LR, LS, MW, MZ, NA, RW, SD, SL, ST, SZ, TZ, UG, ZM, ZW), ユーラシア (AM, AZ, BY, KG, KZ, RU, TJ, TM), ヨーロッパ (AL, AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV, MC, MK, MT, NL, NO, PL, PT, RO, RS, SE, SI, SK, SM, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, KM, ML, MR, NE, SN, TD, TG).
- 添付公開書類:
— 国際調査報告 (条約第21条(3))

(54) Title: ENCRYPTION PROCESSING DEVICE, AND ENCRYPTION PROCESSING METHOD AND PROGRAM

(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



350 Encryption processing unit
 352 Non-linear/linear conversion unit
 370 Key scheduling unit
 371 Key register
 372 Key conversion unit
 AA Streamlined encryption processing configuration when key conversion function (G) has involutory properties
 BB Encryption processing configuration in which rounds are implemented when key conversion function (F) has involutory properties

(57) Abstract: The present invention realizes a highly safe encryption processing structure that improves resistance to various attacks. Provided are: an encryption processing unit that generates output data by repeating a round operation on input data; and a key scheduling unit that outputs to the encryption processing unit a round key to be applied in the round operations by the encryption processing unit. The key scheduling unit has a key register that stores a secret key (K) and a key conversion unit that generates a conversion key (Kd) by conversion processing that applies a conversion function (G) to the secret key (K), and uses the secret key (K) and conversion key (Kd) as the round key to be outputted to the encryption processing unit or as data for generating the round key. A function that has involutory properties whereby the inverse function (G⁻¹) becomes the same function as the function (G) and has full diffusion properties is utilized as the conversion function (G).

(57) 要約:

[続葉有]

WO 2015/146430 A1



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を実現する。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へ出力する鍵スケジュール部を有し、鍵スケジュール部は、秘密鍵 K を格納した鍵レジスタと、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 K_d を生成する鍵変換部を有し、秘密鍵 K 、および変換鍵 K_d を暗号処理部へ出力するラウンド鍵、またはラウンド鍵生成用データとする。変換関数 G として、逆関数 G^{-1} が関数 G と同じ関数となるインボリューション性と、フルディフュージョン性を有する関数を適用する。

明 細 書

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

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、

前記鍵スケジュール部は、

秘密鍵 K を格納した鍵レジスタと、

前記秘密鍵 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 K_d を生成する鍵変換部を有し、

前記秘密鍵 K 、および前記変換鍵 K_d を前記暗号処理部に出力するラウンド鍵、またはラウンド鍵生成用データとする構成であり、

前記変換関数 G は、逆関数 G^{-1} が関数 G と同じ関数となるインボリュージョン性を有する関数であ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、

前記鍵スケジュール部は、

秘密鍵 K を格納した鍵レジスタと、

前記秘密鍵 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 K_d を生成する鍵変換部を有し、

前記秘密鍵 K 、および前記変換鍵 K_d を前記暗号処理部に出力するラウンド鍵、またはラウンド鍵生成用データとする構成であり、

前記秘密鍵 K 、および前記変換鍵 K_d は、いずれも各要素が1ビット以上の $m \times n$ 個の要素からなる状態であり、

前記変換関数 G は、前記秘密鍵 K の状態要素の全てが、前記変換鍵 K_d の状態要素全てに影響を及ぼすフルディフュージョン性を有する関数であ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へ出力する鍵スケジュール部を有し、
前記鍵スケジュール部は、
鍵レジスタに格納した秘密鍵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K_dを生成し、
前記秘密鍵K、および前記変換鍵K_dを前記暗号処理部へ出力するラウンド鍵、またはラウンド鍵生成用データとする処理を行ない、
前記変換関数Gは、逆関数G⁻¹が関数Gと同じ関数となるインボリュージョン性を有する関数であ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へ出力する鍵スケジュール部を有し、
前記鍵スケジュール部は、
鍵レジスタに格納した秘密鍵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K_dを生成し、
前記秘密鍵K、および前記変換鍵K_dを前記暗号処理部へ出力するラウンド鍵、またはラウンド鍵生成用データとする処理を行ない、
前記秘密鍵K、および前記変換鍵K_dは、いずれも各要素が1ビット以上のm×n個の要素からなる状態であり、

前記変換関数 G は、前記秘密鍵 K の状態要素の全てが、前記変換鍵 K_d の状態要素全てに影響を及ぼすフルディフュージョン性を有する関数であ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へ出力する鍵スケジュール部を有し、
前記プログラムは、前記鍵スケジュール部に、
鍵レジスタに格納した秘密鍵 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 K_d を生成する処理と、
前記秘密鍵 K 、および前記変換鍵 K_d を前記暗号処理部へ出力するラウンド鍵、またはラウンド鍵生成用データとする処理を実行させ、
前記変換関数 G は、逆関数 G^{-1} が関数 G と同じ関数となるインボリュージョン性を有する関数であ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へ出力する鍵スケジュール部を有し、
前記プログラムは、前記鍵スケジュール部に、
鍵レジスタに格納した秘密鍵 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 K_d を生成する処理と、
前記秘密鍵 K 、および前記変換鍵 K_d を前記暗号処理部へ出力するラウン

ド鍵、またはラウンド鍵生成用データとする処理を実行させ、

前記秘密鍵 K 、および前記変換鍵 K_d は、いずれも各要素が1ビット以上の $m \times n$ 個の要素からなる状態であり、

前記変換関数 G は、前記秘密鍵 K の状態要素の全てが、前記変換鍵 K_d の状態要素全てに影響を及ぼすフルディフュージョン性を有する関数であ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を向上させた安全性の高い暗号処理構成が実現される。

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、鍵スケジュール部は、秘密鍵 K を格納した鍵レジスタと、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 K_d を生成する鍵変換部を有し、秘密鍵 K 、および変換鍵 K_d を暗号処理部に出力するラウンド鍵、またはラウンド鍵生成用データとする。変換関数 G として、逆関数 G^{-1} が関数 G と同じ関数となるインボリューション性と、フルディフュージョン性を有する関数を適用する。

本構成により拡散特性が向上し、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] S P N 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] F e i s t e l 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 F e i s t e l 構造の一例について説明する図である。

[図8] 拡張 F e i s t e l 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P 1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P 2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P 3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]鍵変換関数がインポ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]鍵変換関数がインポ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]フルディフュージョン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]フルディフュージョン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]フルディフュージョン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]フルディフュージョン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]フルディフュージョン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]インボリュ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]インボリュ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 - b o x) 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. 鍵変換部のインポ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 $m \times m$ 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 $m \times n$ 配列データとして表現することがある。この $m \times n$ 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 $m \times n$ 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 $A = (a_0 a_1 a_2 \cdots a_{m n - 2} a_{m n - 1})$ である。

なお、 a_0 はMSB、 $a_{m n - 1}$ 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 $GF(2^s)^{m n}$ の要素であり、

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

なお、 b_0 はMSB、 $b_{m n - 1}$ はLSB側のビットデータである。

- [0038] 図に示すように、 $m \times n$ 配列のステートには $m \times n$ 個の要素が含まれる。
 例えば図11に示すステートAには $a_0 \sim a_{m n - 1}$ の $m n$ 個の要素が含まれる。
 ステートBの要素は、 $b_0 \sim b_{m n - 1}$ の $m n$ 個の要素である。

これらの $m n$ 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 $GF(2^4)^{4 \times 4}$ 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 $GF(2^4)^{4 \times 4}$ 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 $a_0 \sim a_{15}$ 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を状態として示すと、図12に示す各要素が4ビットデータからなる4×4配列を持つ状態Aとして表現できる。

[0041] 同様に、図12に示す状態Bには $b_0 \sim b_{15}$ 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を状態として示すと、図12に示す各要素が4ビットデータからなる4×4配列を持つ状態Bとして表現できる。

[0042] (1-8. 状態表現データに対する基本的演算)

次に、状態(状態表現データ)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、状態の各要素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、入力状態Aに対する非線形変換処理によって、状態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、

$$b_i = S(a_i)$$

$$i = 0, 1, \dots, 15,$$

である。

[0043] (2) 線形変換処理(P)

図13(2)に示すように、入力状態Aに対する線形変換処理によって、状態Bが生成されるとする。

4×4状態に対する線形変換処理は、例えば4×4状態の行ごとの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。

変換処理後の状態各要素4ビット単位の出力 b_i と入力 a_i の関係は、

$${}^t(b_i, b_{i+4}, b_{i+8}, b_{i+12}) = M \times {}^t(a_i, a_{i+4}, a_{i+8}, a_{i+12})$$

2)

$$i = 0, 1, 2, 3,$$

である。なお、 tX 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、入力状態Aに対する排他的論理和演算処理によって、状態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状態表現された64ビットデータである。

変換処理後の状態各要素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、

$$b_i = a_i \text{ (XOR) } k_i$$

$$i = 0, 1, \dots, 15,$$

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

図15に示すように、

MC [X_0, X_1, \dots, 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, \dots, 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。

$B = MC [X_0, X_1, \dots, X_{n-1}] (A)$

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。

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

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

...

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

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]

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

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

[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 [X₀, X₁, X₂, X₃]

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状態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状態Bの要素は、以下の通りである。

$${}^t(b_0 b_1 b_2 b_3) = X_0 \times {}^t(a_0 a_1 a_2 a_3),$$

$${}^t(b_4 b_5 b_6 b_7) = X_1 \times {}^t(a_4 a_5 a_6 a_7),$$

$${}^t(b_8 b_9 b_{10} b_{11}) = X_2 \times {}^t(a_8 a_9 a_{10} a_{11}),$$

$${}^t(b_{12} b_{13} b_{14} b_{15}) = X_3 \times {}^t(a_{12} a_{13} a_{14} a_{15}),$$

実際の状態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、状態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. 状態表現データに対する行拡散演算)

次に $m \times n$ のマトリクス配列として示された状態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, \dots, X_{m-1} の各々を、各要素がGF(2^s)上の要素からなる $n \times n$ 行列とする。

図17に示すように、

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

上記演算を、状態表現データの要素に対して、状態の各行（0～ $n-1$ ）の要素と、各行対応の行列 X_0, X_1, \dots, 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（Mix Row）を意味する。

[0057] 行拡散演算では、状態の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、状態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、入力データである状態Aに対して、行拡散演算を実行して出力データである状態Bを算出する演算式は、以下のように表現できる。

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

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状態Bの要素は、以下の通りである。

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

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

...

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

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

実際の状態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]

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

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

[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 [X₀, X₁, X₂, X₃]

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 [X₀, X₁, X₂, X₃] (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。

$${}^t (b_0 b_4 b_8 b_{12}) = X_0 \times {}^t (a_0 a_4 a_8 a_{12}),$$

$${}^t (b_1 b_5 b_9 b_{13}) = X_1 \times {}^t (a_1 a_5 a_9 a_{13}),$$

$${}^t (b_2 b_6 b_{10} b_{14}) = X_2 \times {}^t (a_2 a_6 a_{10} a_{14}),$$

$${}^t (b_3 b_7 b_{11} b_{15}) = X_3 \times {}^t (a_3 a_7 a_{11} a_{15}),$$

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. インポ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 K 1, K 2, . . . , K R 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。

$$C = E (P, K 1, K 2, \dots, K R)$$

[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、

$$P = D (C, k 1, k 2, \dots, k r)$$

となるが、

上記の復号関数 D が、以下を満たすとき、

$$D (C, k 1, k 2, \dots, k r) = E (C, K R, \dots, K 2, K 1)$$

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こと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、強い相関関係を見つけることで、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、 $n \times a$ ビットデータから $n \times b$ 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 ($\text{Branch}_n(\theta)$) を次のように定義する。

$$[0072] \quad \text{Branch}_n(\theta) = \min_{\alpha \neq 0} \{ \text{hw}_n(\alpha) + \text{hw}_n(\theta(\alpha)) \}$$

ただし、

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

、
 $\text{hw}_n(Y)$ 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 $\text{Branch}_n(\theta)$ 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、

$$A A^{-1} = A^{-1} A = E、$$

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對一に決定される。非線形変換の前後では、入力差分と出力差分の関係は一對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对一に決定される。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对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、前述した状態表現データであり、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状態表現された 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

これらの各処理を状態に対する処理として実行する。状態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 - b o x) 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 $4 \times 16 = 64$ 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換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 P 1, P 2, P 3 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 P 1 ~ P 3 には、図24(1)に示す4つの異なる行列 $M_0 \sim M_3$ 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) $M_0 \sim M_3$ を組み合わせて構成する。

[0096] [数5]

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

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

[0097] 線形変換処理 P 1 ~ P 3 には、上記の4つの異なる行列 $M_0 \sim M_3$ 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個の要素からなる状態を示している。すなわち64ビットの4×4状態である。

この4×4状態の入力データに対して、行列M₀~M₃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状態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M₀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 (M_i × C_{olum}) 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 [M₀] は、状態の各列に対して、同一の行列M₀を適用した行列演算を示す式であり、状態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状態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 ($M_i \times Row$) 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 ($M_i \times Row$) 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 ($M_i \times Row$) 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 ($M_i \times Row$) 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 ($M_i \times Row1$)、線形変換処理P3を行拡散演算タイプ2 ($M_i \times Row2$)、

と呼ぶ。

線形変換処理 P 1 は、列拡散演算 (M i x C o l u m n) である。

[0106] これら 3 つの線形変換処理 P 1 ~ P 3 の具体的な行列演算の計算処理例について、図 2 5 以下を参照して説明する。

[0107] 図 2 5 は、線形変換処理 P 1、すなわち列拡散演算 (M i x C o l u m n) の具体的な計算処理例を説明する図である。

図 2 5 (1) には、線形変換部 P 1 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 1 6 個の要素 $a_0 \sim a_{15}$ からなる状態である。

。

出力 B も n ビットデータの 1 6 個の要素 $b_0 \sim b_{15}$ からなる状態である。

。

なお、入出力データの各要素 a_i, b_i (ただし $i = 0 \sim 15$) 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 6 4 ビットである。

[0108] 図 2 5 (2) には、線形変換処理 P 1、すなわち列拡散演算 (M i x C o l u m n) の具体的な計算処理例を示している。

線形変換処理 P 1 として行われる列拡散演算 (M i x C o l u m n) 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 $b_0 \sim b_{15}$ は、行列 M_0 と、入力Aの16個の要素 $a_0 \sim a_{15}$ に基づいて以下の演算によって算出される。

$$\begin{aligned} b_0 &= a_1 (+) a_2 (+) a_3, \\ b_1 &= a_0 (+) a_2 (+) a_3, \\ b_2 &= a_0 (+) a_1 (+) a_3, \\ b_3 &= a_0 (+) a_1 (+) a_2, \\ b_4 &= a_5 (+) a_6 (+) a_7, \\ b_5 &= a_4 (+) a_6 (+) a_7, \\ b_6 &= a_4 (+) a_5 (+) a_7, \\ b_7 &= a_4 (+) a_5 (+) a_6, \\ b_8 &= a_9 (+) a_{10} (+) a_{11}, \\ b_9 &= a_8 (+) a_{10} (+) a_{11}, \\ b_{10} &= a_8 (+) a_9 (+) a_{11}, \\ b_{11} &= a_8 (+) a_9 (+) a_{10}, \\ b_{12} &= a_{13} (+) a_{14} (+) a_{15}, \end{aligned}$$

$$b_{13} = a_{12} (+) a_{14} (+) a_{15},$$

$$b_{14} = a_{12} (+) a_{13} (+) a_{15},$$

$$b_{15} = a_{12} (+) a_{13} (+) a_{14}$$

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 P 1 として行われる列拡散演算 (M i x C o l u m n) 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 $a_0 \sim a_{15}$ に基づいて出力 B の 16 個の要素 $b_0 \sim b_{15}$ を算出する。

[0112] 図 26 は、線形変換処理 P 2、すなわち行拡散演算タイプ 1 (M i x R o w 1) 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 P 2 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 $a_0 \sim a_{15}$ からなる状態である。

。

出力 B も n ビットデータの 16 個の要素 $b_0 \sim b_{15}$ からなる状態である。

。

なお、入出力データの各要素 a_i , b_i (ただし $i = 0 \sim 15$) 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 P 2、すなわち行拡散演算タイプ 1 (M i x R o w 1) の具体的な計算処理例を示している。

線形変換処理 P 2 として行われる行拡散演算タイプ 1 (M i x R o w 1) は、以下の式に従った行列演算である。

[0114]

[数7]

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

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

[0115] 出力Bの16個の要素 $b_0 \sim b_{15}$ は、行列 M_0, M_1, M_2, M_3 と、入力Aの16個の要素 $a_0 \sim a_{15}$ 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 P 2 として行われる行拡散演算タイプ 1 (M i x R o w 1) 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 $a_0 \sim a_{15}$ に基づいて出力 B の 16 個の要素 $b_0 \sim b_{15}$ を算出する。

[0117] 図 27 は、線形変換処理 P 3、すなわち行拡散演算タイプ 2 (M i x R o w 2) 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 P 2 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 $a_0 \sim a_{15}$ からなる状態である。

。

出力 B も n ビットデータの 16 個の要素 $b_0 \sim b_{15}$ からなる状態である。

。

なお、入出力データの各要素 a_i , b_i (ただし $i = 0 \sim 15$) 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 P 2、すなわち行拡散演算タイプ 2 (M i x R o w 2) の具体的な計算処理例を示している。

線形変換処理 P 3 として行われる行拡散演算タイプ 2 (M i x R o w 2) は、以下の式に従った行列演算である。

[0119]

[数8]

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

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

[0120] 出力Bの16個の要素 $b_0 \sim b_{15}$ は、行列 M_0, M_1, M_2, M_3 と、入力Aの16個の要素 $a_0 \sim a_{15}$ に基づいて以下の演算によって算出される。

$$\begin{aligned} b_0 &= a_0 (+) a_4 (+) a_8, \\ b_1 &= a_5 (+) a_9 (+) a_{13}, \\ b_2 &= a_2 (+) a_6 (+) a_{14}, \\ b_3 &= a_3 (+) a_{11} (+) a_{15}, \\ b_4 &= a_0 (+) a_4 (+) a_{12}, \\ b_5 &= a_1 (+) a_9 (+) a_{13}, \\ b_6 &= a_2 (+) a_6 (+) a_{10}, \\ b_7 &= a_7 (+) a_{11} (+) a_{15}, \\ b_8 &= a_0 (+) a_8 (+) a_{12}, \\ b_9 &= a_1 (+) a_5 (+) a_{13}, \\ b_{10} &= a_6 (+) a_{10} (+) a_{14}, \\ b_{11} &= a_3 (+) a_7 (+) a_{11}, \\ b_{12} &= a_4 (+) a_8 (+) a_{12}, \\ b_{13} &= a_1 (+) a_5 (+) a_9, \end{aligned}$$

$$b_{14} = a_2 (+) a_{10} (+) a_{14},$$

$$b_{15} = a_3 (+) a_7 (+) a_{15}$$

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 P 3 として行われる行拡散演算タイプ 2 (M i x R o w 2) 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 $a_0 \sim a_{15}$ に基づいて出力 B の 16 個の要素 $b_0 \sim b_{15}$ 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 P 1 ~ P 3 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 $R K_1 \sim R K_7$ 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 P 1 は、行列 M_0 を適用した列拡散演算 (M i x C o l u m n

)である。

線形変換処理 P 2 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 ($M \times Row 1$) である。

線形変換処理 P 3 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 ($M \times Row 2$) 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 P 1、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 64 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、 $4 \times 16 = 64$ 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 $16 \times 6 = 96$ 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 個の要素からなる状態とし、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 $m \times n$ 個の要素からなる状態とした場合、線形変換部は、状態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、状態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 $M_0 \sim M_k$ (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、状態の各列要素単位で行列 $M_0 \sim M_k$ 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、状態の各行要素単位で行列 $M_0 \sim M_k$ 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 $M_0 \sim M_k$ 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 $M_0 \sim M_k$ 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 $M_0 \sim M_k$ 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 $M_0 \sim M_k$ 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 $M_0 \sim M_k$ 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 $M_0 \sim M_k$ 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 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であり、

(a) ステートの各列要素単位で行列 M_0 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 M_0, M_1, M_2, M_3 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 302 a に入力する。鍵変換部 302 a は、鍵 K_1 に対する所定の演算を実行して変換鍵 K_{d1} を生成する。

さらに、鍵変換部 302 a の生成した変換鍵 K_{d1} 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 K_{d1} 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 K_{d1} を鍵変換部 302 b に入力する。鍵変換部 302 b は、変換鍵 K_{d1} 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 K_{d1} 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 302 b 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 K_{d1} を生成し、

変換鍵 K_{d1} 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。

$$K_{d1} = G(K_1)$$

$$K_1 = G^{-1}(K_{d1})$$

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 K_{d1} を変換鍵 K_d 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 状態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 K_d 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 K_d 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

$n_K S F ()$) を適用した演算を実行して中間鍵 S を生成する。

(S 2) 中間鍵 S に対して、変換鍵生成列拡散演算 ($M i \times R o w_K S F ()$) を適用した演算を実行して変換鍵 K d を生成する。

[0152] ステップ S 1 で実行する列拡散演算 ($M i \times C o l u m n$) 、およびステップ S 2 で実行する行拡散演算 ($M i \times R o w$) は、先に図 2 4 ~ 図 2 7 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 (H a d a m a r d) M D S 行列とよばれる行列である。

M D S 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、

$$A A^{-1} = A^{-1} A = E、$$

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 B r a n c h (θ) が $b + 1$ であるような写像 θ は、最適拡散変換 (O p t i m a l D i f f u s i o n M a p p i n g s) と呼ばれ、M D S 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 (H a d a m a r d) M D S 行列 M_D を適用して、図 3 4 に示すステップ S 1 の列拡散演算と、ステップ S 2 の行拡散演算を実行する。

ステップS 1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S 2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S 1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 (H a d a m a r d) M D S行列 M_D を適用した行列演算を実行する。

また、ステップS 2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 (H a d a m a r d) M D S行列 M_D を適用した行列演算を実行する。

[0157] アダマール (H a d a m a r d) M D S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。

$$M_D () : \{0, 1\}^{16} \rightarrow \{0, 1\}^{16}$$

$$\text{Input} : \{x_0, x_1, x_2, x_3\}, x_i \in \{0, 1\}^4$$

$$\text{Output} : \{y_0, y_1, y_2, y_3\}, y_i \in \{0, 1\}^4$$

Operation

$$\cdot y_0 = x_0 (+) 2 (\times) x_1 (+) 4 (\times) x_3 (+) 6 (\times) x_4$$

$$\cdot y_1 = 2 (\times) x_0 (+) x_2 (+) 6 (\times) x_3 (+) 4 (\times) x_4$$

$$\cdot y_2 = 4 (\times) x_0 (+) 6 (\times) x_2 (+) x_3 (+) 2 (\times) x_4$$

$$\cdot y_3 = 6 (\times) x_0 (+) 4 (\times) x_2 (+) 2 (\times) x_3 (+) x_4$$

ただし、

(+) は、排他的論理和演算、

(\times) 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S 1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]

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

[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 : {k₀, k₁, ..., k₁₅}, k_i ∈ {0, 1} ⁴

Output : {s₀, s₁, ..., s₁₅}, s_i ∈ {0, 1} ⁴

Operation

· (s₀, s₁, s₂, s₃) = M_D (k₀, k₁, k₂, k₃)

· (s₄, s₅, s₆, s₇) = M_D (k₄, k₅, k₆, k₇)

· (s₈, s₉, s₁₀, s₁₁) = M_D (k₈, k₉, k₁₀, k₁₁)

· (s₁₂, s₁₃, s₁₄, s₁₅) = M_D (k₁₂, k₁₃, k₁₄, k₁₅)

[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]

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

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

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

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。

$$\text{入力ステート } A = (a_0, a_1, a_2, \dots, a_{15})$$

$$\text{出力ステート } B = (b_0, b_1, b_2, \dots, b_{15})$$

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。

$$b_i = f(a_0, a_1, a_2, \dots, a_{15})$$

[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、

$$\text{ベース鍵 } K = (k_0, k_1, k_2, \dots, k_{15})$$

$$\text{変換鍵 } Kd = (kd_0, kd_1, kd_2, \dots, kd_{15})$$

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、

$$Kd_i = f(k_0, k_1, k_2, \dots, k_{15})$$

ただし、 $i = 0 \sim 15$

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。

$$b_i = f(k_0, k_1, k_2, \dots, k_{15})$$

ただし、 $i = 0 \sim 15$

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 K_{d_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 K_1 に基づいて、 K_{d1} 、 K_{d2} 、 K_{d3} 、 K_{d4} 、 K_{d5} 、 K_{d6} 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 K_{d1} 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 、 K_{d1} 、 K_1 、 K_{d1} 、 $K_1 \cdot \cdot \cdot$ 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 K_{d1} 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 K_{d_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 K_1 に基づいて、 K_{d_1} 、 K_{d_2} 、 K_{d_3} 、 K_{d_4} 、 K_{d_5} 、 K_{d_6} 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 $K_{d_1} \sim K_{d_6}$ 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、 $K_1, K_{d_1}, K_1, K_{d_1}, K_1, \dots$ 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 K_{d_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 K_{d_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 K_{d1} 、 K_{d2} 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 K_{d1} 、

変換鍵 $K d_2$ 、

鍵 K_1 、

鍵 K_2 、

変換鍵 $K d_1$ 、

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、インポリューション性、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 $K d_1$ 、

変換鍵 $K d_2$ 、

排他的論理和演算鍵 $K_1 (+) K_2$ 、

排他的論理和演算変換鍵 $K d_1 (+) K d_2$ 、

図 4 3 (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 4 3 (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 $K d_1$ 、

変換鍵 $K d_2$ 、

排他的論理和演算鍵 $K_1 (+) K_2$ 、

排他的論理和演算変換鍵 $K d_1 (+) K d_2$ 、

排他的論理和演算鍵 $K_1 (+) K_2$ 、

排他的論理和演算変換鍵 $K d_1 (+) K d_2$ 、

排他的論理和演算鍵 $K_1 (+) K_2$ 、

変換鍵 $K d_2$ 、

変換鍵 $K d_1$ 、

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 K_d 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、入力状態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 K_{d_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 K_{d_1} 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 K_{d_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 K_{d_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 $x_0, x_1, x_2, \dots, x_{15}$ 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 $y_0, y_1, y_2, \dots, y_{15}$ 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, \dots, x_{15} (各1bit)

出力： y_0, y_1, \dots, 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 $b_0 \sim b_{15}$ は、ベース鍵Aの各要素 $a_0 \sim a_{15}$ 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 $b_0 \sim b_{15}$ は、ベース鍵Aの各要素 $a_0 \sim a_{15}$ 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 $b_0 \sim b_{15}$ は、ベース鍵Aの各要素 $a_0 \sim a_{15}$ 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。

(S 4 5) ステップ S 4 4 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 1 6 ビットデータ各々に対して、1 6 ビット置換関数 (B p₁₆) 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 b₀ ~ b₁₅ は、ベース鍵 A の各要素 a₀ ~ a₁₅ 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 (D f₄)、(b) 1 6 ビット置換関数 (B p₁₆) 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 5 1 も、

(a) フルディフュージョン 4 ビット関数 (D f₄)

(b) 1 6 ビット置換関数 (B p₁₆)

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 (D f₄)

は、インボリューション性も有する関数である。

[0228] 図 5 1 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。

(S 5 1) ベース鍵 A の 1 6 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 (D f₄) を適用して変換する。

(S 5 2) ステップ S 5 1 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 1 6 ビットデータ各々に対して、1 6 ビット置換関数 (B p₁₆) を適用して変換する。

(S 5 3) ステップ S 5 2 の変換処理によって生成されたデータ (4 × 4 ステート) の 1 6 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 (D f₄) を適用して変換する。

(S 5 4) ステップ S 5 3 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 1 6 ビットデータ各々に対して、1 6 ビット置換関数 (B p₁₆) を適用して変換する。

(S 5 5) ステップ S 5 4 の変換処理によって生成されたデータ (4 × 4 ステート) の 1 6 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 (D f₄) 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 b₀ ~ b₁₅ は、ベース鍵 A の各要素 a₀ ~ a₁₅ 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 (D f₄)、(b) 1 6 ビット置換関数 (B p₁₆) 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 4 7 ~ 図 5 1 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 4 2 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 5 2 に示す暗号処理装置は、鍵スケジュール部 3 8 0 と、暗号処理部 3 8 5 を有する。鍵スケジュール部 3 8 0 の鍵レジスタ 3 8 1 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 K₁ と鍵 K₂ 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_1 、 G_2 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 G_1 、 G_2 の組み合わせとしては、例えば以下のような設定が可能である。

(a) G_1 、 G_2 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) G_1 、 G_2 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) G_1 、 G_2 ともフルディフュージョン性を有し、 G_1 と G_2 が逆関数の設定、すなわち、 $G_2 = G_1^{-1}$ の関係にある。

鍵変換関数 G_1 、 G_2 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 $X + \Delta A$ に変換されたものとする。

この場合、値 $X + \Delta A$ 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 $X + \Delta A$ に対応する出力値が $Y + \Delta B$ 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、 $X + \Delta A$ 、

暗号処理装置の入出力値の対応が、 Y と、 $Y + \Delta B$ 、

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、インボリュージ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 E 431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ジョン性を有し、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 K_r 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 K_r 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。

$$B = A (+) K_r (+) CON$$

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($con_0 \sim con_{15}$)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]

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

[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 P 1 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 (M i x C o l u m n) 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 4 3 7 は、上記の行列 M₀ を適用した列拡散演算 (M i x C o l u m n) 、すなわち、

MC [M₀]

を実行する。

[0260] このとき、定数 CON の条件は、この線形変換部 4 3 7 において適用する線形変換行列 M₀ 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 5 8 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 M₀ を適用した列拡散演算 (M

$i \times \text{Column}) : Mc [M_0]$ 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、インポ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 $K d_1 (+) K d_2$ 、
 排他的論理和演算鍵 $K_1 (+) K_2$ 、
 変換鍵 $K d_2$ 、
 変換鍵 $K d_1$ 、
 鍵 K_2 、
 鍵 K_1 、

[0265] なお、図 4 3 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 5 9 に示す例では、線形変換部 4 5 2 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 5 9 に示す構成は、図 4 3 に示すラウンド演算部 R_6 を、線形変換部 4 5 2 と、線形変換部 4 5 2 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 5 9 に示す構成において、定数 CON は、暗号処理部 4 5 3 (E^{-1}) の排他的論理和部 4 6 1、排他的論理和部 4 6 3、排他的論理和部 4 6 5、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ステータの定数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 (Mix Column)、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 P_1 , P_2 を個別に入力して暗号処理結果 C_1 , C_2 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 P_1 , P_2 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 E_{431} 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 E_{431} 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} , 4 3 3 に対して、同様に線形変換部 4 3 2 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 $S_2 (= S_1 (+) \Delta X)$ を入力して、出力 $T (+) \Delta Y$ を得る。

[0278] この2つの関数、すなわち、変換関数 E 4 3 1 と、変換関数 E 4 3 1 の逆関数である変換関数 E^{-1} , 4 3 3 に対して、図 6 1 に示すように、逆方向に差分 ΔX を有する入力値 S_1 , S_2 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 6 2 は、前述の図 5 9 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 6 2 に示す暗号処理装置の暗号処理部 4 5 1 (E) の線形変換部 4 5 2 側から入力値 S_1 を入力し、暗号処理部 4 5 1 (E) を適用した暗号処理を実行する。

一方、暗号処理部 4 5 1 (E) の逆関数である暗号処理部 4 5 3 (E^{-1}) の線形変換部 4 5 2 側から入力値 S_1 に対して差分 ΔX を設定した入力値 S_2 を入力し、暗号処理部 4 5 3 (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}

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 $K d_1$ 、
 変換鍵 $K d_2$ 、
 排他的論理和演算鍵 $K_1 (+) K_2$ 、
 排他的論理和演算変換鍵 $K d_1 (+) K d_2$ 、
 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 $K d_1 (+) K d_2$ 、
 排他的論理和演算鍵 $K_1 (+) K_2$ 、
 変換鍵 $K d_2$ 、
 変換鍵 $K d_1$ 、
 鍵 K_2 、
 鍵 K_1 、

[0285] なお、図 4 3 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 6 3 に示す例では、線形変換部 4 5 2 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 6 3 に示す構成は、図 4 3 に示すラウンド演算部 R_6 を、
 線形変換部 4 5 2 と、線形変換部 4 5 2 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481、482、483は、いずれも前述の行列M₀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 $4 \times 16 = 64$ 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（ i_n ）と出力値（ $o_u t$ ）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65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。

[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]

$$\begin{aligned} a_{out} &= c_{in} \oplus \sim((d_{in} \oplus \sim(a_{in} | b_{in})) | (a_{in} \oplus \sim(b_{in} | c_{in}))) \\ b_{out} &= d_{in} \oplus \sim(a_{in} | b_{in}) \\ c_{out} &= a_{in} \oplus \sim(b_{in} | c_{in}) \\ d_{out} &= b_{in} \oplus \sim(b_{out} | a_{out}) \\ &= b_{in} \oplus \sim((d_{in} \oplus \sim(a_{in} | b_{in})) | (c_{in} \oplus \sim((d_{in} \oplus \sim(a_{in} | b_{in})) | (a_{in} \oplus \sim(b_{in} | c_{in})))))) \end{aligned}$$

[0305] なお、上記式において、 $\sim(x | y)$ 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⁻¹413、

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⁻¹413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 6 5、図 6 6 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 6 7 (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 $X = (x_0, x_1, x_2, x_3)$ 、出力 4 ビット $Y = (y_0, y_1, y_2, y_3)$ 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、

$$y_i \neq x_i$$

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 6 7 (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 P_4 の条件を、式で示すと以下のように示すことができる。

$$(a) P_4 (P_4 (X)) = X$$

$$(b) y_i \neq x_i \quad \text{ただし } i = 0, 1, 2, 3、$$

上記 (a) は置換関数 P_4 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 6 5、図 6 6 を参照して構成と異なる構成を持つ例について、図 6 8 以下を参照して説明する。

[0314] 図 6 8 に示す S ボックスは、

非線形変換層 1、5 3 1、

線形変換層 5 3 2、

非線形変換層 2, 5 3 3、

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, 5 3 1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 5 3 2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, 5 3 3は、非線形変換層 1, 5 3 1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 (i_n) と出力値 ($o_u t$) 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]

$$\begin{aligned}
 a_{out} &= d_{in} \oplus \sim(b_{out} | d_{out}) = d_{in} \oplus \sim((c_{in} \oplus \sim(a_{in} \& b_{in})) | (a_{in} \oplus \sim(b_{in} | d_{in}))) \\
 b_{out} &= c_{in} \oplus \sim(a_{in} \& b_{in}) \\
 c_{out} &= b_{in} \oplus (a_{out} \& b_{out}) \\
 &= b_{in} \oplus ((d_{in} \oplus \sim((c_{in} \oplus \sim(a_{in} \& b_{in})) | (a_{in} \oplus \sim(b_{in} | d_{in})))) \& (c_{in} \oplus \sim(a_{in} \& b_{in}))) \\
 d_{out} &= a_{in} \oplus \sim(b_{in} | d_{in})
 \end{aligned}$$

[0320] なお、上記式において、

$\sim(x | y)$ 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。

$\sim(x \& y)$ 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 E_{411} 、

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 E_{411} 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 5 4 2、

非線形変換層 2, 5 4 3、

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, 5 4 1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 5 4 2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, 5 4 3は、非線形変換層 1, 5 4 1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 (i_n) と出力値 ($o_u t$) 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70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。

[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、

$a_{i n}, b_{i n}, c_{i n}, d_{i n}$ とし、

Sボックス (S-box) からの4ビット出力を、

$a_{o u t}, b_{o u t}, c_{o u t}, d_{o u t}$ 、

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]

$$\begin{aligned}
 a_{out} &= c_{in} \oplus ((d_{in} \oplus (a_{in} | b_{in})) | (a_{in} \oplus (b_{in} | c_{in}))) \\
 b_{out} &= d_{in} \oplus (a_{in} | b_{in}) \\
 c_{out} &= a_{in} \oplus (b_{in} | c_{in}) \\
 d_{out} &= b_{in} \oplus (b_{out} | a_{out}) \\
 &= b_{in} \oplus ((d_{in} \oplus (a_{in} | b_{in})) | (c_{in} \oplus ((d_{in} \oplus (a_{in} | b_{in})) | (a_{in} \oplus (b_{in} | c_{in}))))))
 \end{aligned}$$

[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E_{4 1 1}、

線形変換部4 1 2、

変換関数E⁻¹_{4 1 3}、

これらの構成を持つ暗号処理部内の変換関数E_{4 1 1}と、変換関数E⁻¹_{4 1 3}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 RK_n 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 RK_n 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 P 1、

線形変換部 P 2、

線形変換部 P 3、

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 P 1 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 ($M_i \times C$ o l u m) である。

すなわち、線形変換処理部 P 1 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 P 2 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 P 2 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 ($M_i \times R$ o w) を実行する。

[0343] 線形変換処理部 P 3 も、線形変換処理部 P 2 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 P 3 は、線形変換処理 P

2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 ($M_i \times Row$) 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 ($M_i \times Row$) 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 P 3、
ラウンド鍵 R K₅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 P 1、
ラウンド鍵 R K₆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 R K₇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 P 2、
ラウンド鍵 R K₇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 R K₈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 P 1、
非線形変換部 S、
ラウンド鍵 R K₉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 P 3、
非線形変換部 S、
ラウンド鍵 R K₁₀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 P 1、
非線形変換部 S、
ラウンド鍵 R K₁₁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 P 2、
非線形変換部 S、
ラウンド鍵 R K₁₂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 P 1、
非線形変換部 S、
ラウンド鍵 R K₁₃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 5 9 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 6 4 ~ 図 6 6 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 7 2 0 のラウンド鍵供給部 7 2 1 は、鍵レジスタ 7 2 2 と鍵変換部 7 2 3 を有する。このラウンド鍵供給部 7 2 1 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 3 2 ~ 図 4 3 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 7 5 0 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 6 4 ビットのベース鍵 K_1 と K_2 との連結データである 1 2 8 ビット鍵データである。

鍵変換部 7 2 3 は、ベース鍵 K_1 に基づく変換鍵 $K d_1$ 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 $K d_2$ 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。

$$K d_1 = G (K_1)$$

$$K_1 = G^{-1} (K d_1)$$

なお、

$$G = G^{-1}$$

が成立する。

すなわち、鍵変換部 7 2 3 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 3 3 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 3 4 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、以下の効果をもたらされる。

[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、インポ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 K_{d1} 、

変換鍵 K_{d2} 、

排他的論理和演算鍵 $K_1 (+) K_2$ 、

排他的論理和演算変換鍵 $K_{d1} (+) K_{d2}$ 、

排他的論理和演算鍵 $K_1 (+) K_2$ 、

排他的論理和演算変換鍵 $K_{d1} (+) K_{d2}$ 、

排他的論理和演算鍵 $K_1 (+) K_2$ 、

変換鍵 K_{d2} 、

変換鍵 K_{d1} 、

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 $K_1 \sim K_{13}$ 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 $RK_8 = K_{d1} (+) K_{d2} (+) CON$

ラウンド鍵 $RK_{10} = K_{d2} (+) CON$

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 $RK_1 \sim RK_{13}$ 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处理がなされる。

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处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ジョン性を有する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^{-1} 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ジ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、暗号処理に必要な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へ出力する鍵スケジュール部を有し、

前記鍵スケジュール部は、

秘密鍵 K を格納した鍵レジスタと、

前記秘密鍵 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 K_d を生成する鍵変換部を有し、

前記秘密鍵 K 、および前記変換鍵 K_d を前記暗号処理部へ出力するラウンド鍵、またはラウンド鍵生成用データとする構成であり、

前記変換関数 G は、逆関数 G^{-1} が関数 G と同じ関数となるインボリューション性を有する関数である暗号処理装置。

[0383] (2) 前記秘密鍵 K 、および前記変換鍵 K_d は、いずれも各要素が1ビット

ト以上の $m \times n$ 個の要素からなる状態であり、前記変換関数 G は、前記秘密鍵 K の状態要素の全てが、前記変換鍵 K_d の状態要素全てに影響を及ぼすフルディフュージョン性を有する関数である前記 (1) に記載の暗号処理装置。

[0384] (3) 前記変換関数 G は、前記秘密鍵 K の状態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、前記秘密鍵 K の状態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を実行して前記変換鍵 K_d を生成する前記 (2) に記載の暗号処理装置。

[0385] (4) 前記秘密鍵 K 、および前記変換鍵 K_d は、いずれも各要素が 4 ビットの 4×4 個の要素からなる状態であり、前記変換関数 G は、前記秘密鍵 K の状態要素の全てが、前記変換鍵 K_d の状態要素全てに影響を及ぼすフルディフュージョン性を有する関数である前記 (1) ~ (3) いずれかに記載の暗号処理装置。

[0386] (5) 前記変換関数 G は、前記秘密鍵 K の状態の各列要素単位でアダマール MDS (Hadamard MDS) 行列を適用して線形変換を行う列拡散演算と、前記秘密鍵 K の状態の各行要素単位でアダマール MDS (Hadamard MDS) 行列を適用して線形変換を行う行拡散演算を実行して前記変換鍵 K_d を生成する前記 (4) に記載の暗号処理装置。

[0387] (6) 前記鍵スケジュール部は、前記秘密鍵 K は、分割鍵 K_1 、 K_2 の連結データであり、前記鍵変換部は、前記分割鍵 K_1 、 K_2 の各々に対する前記変換関数 G を適用した変換処理により変換鍵 K_{d1} 、 K_{d2} を生成し、前記分割鍵 K_1 、 K_2 、および前記変換鍵 K_{d1} 、 K_{d2} を前記暗号処理部へ出力するラウンド鍵、またはラウンド鍵生成用データとする前記 (1) ~ (5) いずれかに記載の暗号処理装置。

[0388] (7) 前記鍵スケジュール部は、
(a) 前記分割鍵 K_1 、 K_2 、
(b) 前記変換鍵 K_{d1} 、 K_{d2} 、
(c) 前記分割鍵 K_1 と前記分割鍵 K_2 との排他的論理和演算によって生

成した鍵、

(d) 前記変換鍵 $K_d 1$ と前記変換鍵 $K_d 2$ との排他的論理和演算によって生成した鍵、

上記(a)～(d)の6種類の鍵を前記暗号処理部へ出力するラウンド鍵、またはラウンド鍵生成用データとする前記(6)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へ出力する前記(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、

前記鍵スケジュール部は、

秘密鍵 K を格納した鍵レジスタと、

前記秘密鍵 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 K_d を生成する鍵変換部を有し、

前記秘密鍵 K 、および前記変換鍵 K_d を前記暗号処理部に出力するラウンド鍵、またはラウンド鍵生成用データとする構成であり、

前記秘密鍵 K 、および前記変換鍵 K_d は、いずれも各要素が1ビット以上の $m \times n$ 個の要素からなる状態であり、

前記変換関数 G は、前記秘密鍵 K の状態要素の全てが、前記変換鍵 K_d の状態要素全てに影響を及ぼすフルディフュージョン性を有する関数である暗号処理装置。

[0395] (14) 前記変換関数 G は、フルディフュージョン変換関数とビット置換関数の組み合わせである前記(13)に記載の暗号処理装置。

[0396] (15) 前記秘密鍵 K 、および前記変換鍵 K_d は、いずれも各要素が4ビットの 4×4 個の要素からなる状態であり、前記変換関数 G は、フルディフュージョン4ビット変換関数と16ビット置換関数の組み合わせである前記(13)または(14)に記載の暗号処理装置。

[0397] (16) 前記鍵スケジュール部は、前記秘密鍵 K は、分割鍵 K_1 、 K_2 の連結データであり、前記鍵変換部は、前記分割鍵 K_1 に対する変換関数 G_1 を適用した変換処理により変換鍵 K_{d1} を生成し、前記分割鍵 K_2 に対する変換関数 G_2 を適用した変換処理により変換鍵 K_{d2} を生成する構成であり、前記変換関数 G_1 、 G_2 は、

(a) G_1 、 G_2 ともフルディフュージョン性を有するがインボリユーション性を有さない変換関数、

(b) G_1 、 G_2 ともフルディフュージョン性と、インボリユーション性を有する変換関数、

(c) G_1 , G_2 ともフルディフュージョン性を有し、 G_1 と G_2 が逆関数の設定、すなわち、 $G_2 = G_1^{-1}$ の関係にある変換関数、

上記(a)～(c)のいずれかの組み合わせである前記(13)～(15)いずれか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へ出力する鍵スケジュール部を有し、

前記鍵スケジュール部は、

鍵レジスタに格納した秘密鍵 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 K_d を生成し、

前記秘密鍵 K 、および前記変換鍵 K_d を前記暗号処理部へ出力するラウンド鍵、またはラウンド鍵生成用データとする処理を行ない、

前記変換関数 G は、逆関数 G^{-1} が関数 G と同じ関数となるインボリュージョン性を有する関数である暗号処理方法。

[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へ出力する鍵スケジュール部を有し、

前記鍵スケジュール部は、

鍵レジスタに格納した秘密鍵 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 K_d を生成し、

前記秘密鍵 K 、および前記変換鍵 K_d を前記暗号処理部へ出力するラウンド鍵、またはラウンド鍵生成用データとする処理を行ない、

前記秘密鍵 K 、および前記変換鍵 K_d は、いずれも各要素が1ビット以上の $m \times n$ 個の要素からなるステートであり、

前記変換関数 G は、前記秘密鍵 K のステート要素の全てが、前記変換鍵 K_d のステート要素全てに影響を及ぼすフルディフュージョン性を有する関数である暗号処理方法。

[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へ出力する鍵スケジュール部を有し、

前記プログラムは、前記鍵スケジュール部に、

鍵レジスタに格納した秘密鍵 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 K_d を生成する処理と、

前記秘密鍵 K 、および前記変換鍵 K_d を前記暗号処理部へ出力するラウンド鍵、またはラウンド鍵生成用データとする処理を実行させ、

前記変換関数 G は、逆関数 G^{-1} が関数 G と同じ関数となるインボリュージョン性を有する関数であるプログラム。

[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へ出力する鍵スケジュール部を有し、

前記プログラムは、前記鍵スケジュール部に、

鍵レジスタに格納した秘密鍵 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 K_d を生成する処理と、

前記秘密鍵 K 、および前記変換鍵 K_d を前記暗号処理部に出力するラウンド鍵、またはラウンド鍵生成用データとする処理を実行させ、

前記秘密鍵 K 、および前記変換鍵 K_d は、いずれも各要素が 1 ビット以上の $m \times n$ 個の要素からなる状態であり、

前記変換関数 G は、前記秘密鍵 K の状態要素の全てが、前記変換鍵 K_d の状態要素全てに影響を及ぼすフルディフュージョン性を有する関数であるプログラム。

[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を向上させた安全性の高い暗号処理構成が実現される。

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、鍵スケジュール部は、秘密鍵Kを格納した鍵レジスタと、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K dを生成する鍵変換部を有し、秘密鍵K、および変換鍵K dを暗号処理部に出力するラウンド鍵、またはラウンド鍵生成用データとする。変換関数Gとして、逆関数 G^{-1} が関数Gと同じ関数となるインボリューション性と、フルディフュージョン性を有する関数を適用する。

本構成により拡散特性が向上し、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] 1 0 0 暗号処理装置
- 1 1 0 鍵スケジュール部
- 1 2 0 暗号処理部
- 1 2 1 排他的論理和部
- 1 2 2 非線形変換部
- 1 2 3 線形変換部
- 2 0 1 線形変換部 P 1
- 2 0 2 線形変換部 P 2
- 2 0 3 線形変換部 P 3
- 3 0 0 鍵スケジュール部
- 3 0 1 鍵供給部（鍵レジスタ）
- 3 0 2 鍵変換部
- 3 2 0 暗号処理部
- 3 2 1 ~ 3 2 7 排他的論理和部
- 3 3 1, 3 3 3 排他的論理和部
- 3 3 2 非線形／線形変換部（S & P）
- 3 5 0 暗号処理部

- 3 5 1 排他的論理和部
- 3 5 2 非線形／線形変換部 (S & P)
- 3 6 0 鍵スケジュール部
- 3 6 1, 3 6 2 鍵レジスタ
- 3 6 3 鍵変換部
- 3 7 1 鍵レジスタ
- 3 7 2 鍵変換部
- 3 8 1 鍵レジスタ
- 3 9 1 鍵レジスタ
- 3 9 2, 3 9 4 排他的論理和部
- 3 9 3 鍵変換部
- 4 0 1 ラウンド演算実行部
- 4 0 2 定数入力部
- 4 1 1 変換関数 E
- 4 1 2 線形変換部
- 4 1 3 変換関数 E^{-1}
- 4 3 1 変換関数 E
- 4 3 2 線形変換部
- 4 3 3 変換関数 E^{-1}
- 4 3 5 定数入力部
- 4 3 6 排他的論理和部
- 4 3 7 線形変換部
- 4 5 1 変換関数 E
- 4 5 2 線形変換部
- 4 5 3 変換関数 E^{-1}
- 5 2 1 非線形変換層 1
- 5 2 2 線形変換層
- 5 2 3 非線形変換層

- 7 0 0 暗号処理装置
- 7 2 0 鍵スケジュール部
- 7 2 1 ラウンド鍵供給部
- 7 2 2 鍵レジスタ
- 7 2 3 鍵変換部
- 7 2 5 定数供給部
- 7 5 0 暗号処理部
- 7 5 1 排他的論理和部
- 7 5 2 非線形変換部
- 7 5 3 線形変換部
- 8 0 0 ICモジュール
- 8 0 1 CPU (Central processing Unit)
- 8 0 2 メモリ
- 8 0 3 暗号処理部
- 8 0 4 乱数生成部
- 8 0 5 送受信部
- 9 0 0 スマートフォン
- 9 0 1 プロセッサ
- 9 0 2 メモリ
- 9 0 3 ストレージ
- 9 0 4 外部接続インタフェース
- 9 0 6 カメラ
- 9 0 7 センサ
- 9 0 8 マイクロフォン
- 9 0 9 入力デバイス
- 9 1 0 表示デバイス
- 9 1 1 スピーカ
- 9 1 3 無線通信インタフェース

- 9 1 4 アンテナスイッチ
- 9 1 5 アンテナ
- 9 1 7 バス
- 9 1 8 バッテリー
- 9 1 9 補助コントローラ

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、
- 前記鍵スケジュール部は、
- 秘密鍵Kを格納した鍵レジスタと、
- 前記秘密鍵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K dを生成する鍵変換部を有し、
- 前記秘密鍵K、および前記変換鍵K dを前記暗号処理部に出力するラウンド鍵、またはラウンド鍵生成用データとする構成であり、
- 前記変換関数Gは、逆関数 G^{-1} が関数Gと同じ関数となるインボリューション性を有する関数である暗号処理装置。
- [請求項2] 前記秘密鍵K、および前記変換鍵K dは、いずれも各要素が1ビット以上の $m \times n$ 個の要素からなる状態であり、
- 前記変換関数Gは、前記秘密鍵Kの状態要素の全てが、前記変換鍵K dの状態要素全てに影響を及ぼすフルディフュージョン性を有する関数である請求項1に記載の暗号処理装置。
- [請求項3] 前記変換関数Gは、
- 前記秘密鍵Kの状態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、
- 前記秘密鍵Kの状態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を実行して前記変換鍵K dを生成する請求項2に記載の暗号処理装置。
- [請求項4] 前記秘密鍵K、および前記変換鍵K dは、いずれも各要素が4ビットの 4×4 個の要素からなる状態であり、
- 前記変換関数Gは、前記秘密鍵Kの状態要素の全てが、前記変換鍵K dの状態要素全てに影響を及ぼすフルディフュージョン性

を有する関数である請求項1に記載の暗号処理装置。

[請求項5]

前記変換関数Gは、

前記秘密鍵Kのステートの各列要素単位でアダマールMDS (Hadamard MDS) 行列を適用して線形変換を行う列拡散演算と、

前記秘密鍵Kのステートの各行要素単位でアダマールMDS (Hadamard MDS) 行列を適用して線形変換を行う行拡散演算を実行して前記変換鍵 K_d を生成する請求項4に記載の暗号処理装置。

[請求項6]

前記鍵スケジュール部は、

前記秘密鍵Kは、分割鍵 K_1 、 K_2 の連結データであり、

前記鍵変換部は、前記分割鍵 K_1 、 K_2 の各々に対する前記変換関数Gを適用した変換処理により変換鍵 K_{d1} 、 K_{d2} を生成し、

前記分割鍵 K_1 、 K_2 、および前記変換鍵 K_{d1} 、 K_{d2} を前記暗号処理部へ出力するラウンド鍵、またはラウンド鍵生成用データとする請求項1に記載の暗号処理装置。

[請求項7]

前記鍵スケジュール部は、

(a) 前記分割鍵 K_1 、 K_2 、

(b) 前記変換鍵 K_{d1} 、 K_{d2} 、

(c) 前記分割鍵 K_1 と前記分割鍵 K_2 との排他的論理和演算によって生成した鍵、

(d) 前記変換鍵 K_{d1} と前記変換鍵 K_{d2} との排他的論理和演算によって生成した鍵、

上記(a)～(d)の6種類の鍵を前記暗号処理部へ出力するラウンド鍵、またはラウンド鍵生成用データとする請求項6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へ出力する請求項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]

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へ出力する鍵スケジュール部を有し、
前記鍵スケジュール部は、
秘密鍵Kを格納した鍵レジスタと、
前記秘密鍵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

K d を生成する鍵変換部を有し、

前記秘密鍵K、および前記変換鍵K d を前記暗号処理部に出力するラウンド鍵、またはラウンド鍵生成用データとする構成であり、

前記秘密鍵K、および前記変換鍵K d は、いずれも各要素が1ビット以上の $m \times n$ 個の要素からなるステートであり、

前記変換関数Gは、前記秘密鍵Kのステート要素の全てが、前記変換鍵K d のステート要素全てに影響を及ぼすフルディフュージョン性を有する関数である暗号処理装置。

[請求項14]

前記変換関数Gは、

フルディフュージョン変換関数とビット置換関数の組み合わせである請求項13に記載の暗号処理装置。

[請求項15]

前記秘密鍵K、および前記変換鍵K d は、いずれも各要素が4ビットの 4×4 個の要素からなるステートであり、

前記変換関数Gは、

フルディフュージョン4ビット変換関数と16ビット置換関数の組み合わせである請求項13に記載の暗号処理装置。

[請求項16]

前記鍵スケジュール部は、

前記秘密鍵Kは、分割鍵K 1、K 2の連結データであり、

前記鍵変換部は、前記分割鍵K 1に対する変換関数G 1を適用した変換処理により変換鍵K d 1を生成し、前記分割鍵K 2に対する変換関数G 2を適用した変換処理により変換鍵K d 2を生成する構成であり、

前記変換関数G 1、G 2は、

(a) G 1、G 2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変換関数、

(b) G 1、G 2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変換関数、

(c) G 1、G 2ともフルディフュージョン性を有し、G 1とG 2

が逆関数の設定、すなわち、 $G^2 = G^{-1}$ の関係にある変換関数、
上記 (a) ~ (c) のいずれかの組み合わせである請求項 13 に記載の暗号処理装置。

[請求項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へ出力する鍵スケジュール部を有し、
前記鍵スケジュール部は、
鍵レジスタに格納した秘密鍵 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 K_d を生成し、
前記秘密鍵 K 、および前記変換鍵 K_d を前記暗号処理部へ出力するラウンド鍵、またはラウンド鍵生成用データとする処理を行ない、
前記変換関数 G は、逆関数 G^{-1} が関数 G と同じ関数となるインボリューション性を有する関数である暗号処理方法。

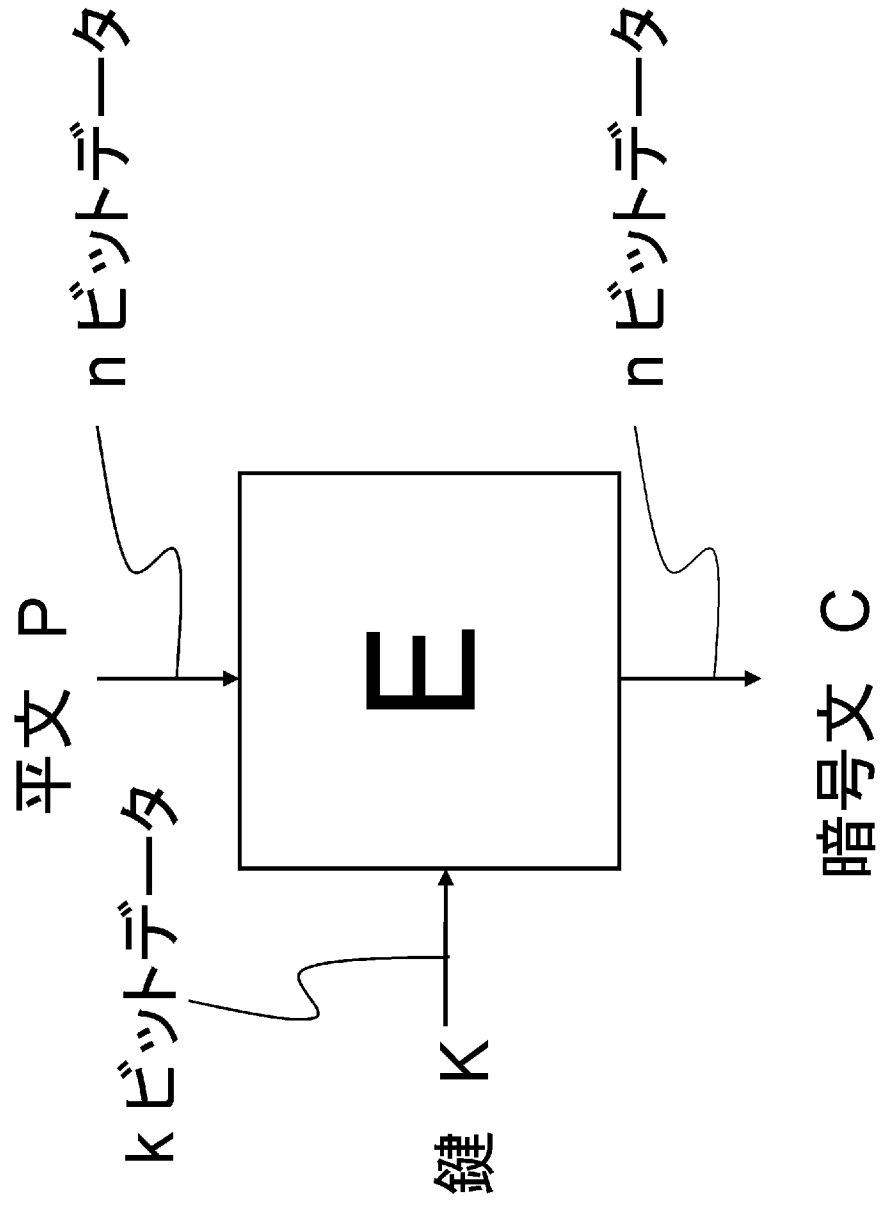
[請求項18]

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へ出力する鍵スケジュール部を有し、
前記鍵スケジュール部は、
鍵レジスタに格納した秘密鍵 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 K_d を生成し、
前記秘密鍵 K 、および前記変換鍵 K_d を前記暗号処理部へ出力するラウンド鍵、またはラウンド鍵生成用データとする処理を行ない、
前記秘密鍵 K 、および前記変換鍵 K_d は、いずれも各要素が 1 ビット

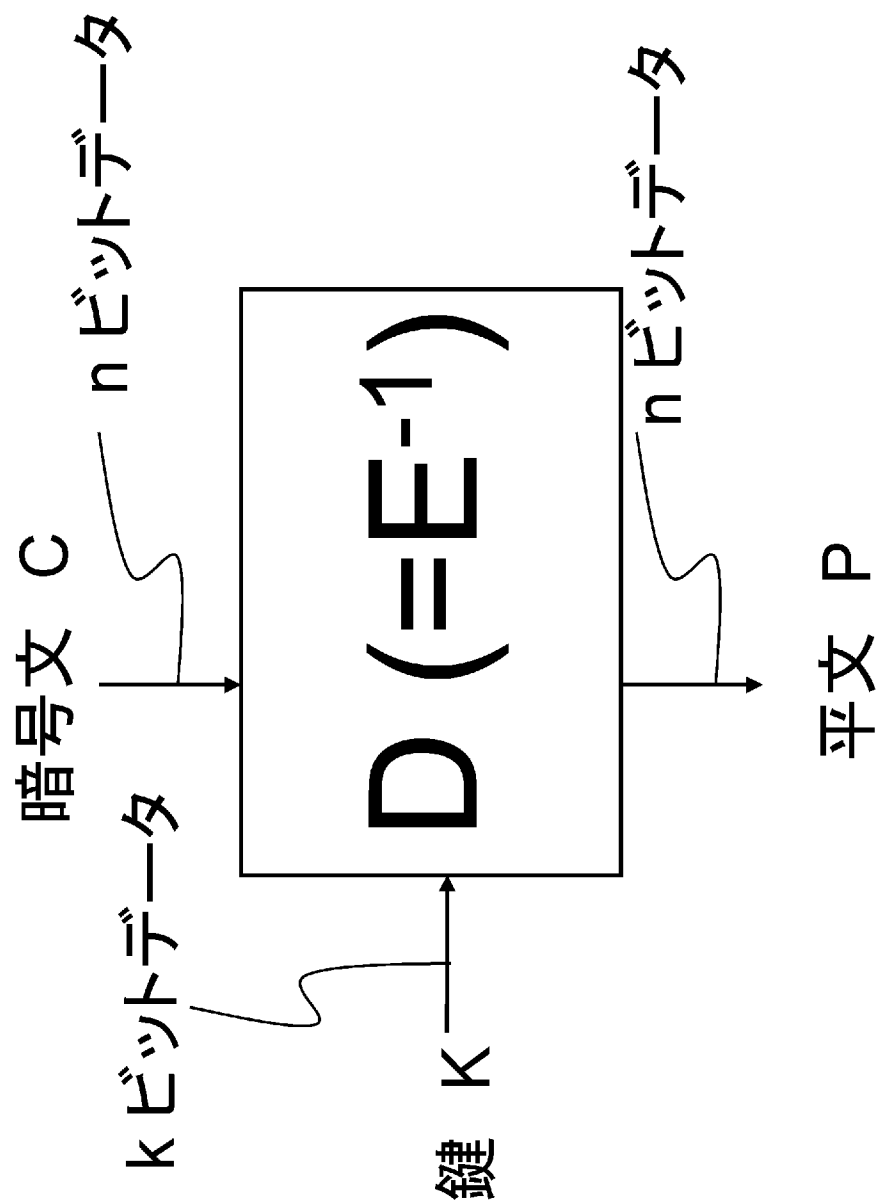
ト以上の $m \times n$ 個の要素からなるステートであり、

前記変換関数 G は、前記秘密鍵 K のステート要素の全てが、前記変換鍵 K_d のステート要素全てに影響を及ぼすフルディフュージョン性を有する関数である暗号処理方法。

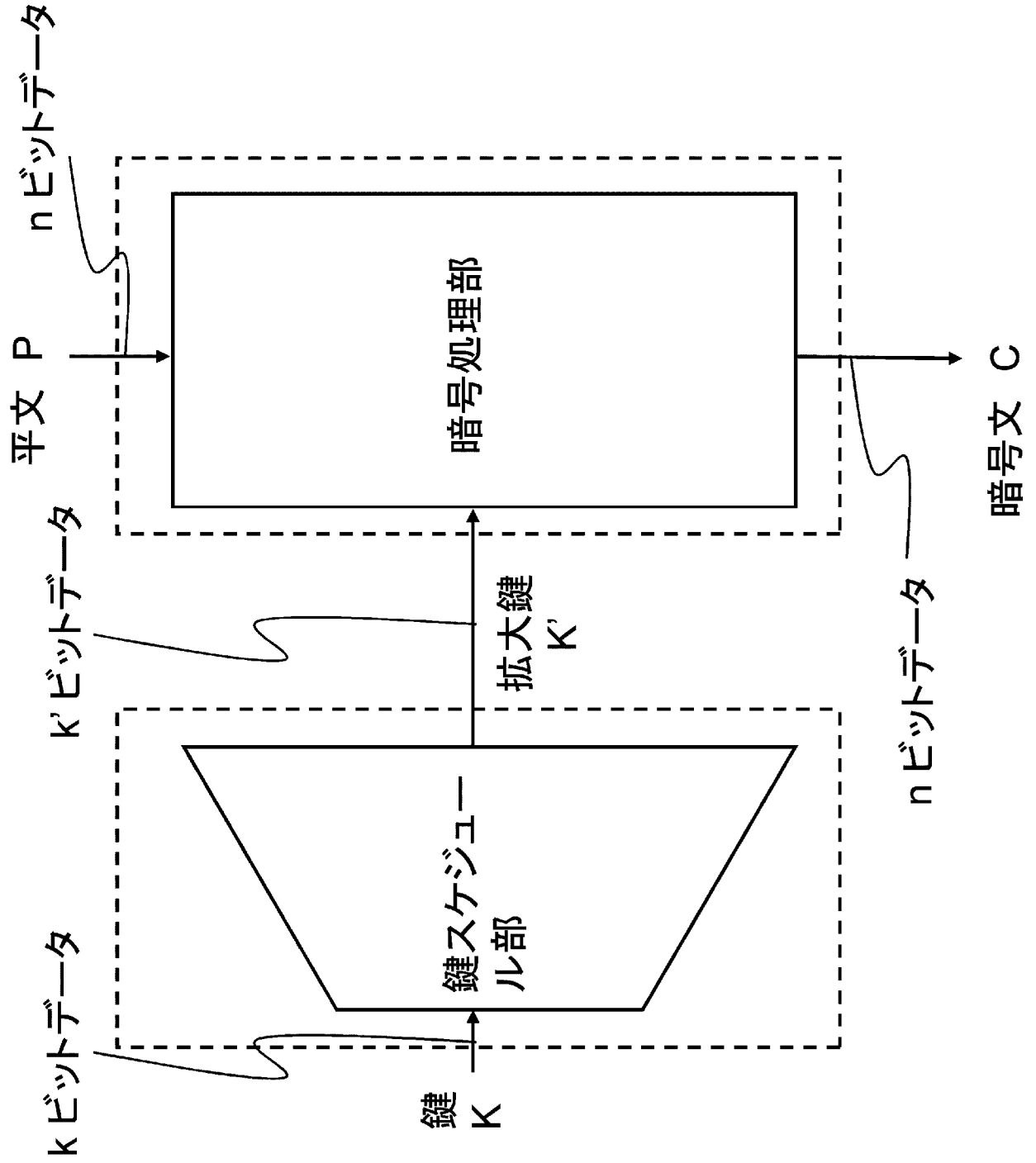
[図1]



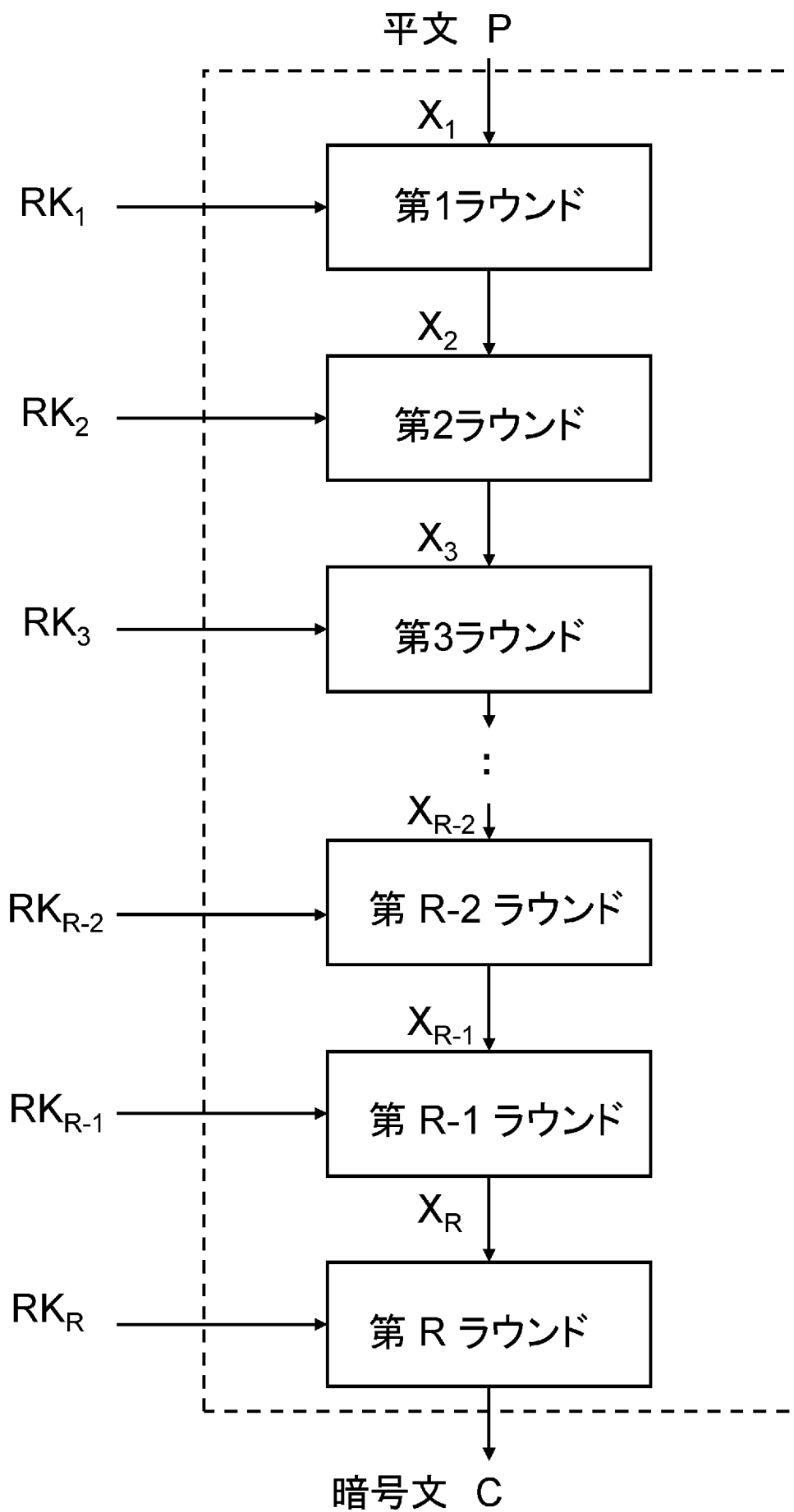
[図2]



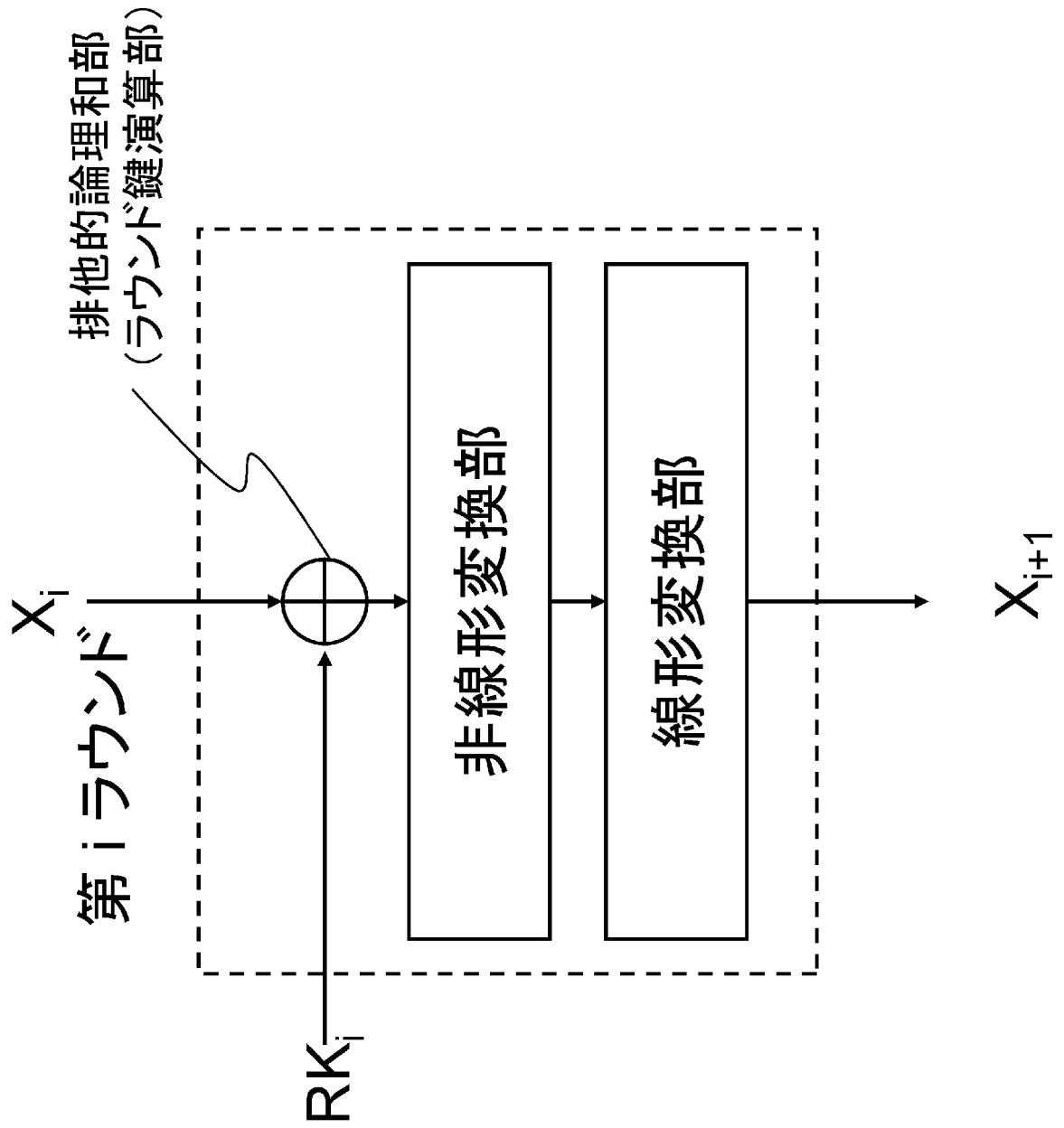
[図3]



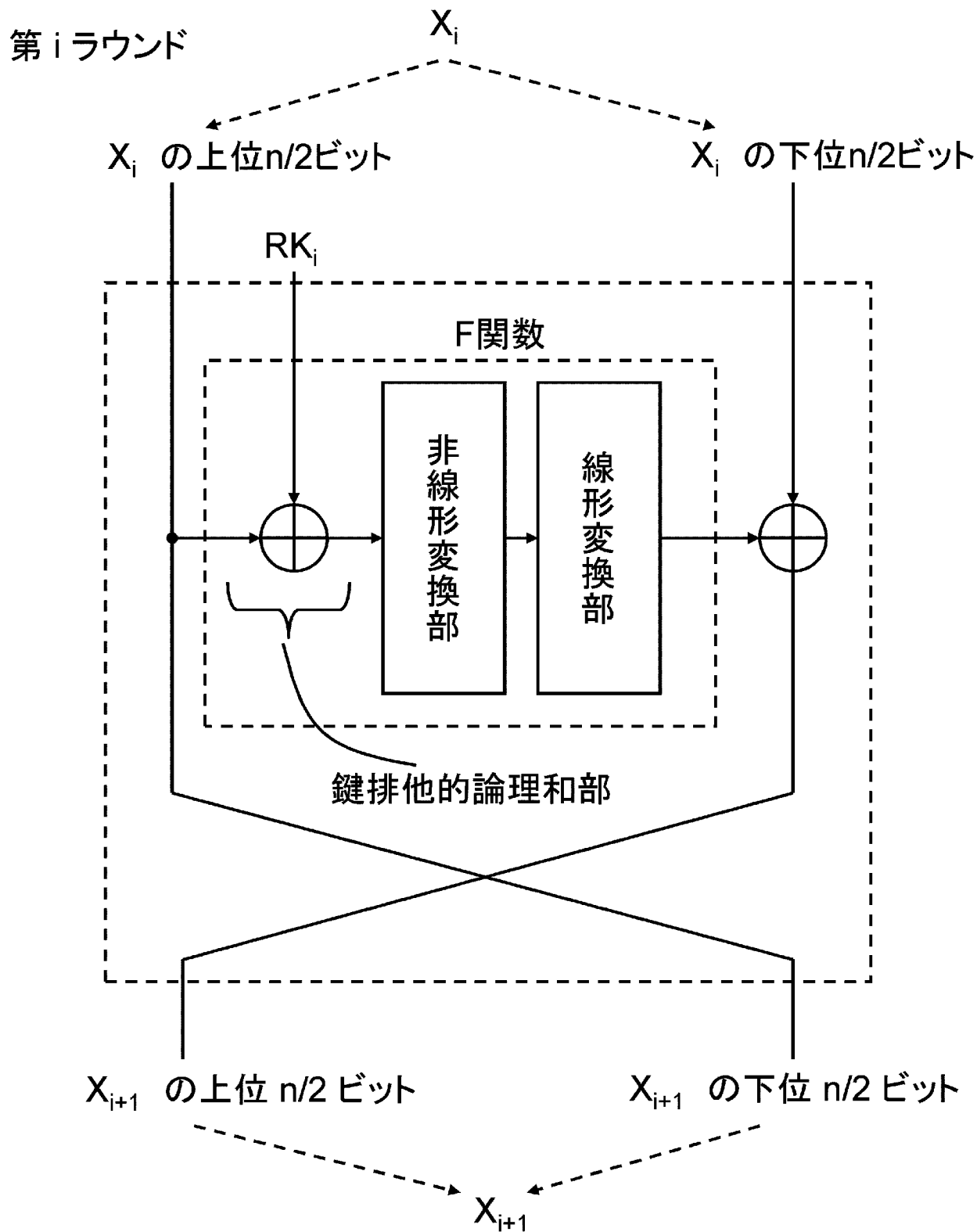
[図4]



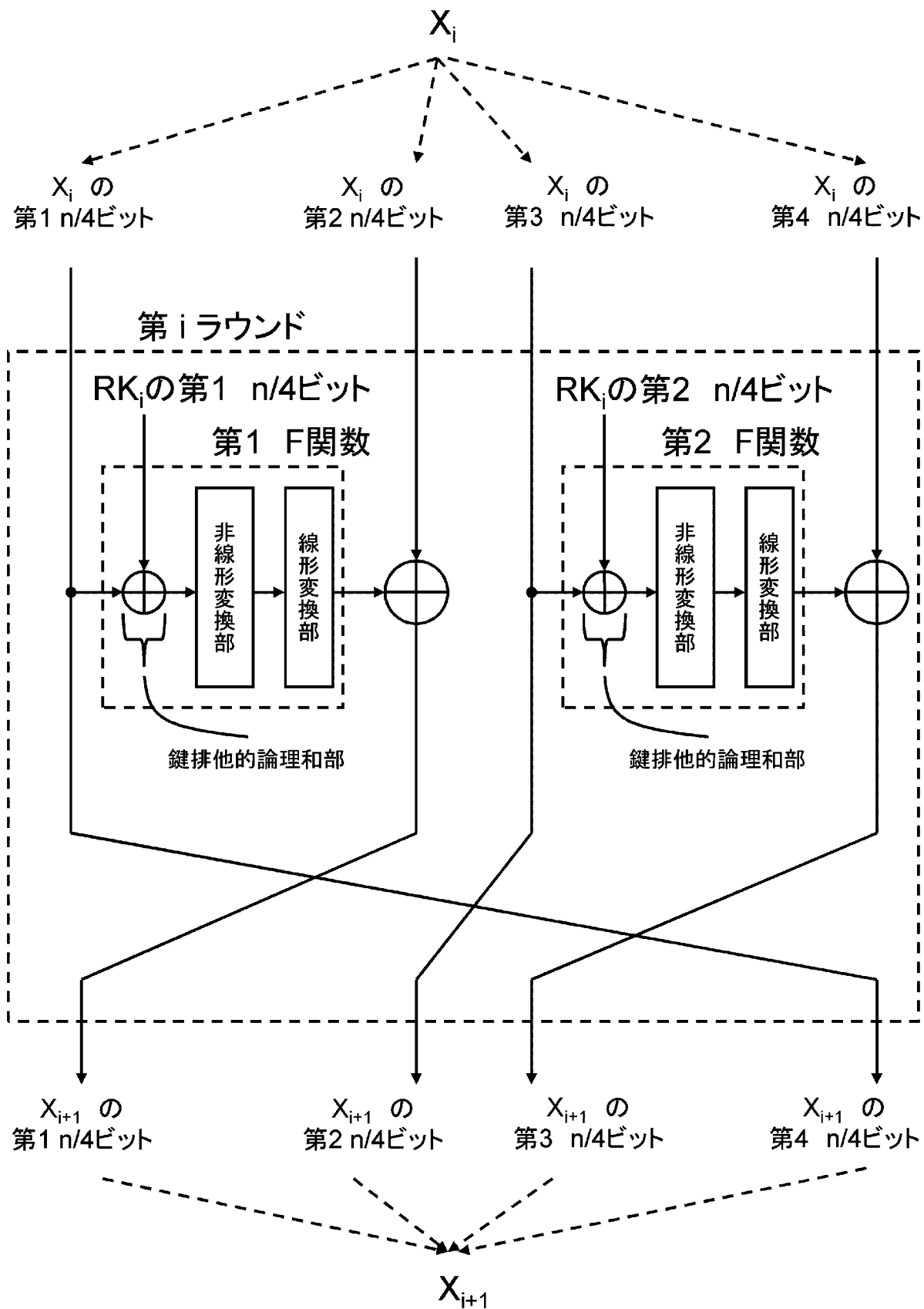
[図5]



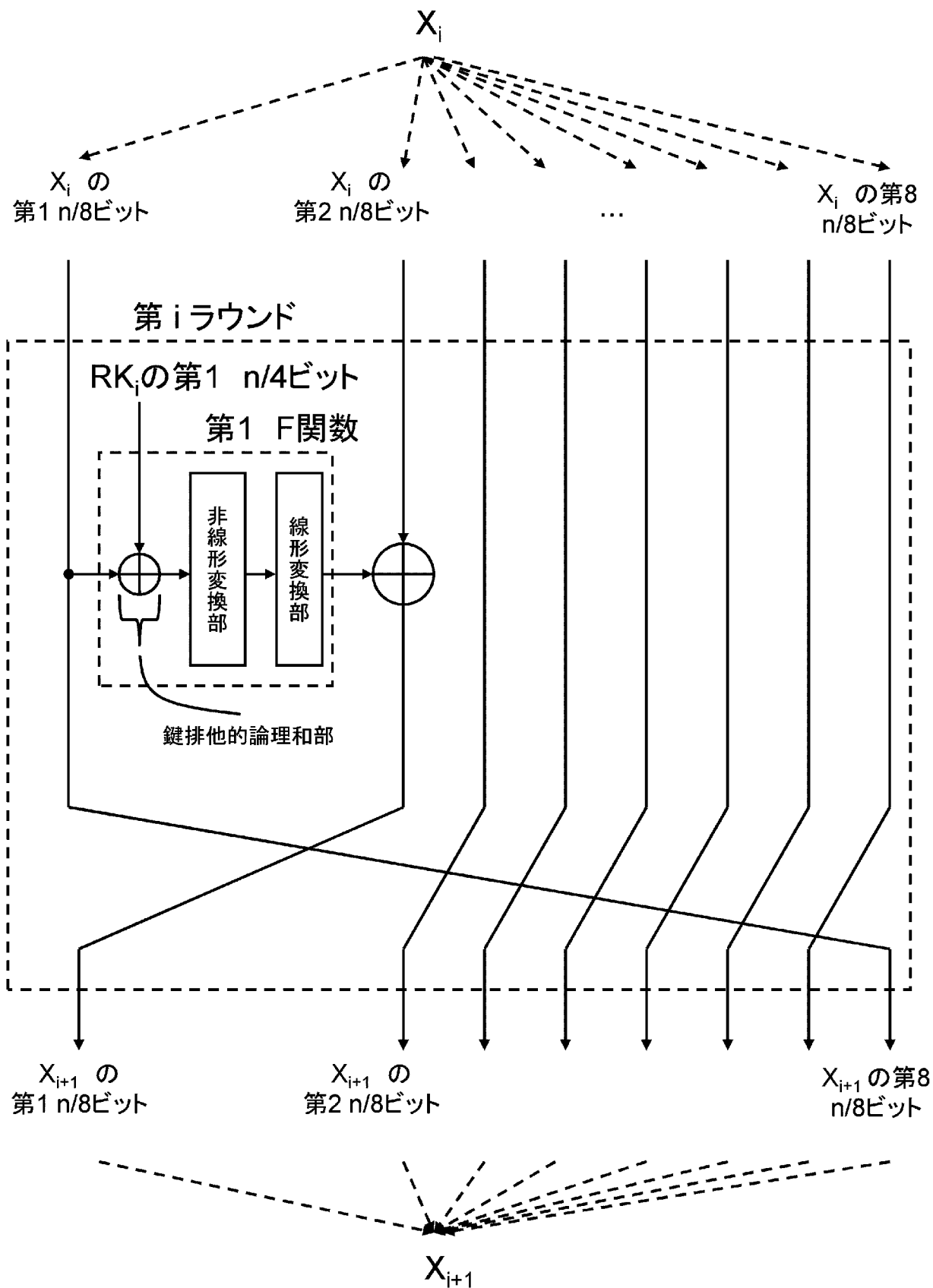
[図6]



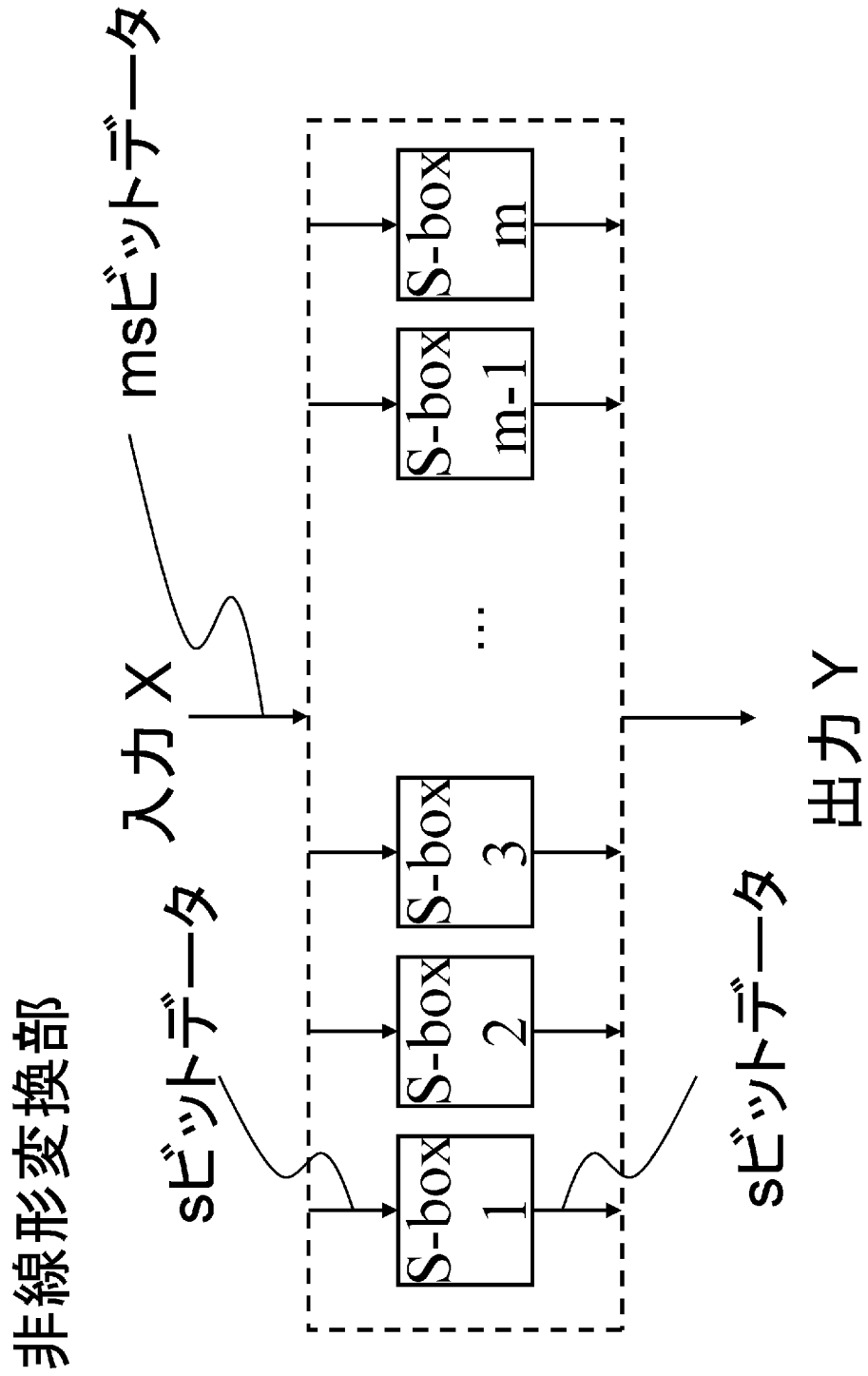
[図7]



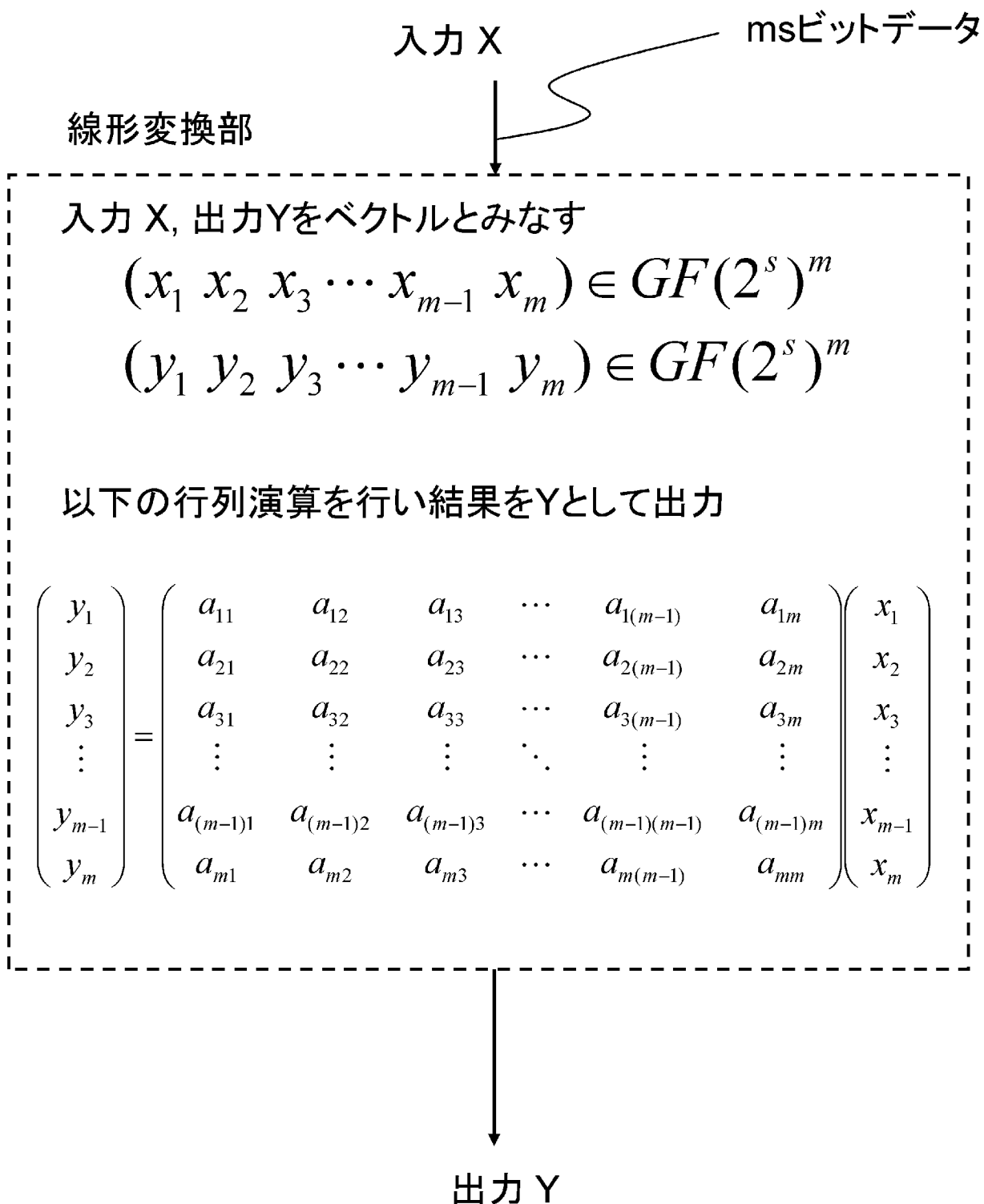
[図8]



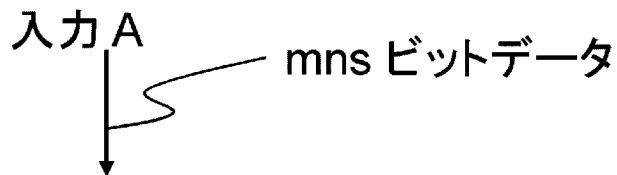
[図9]



[図10]



[図11]

入力 A

 mns ビットデータ

入力 A, 出力 B をベクトルとみなすと以下のように表現できる

$$(a_0 \ a_1 \ a_2 \ \cdots \ a_{mn-2} \ a_{mn-1}) \in GF(2^s)^{mn}$$

$$(b_0 \ b_1 \ b_2 \ \cdots \ b_{mn-2} \ b_{mn-1}) \in GF(2^s)^{mn}$$

ただし, a_0, b_0 はそれぞれ MSB,
 a_{mn-1}, b_{mn-1} はそれぞれ LSB を表す

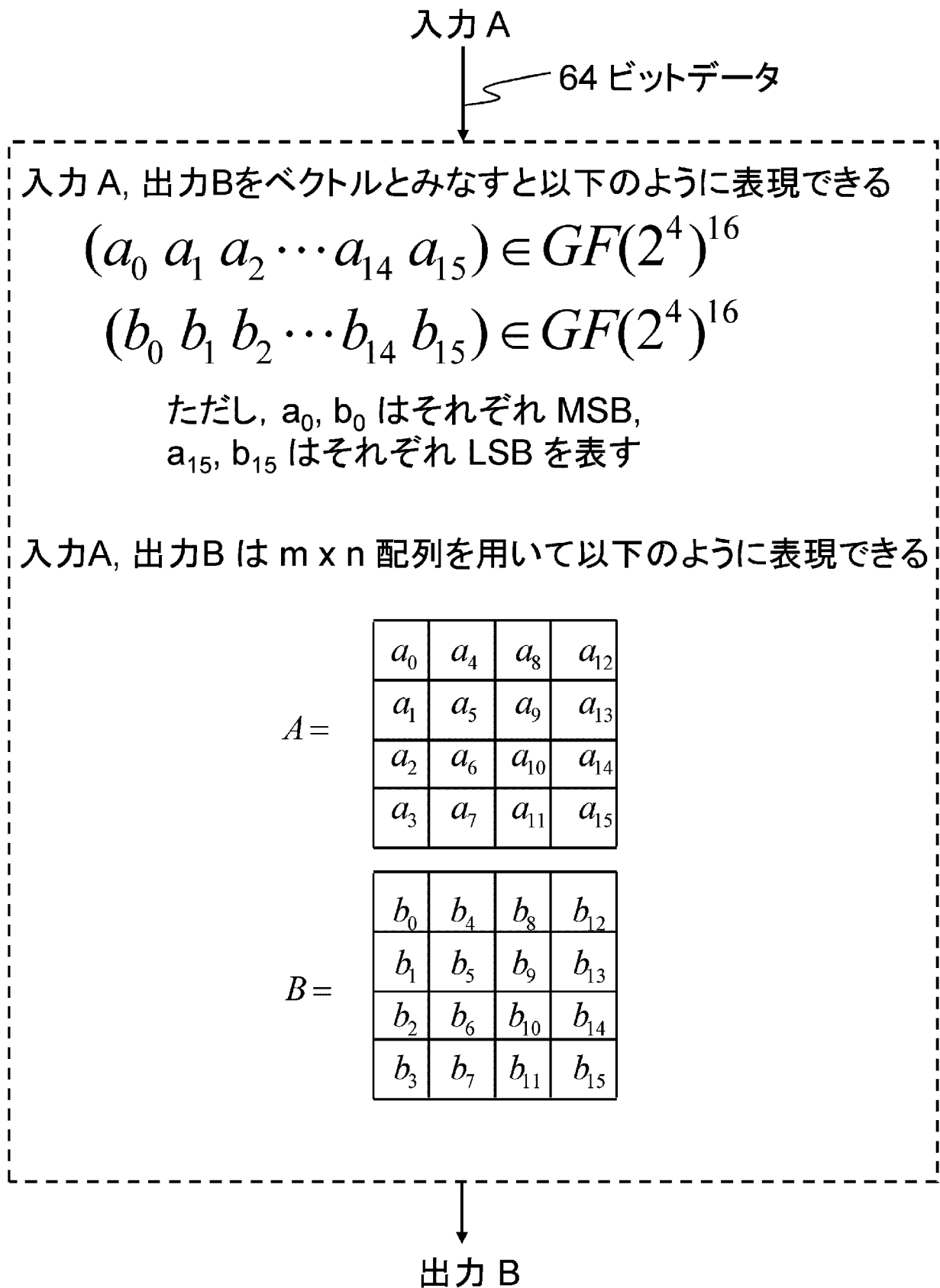
入力 A, 出力 B は $m \times n$ 配列を用いて以下のように表現できる

$$A = \begin{array}{|c|c|c|c|c|} \hline a_0 & a_m & a_{2m} & \cdots & a_{(n-1)m} \\ \hline a_1 & a_{m+1} & a_{2m+1} & \cdots & a_{(n-1)m+1} \\ \hline a_2 & a_{m+2} & a_{2m+2} & \cdots & a_{(n-1)m+2} \\ \hline \vdots & \vdots & \vdots & \ddots & \vdots \\ \hline a_{m-1} & a_{2m-1} & a_{3m-1} & \cdots & a_{nm-1} \\ \hline \end{array}$$

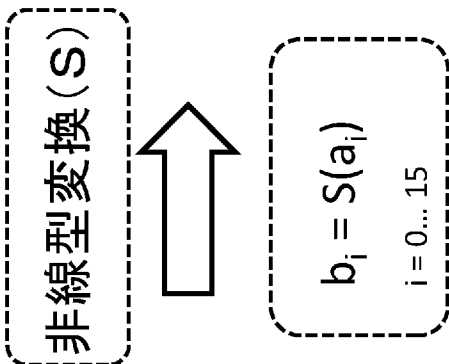
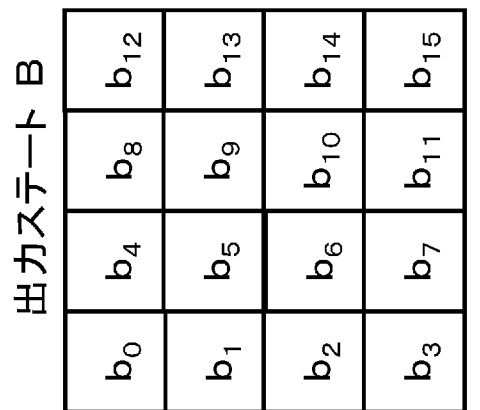
$$B = \begin{array}{|c|c|c|c|c|} \hline b_0 & b_m & b_{2m} & \cdots & b_{(n-1)m} \\ \hline b_1 & b_{m+1} & b_{2m+1} & \cdots & b_{(n-1)m+1} \\ \hline b_2 & b_{m+2} & b_{2m+2} & \cdots & b_{(n-1)m+2} \\ \hline \vdots & \vdots & \vdots & \ddots & \vdots \\ \hline b_{m-1} & b_{2m-1} & b_{3m-1} & \cdots & b_{nm-1} \\ \hline \end{array}$$

出力 B

[図12]



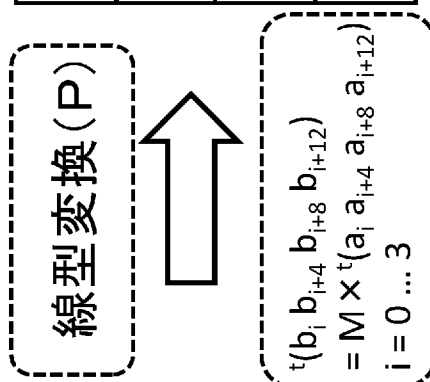
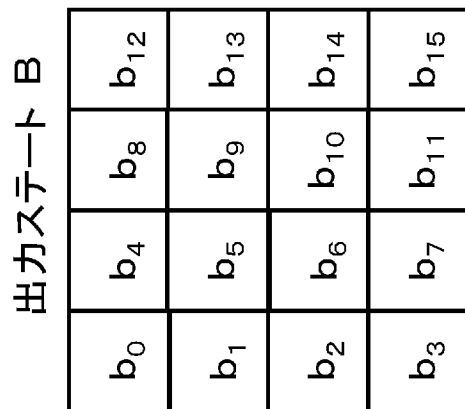
[図13]



入力ステート A

a_0	a_4	a_8	a_{12}
a_1	a_5	a_9	a_{13}
a_2	a_6	a_{10}	a_{14}
a_3	a_7	a_{11}	a_{15}

(1)



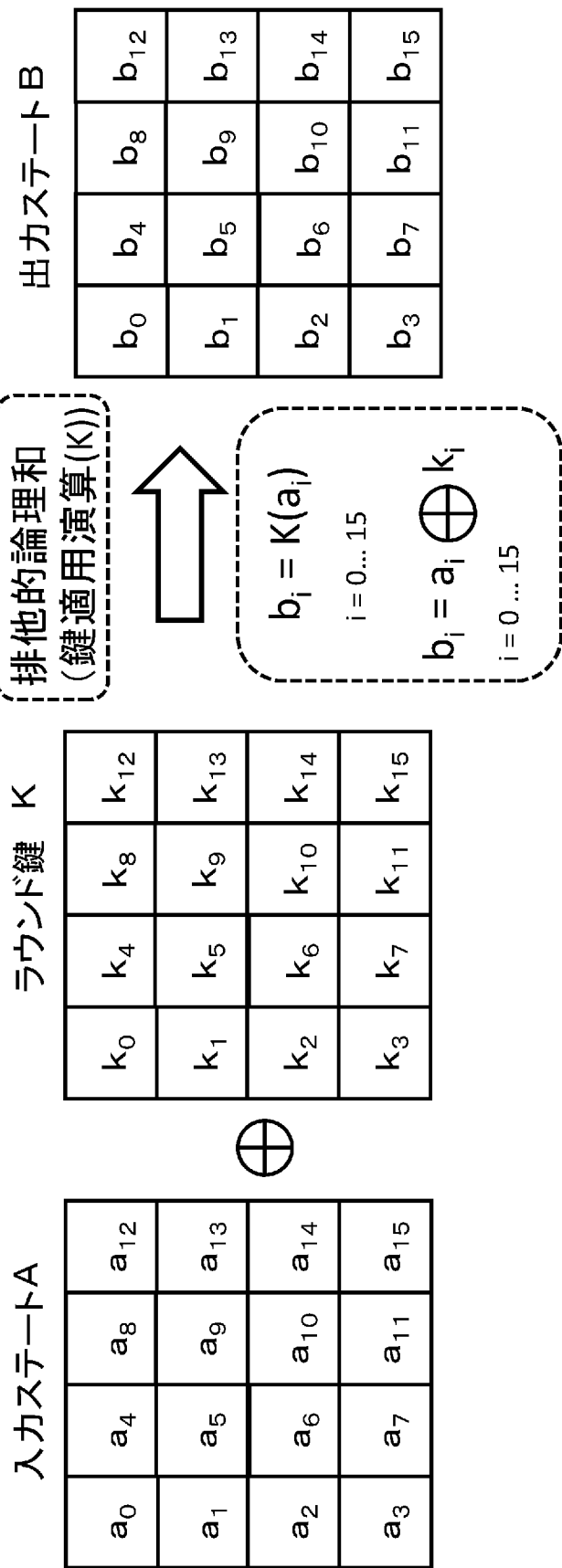
入力ステート A

a_0	a_4	a_8	a_{12}
a_1	a_5	a_9	a_{13}
a_2	a_6	a_{10}	a_{14}
a_3	a_7	a_{11}	a_{15}

(2)

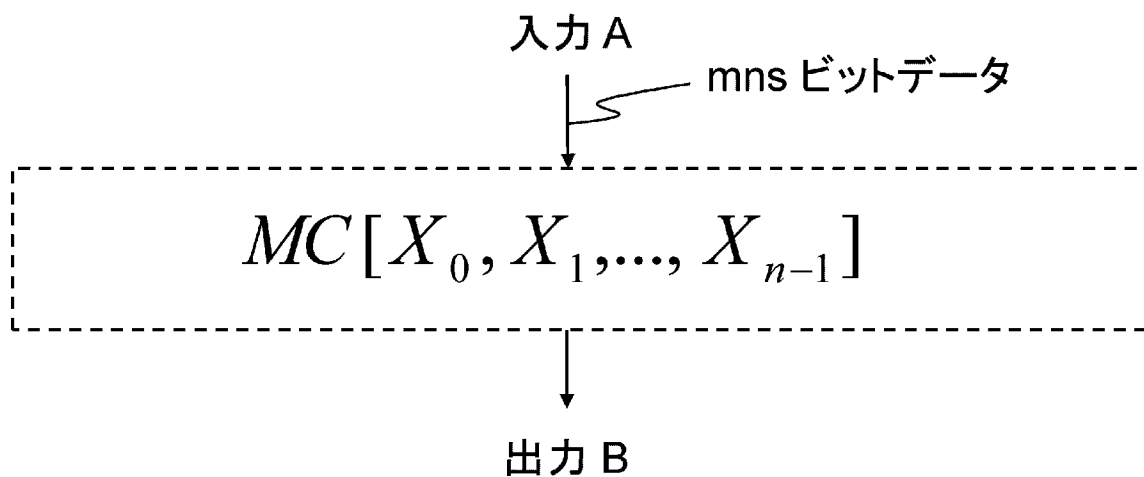
線形変換として行拡散演算を用いた場合

[図14]



(3)

[図15]



$$A =$$

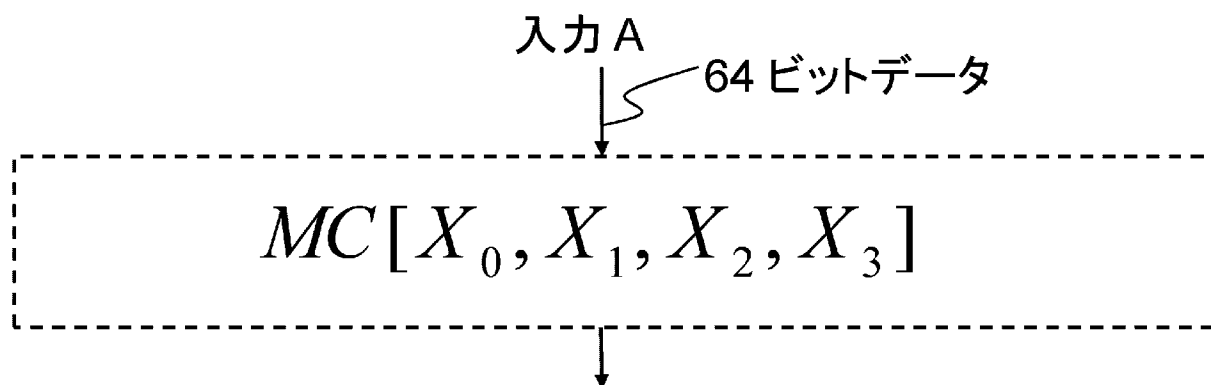
a_0	a_m	a_{2m}	\dots	$a_{(n-1)m}$
a_1	a_{m+1}	a_{2m+1}	\dots	$a_{(n-1)m+1}$
a_2	a_{m+2}	a_{2m+2}	\dots	$a_{(n-1)m+2}$
\vdots	\vdots	\vdots	\ddots	\vdots
a_{m-1}	a_{2m-1}	a_{3m-1}	\dots	a_{nm-1}

$$B =$$

b_0	b_m	b_{2m}	\dots	$b_{(n-1)m}$
b_1	b_{m+1}	b_{2m+1}	\dots	$b_{(n-1)m+1}$
b_2	b_{m+2}	b_{2m+2}	\dots	$b_{(n-1)m+2}$
\vdots	\vdots	\vdots	\ddots	\vdots
b_{m-1}	b_{2m-1}	b_{3m-1}	\dots	b_{nm-1}

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

[図16]



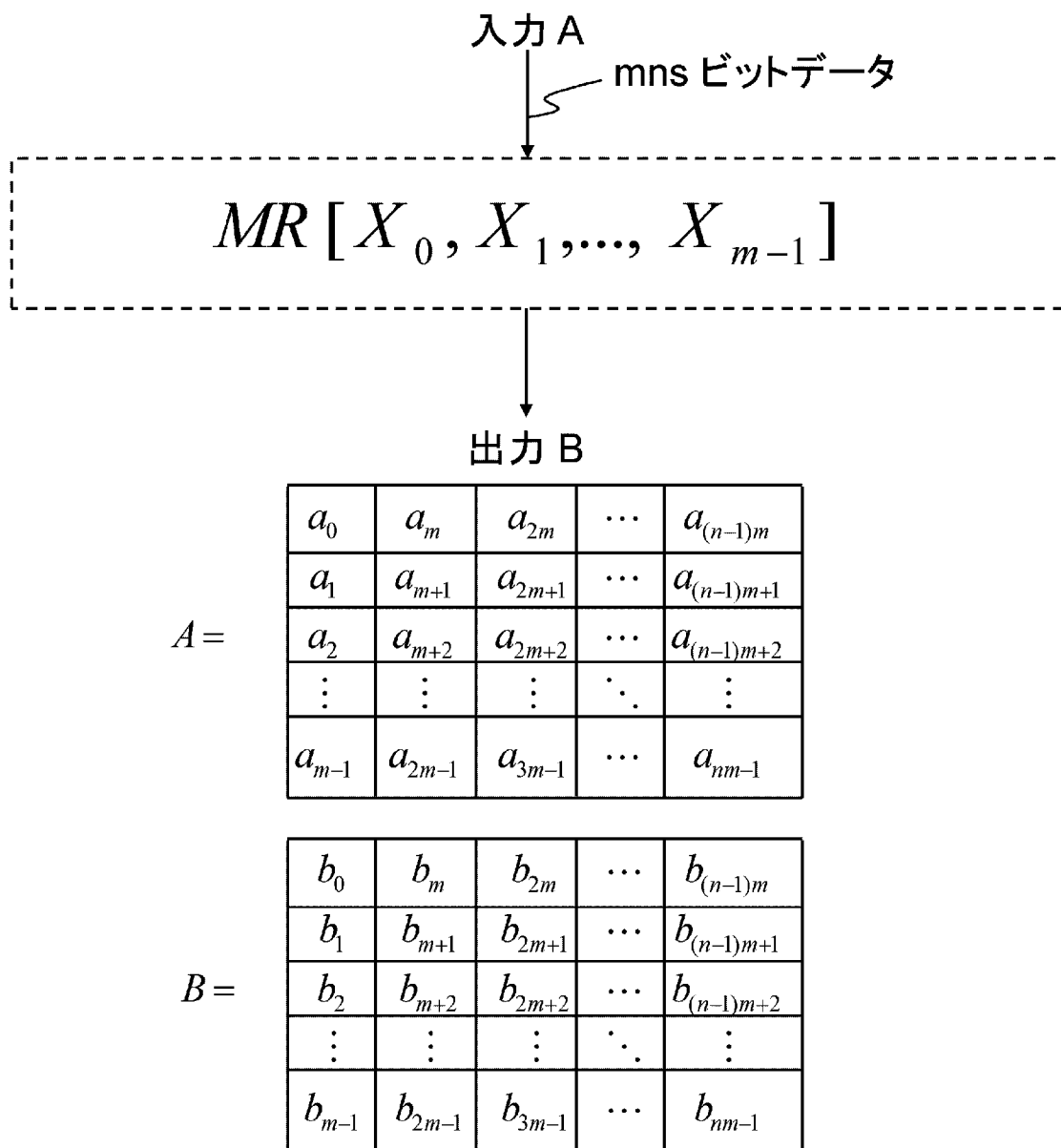
出力 B

$$A = \begin{array}{|c|c|c|c|} \hline a_0 & a_4 & a_8 & a_{12} \\ \hline a_1 & a_5 & a_9 & a_{13} \\ \hline a_2 & a_6 & a_{10} & a_{14} \\ \hline a_3 & a_7 & a_{11} & a_{15} \\ \hline \end{array}$$

$$B = \begin{array}{|c|c|c|c|} \hline b_0 & b_4 & b_8 & b_{12} \\ \hline b_1 & b_5 & b_9 & b_{13} \\ \hline b_2 & b_6 & b_{10} & b_{14} \\ \hline b_3 & b_7 & b_{11} & b_{15} \\ \hline \end{array}$$

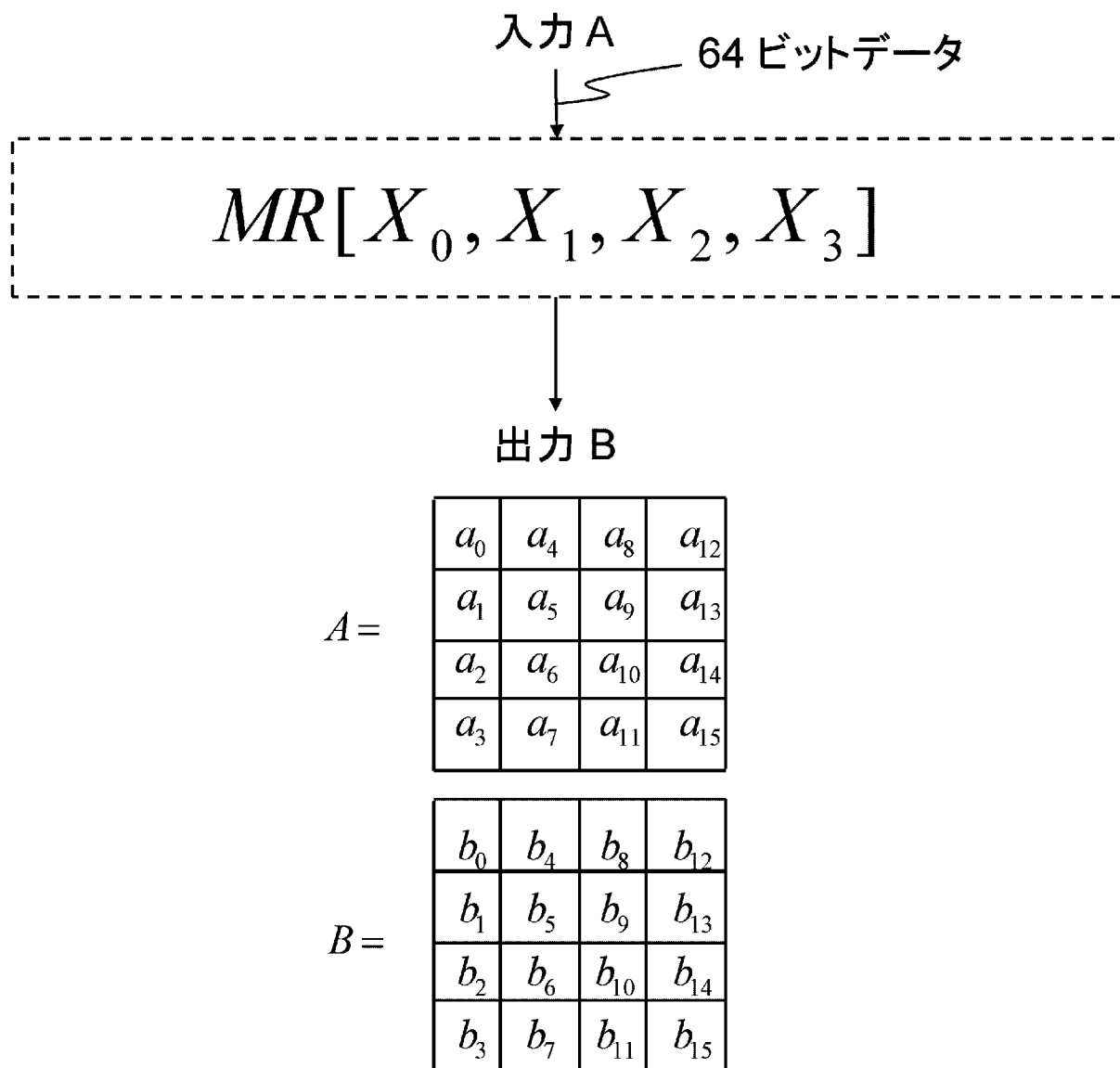
$$\begin{pmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \end{pmatrix} = X_0 \cdot \begin{pmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \end{pmatrix}, \begin{pmatrix} b_4 \\ b_5 \\ b_6 \\ b_7 \end{pmatrix} = X_1 \cdot \begin{pmatrix} a_4 \\ a_5 \\ a_6 \\ a_7 \end{pmatrix}, \begin{pmatrix} b_8 \\ b_9 \\ b_{10} \\ b_{11} \end{pmatrix}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

[図17]



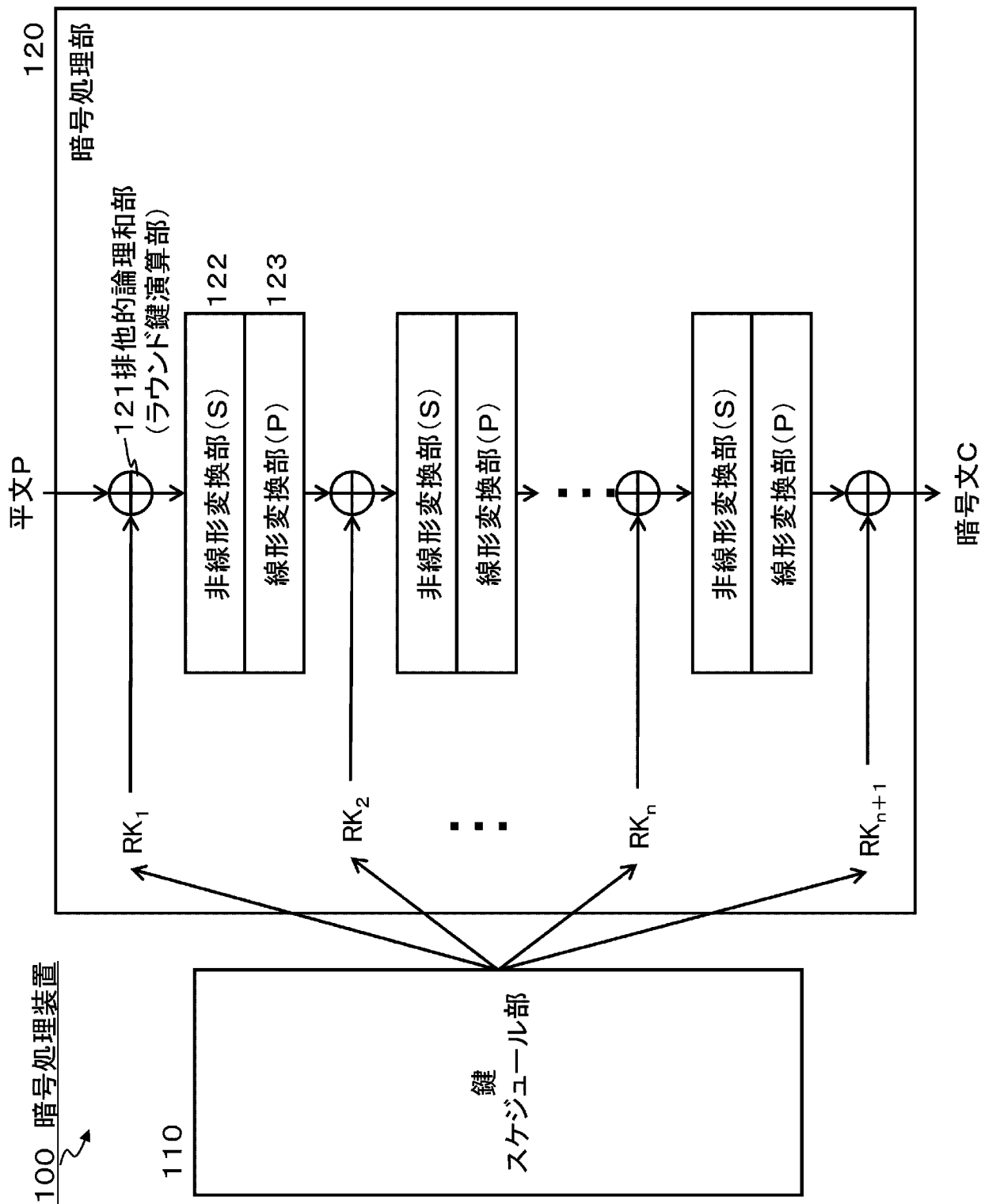
$$\begin{pmatrix} b_0 \\ b_m \\ \vdots \\ b_{(n-1)m} \end{pmatrix} = X_0 \cdot \begin{pmatrix} a_0 \\ a_m \\ \vdots \\ a_{(n-1)m} \end{pmatrix}, \quad \begin{pmatrix} b_1 \\ b_{m+1} \\ \vdots \\ b_{(n-1)m+1} \end{pmatrix} = X_1 \cdot \begin{pmatrix} a_1 \\ a_{m+1} \\ \vdots \\ a_{(n-1)m+1} \end{pmatrix},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

[図18]

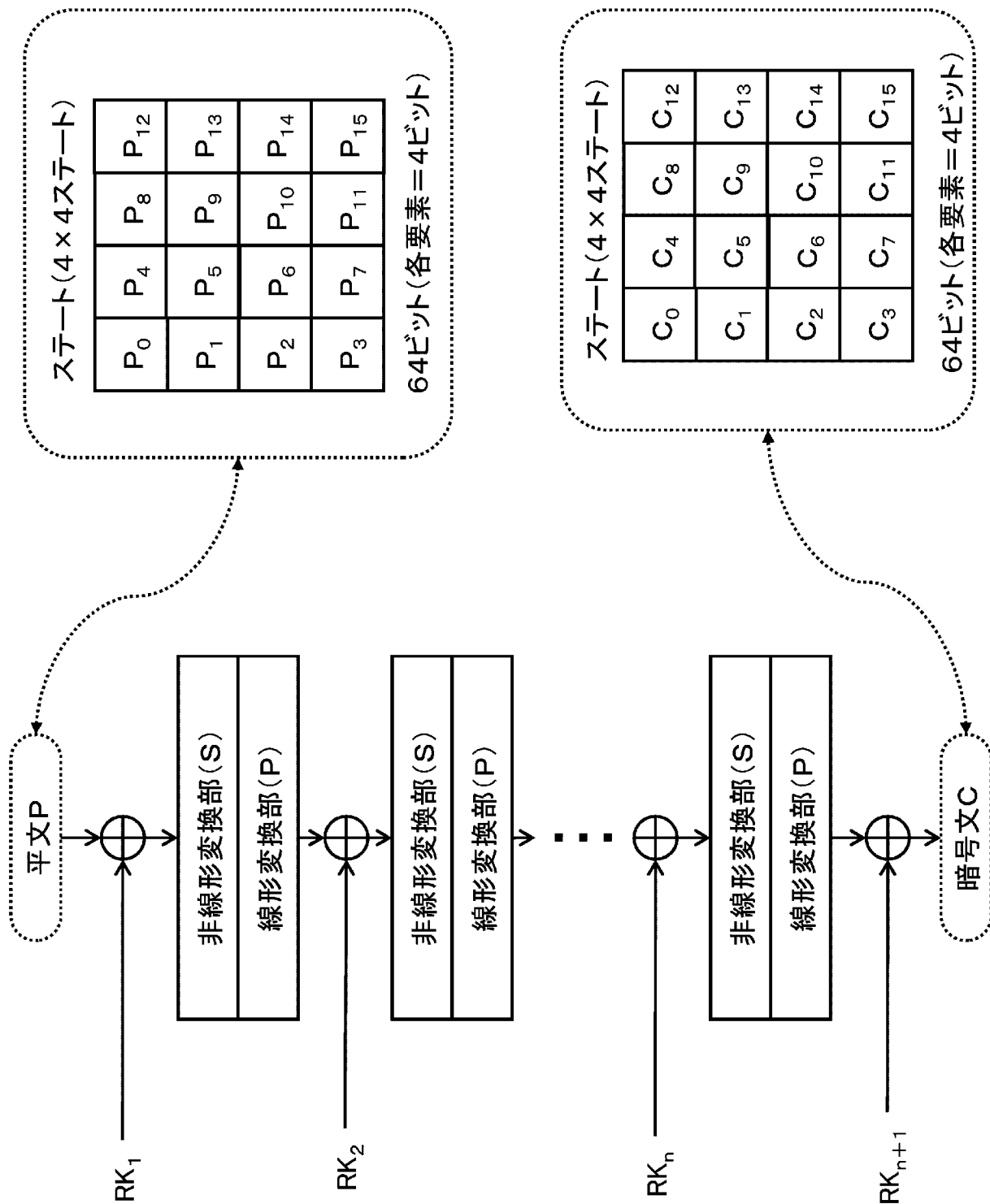


$$\begin{pmatrix} b_0 \\ b_4 \\ b_8 \\ b_{12} \end{pmatrix} = X_0 \cdot \begin{pmatrix} a_0 \\ a_4 \\ a_8 \\ a_{12} \end{pmatrix}, \begin{pmatrix} b_1 \\ b_5 \\ b_9 \\ b_{13} \end{pmatrix} = X_1 \cdot \begin{pmatrix} a_1 \\ a_5 \\ a_9 \\ a_{13} \end{pmatrix}, \begin{pmatrix} b_2 \\ b_6 \\ b_{10} \\ b_{14}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

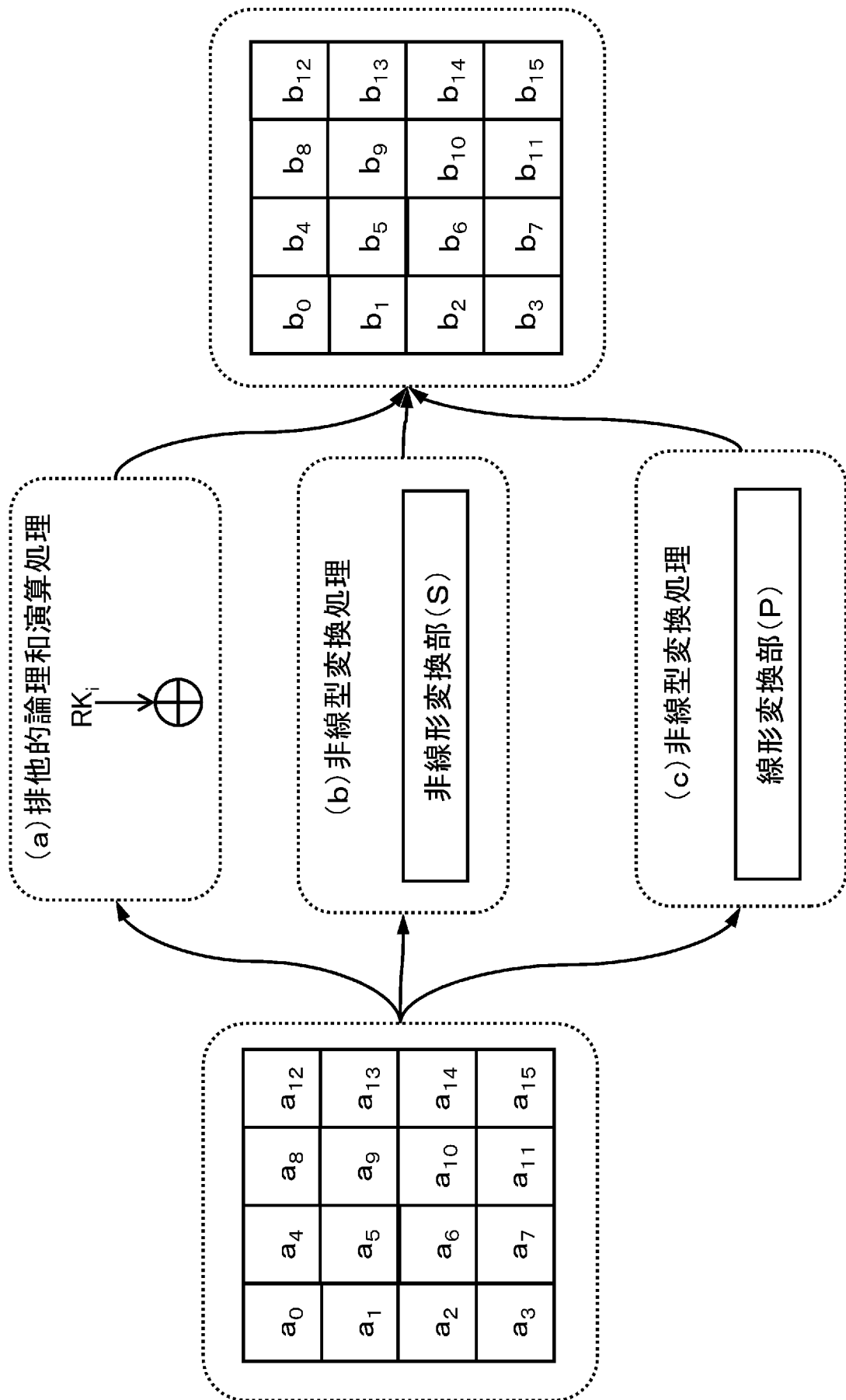
[図19]



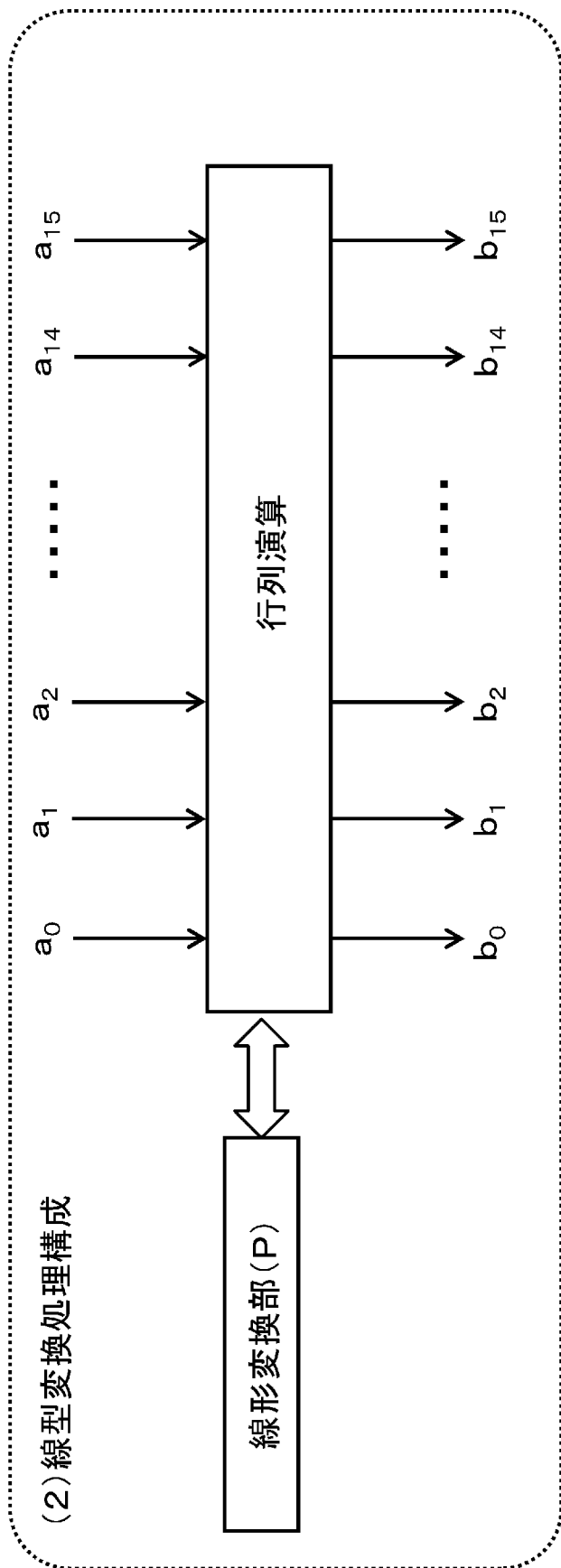
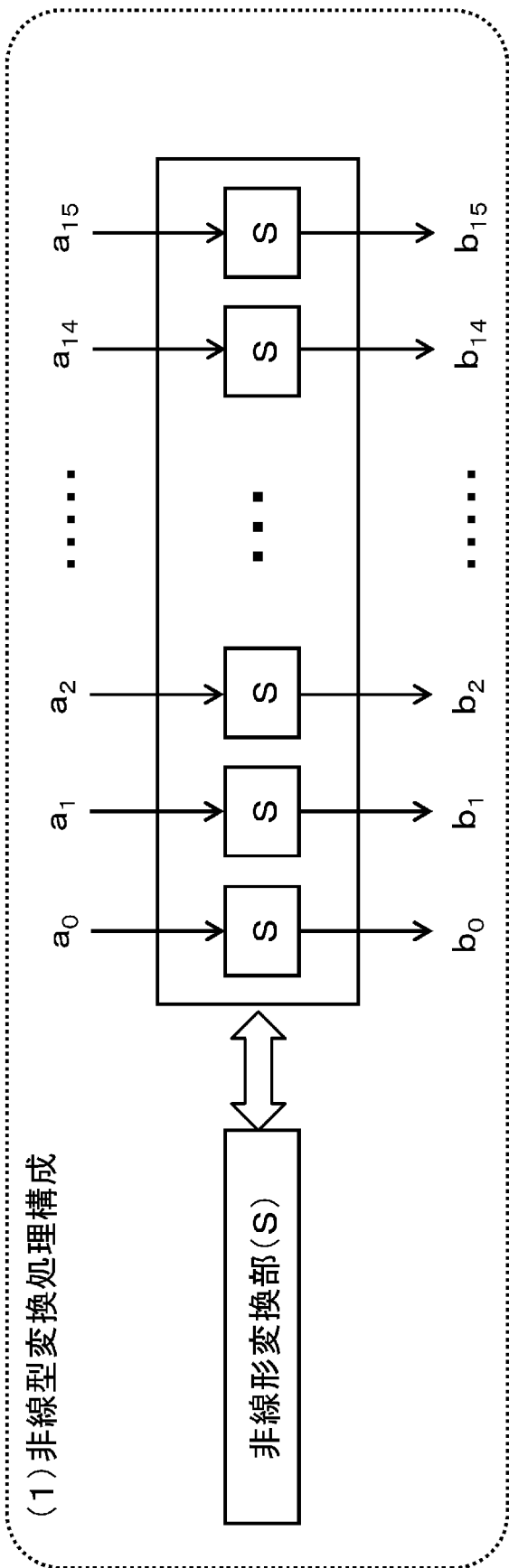
[図20]



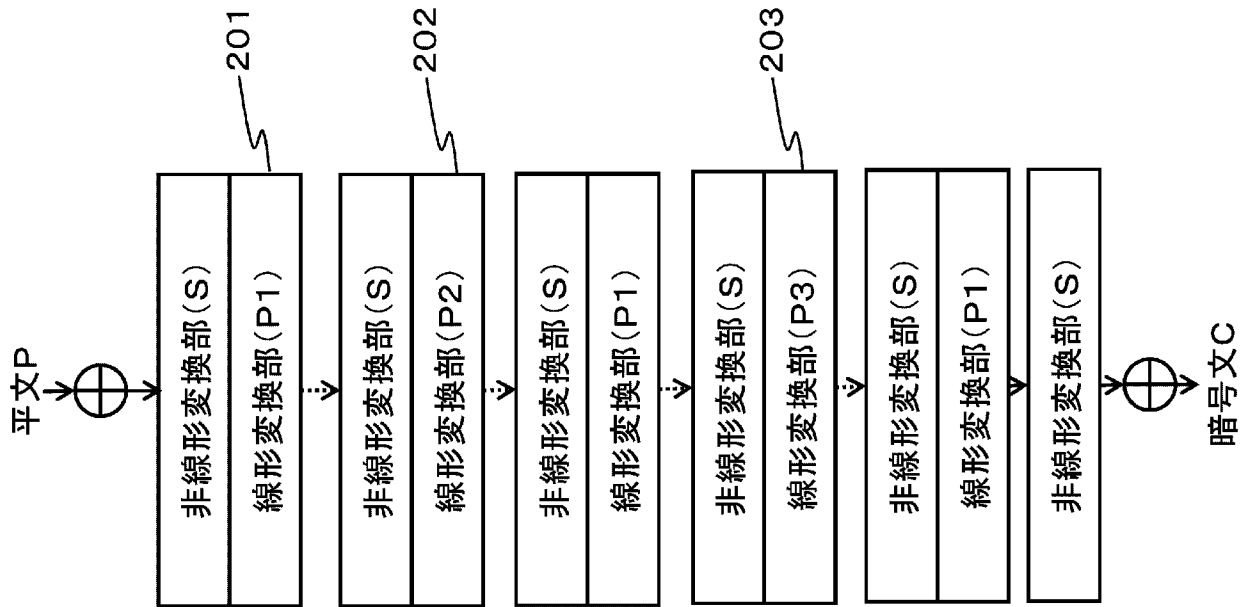
[圖21]



[図22]



[図23]



[図24]

(1) 線形変換処理に適用する4種類の行列(Matrix)の例

$$\begin{array}{ccc}
 M_0 & M_1 & M_2 \\
 \left(\begin{array}{c} 0111 \\ 1011 \\ 1101 \\ 1110 \end{array} \right) & \left(\begin{array}{c} 1101 \\ 1110 \\ 0111 \\ 1011 \end{array} \right) & \left(\begin{array}{c} 1110 \\ 1101 \\ 1011 \\ 0111 \end{array} \right) \\
 M_3 & & \\
 \left(\begin{array}{c} 1011 \\ 0111 \\ 1110 \\ 1101 \end{array} \right) & &
 \end{array}$$

(2) 線形変換処理に適用する3種類の行列演算の例

P1: 列拡散演算
(MixColumn)

M_0 M_0 M_0 M_0

MC[M_0]
(= MC[M_0, M_0, M_0, M_0])

P2: 行拡散演算タイプ1
(MixRow1)

M_0
 M_1
 M_2
 M_3

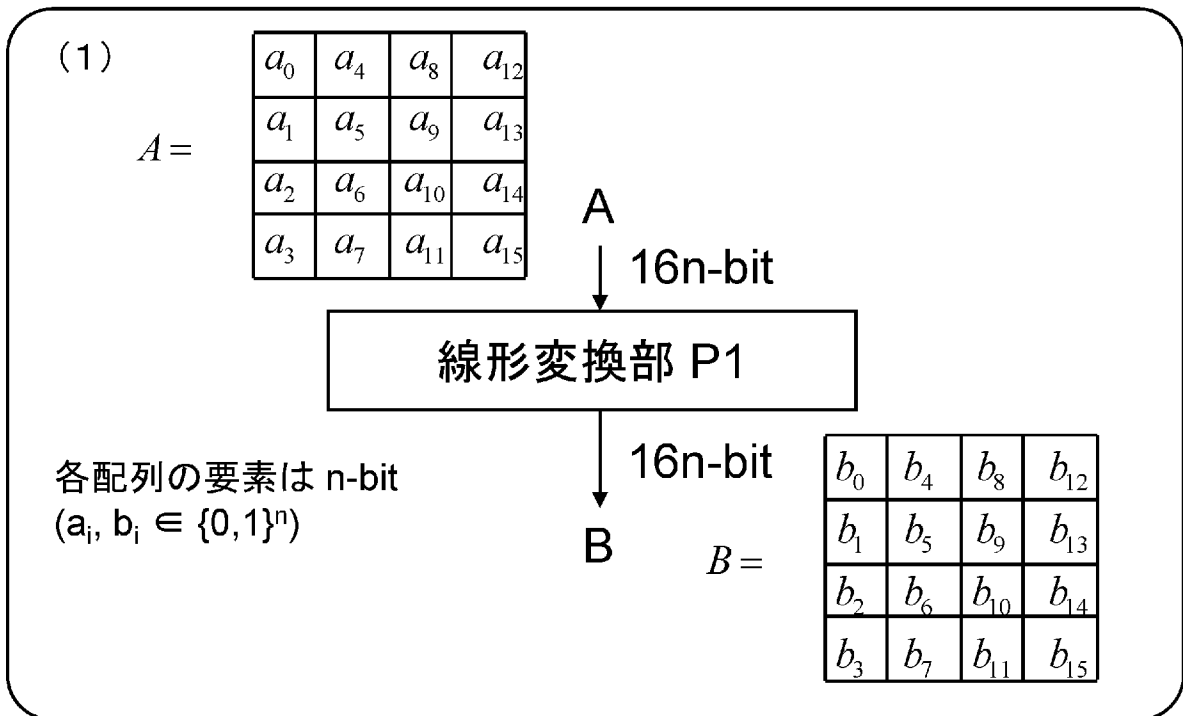
MR[M_0, M_1, M_2, M_3]

P2: 行拡散演算タイプ2
(MixRow2)

M_2
 M_0
 M_1
 M_3

MR[M_2, M_0, M_1, M_3]

[図25]



(2)

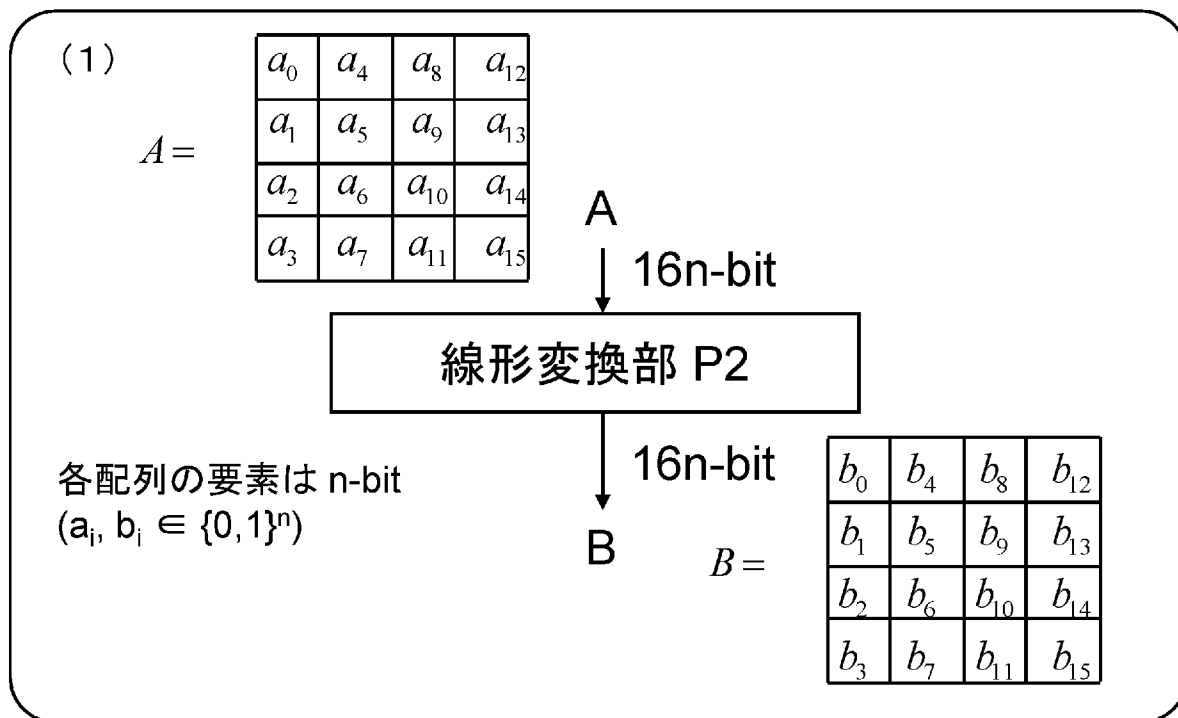
$$\begin{pmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \end{pmatrix} = \begin{pmatrix} 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{pmatrix} \cdot \begin{pmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \end{pmatrix}, \quad \begin{pmatrix} b_4 \\ b_5 \\ b_6 \\ b_7 \end{pmatrix} = \begin{pmatrix} 0 & 1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

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

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

(ただし、ここで + は、XOR(排他的論理和)を表す)

[図26]



(2)

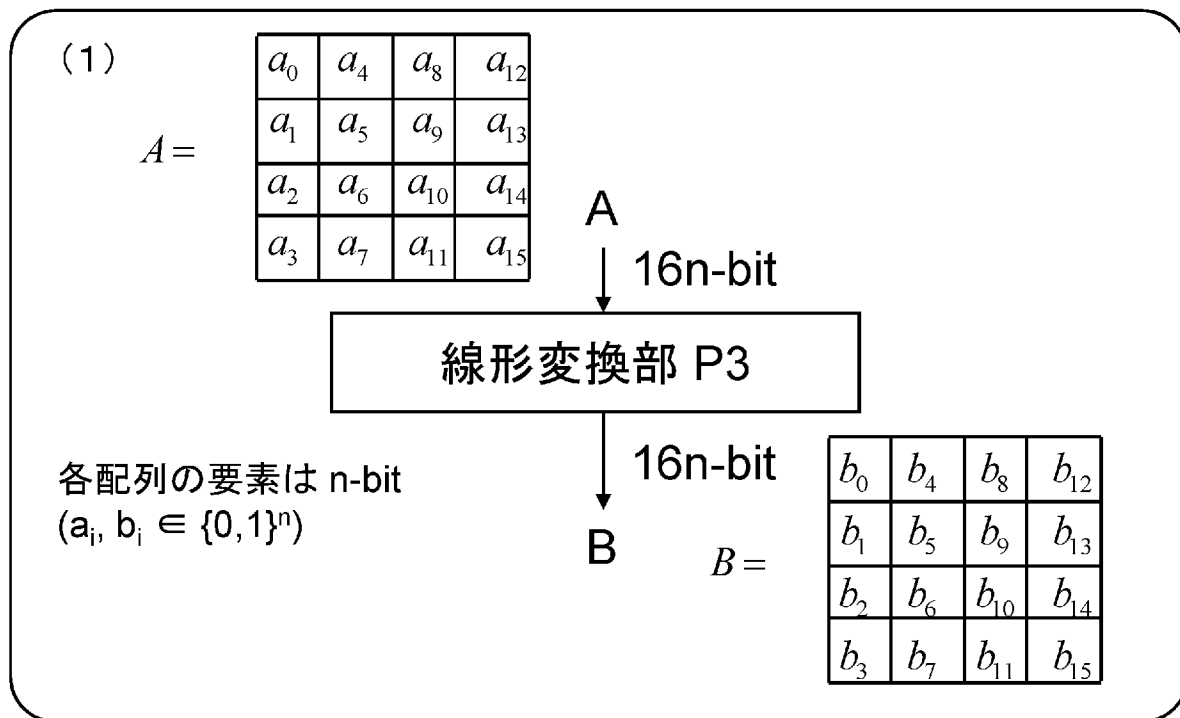
$$\begin{pmatrix} b_0 \\ b_4 \\ b_8 \\ b_{12} \end{pmatrix} = \begin{pmatrix} 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{pmatrix} \begin{pmatrix} a_0 \\ a_4 \\ a_8 \\ a_{12} \end{pmatrix}, \quad \begin{pmatrix} b_1 \\ b_5 \\ b_9 \\ b_{13} \end{pmatrix} = \begin{pmatrix} 1 &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

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

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

(ただし、ここで + は、XOR(排他的論理和)を表す)

[図27]



(2)

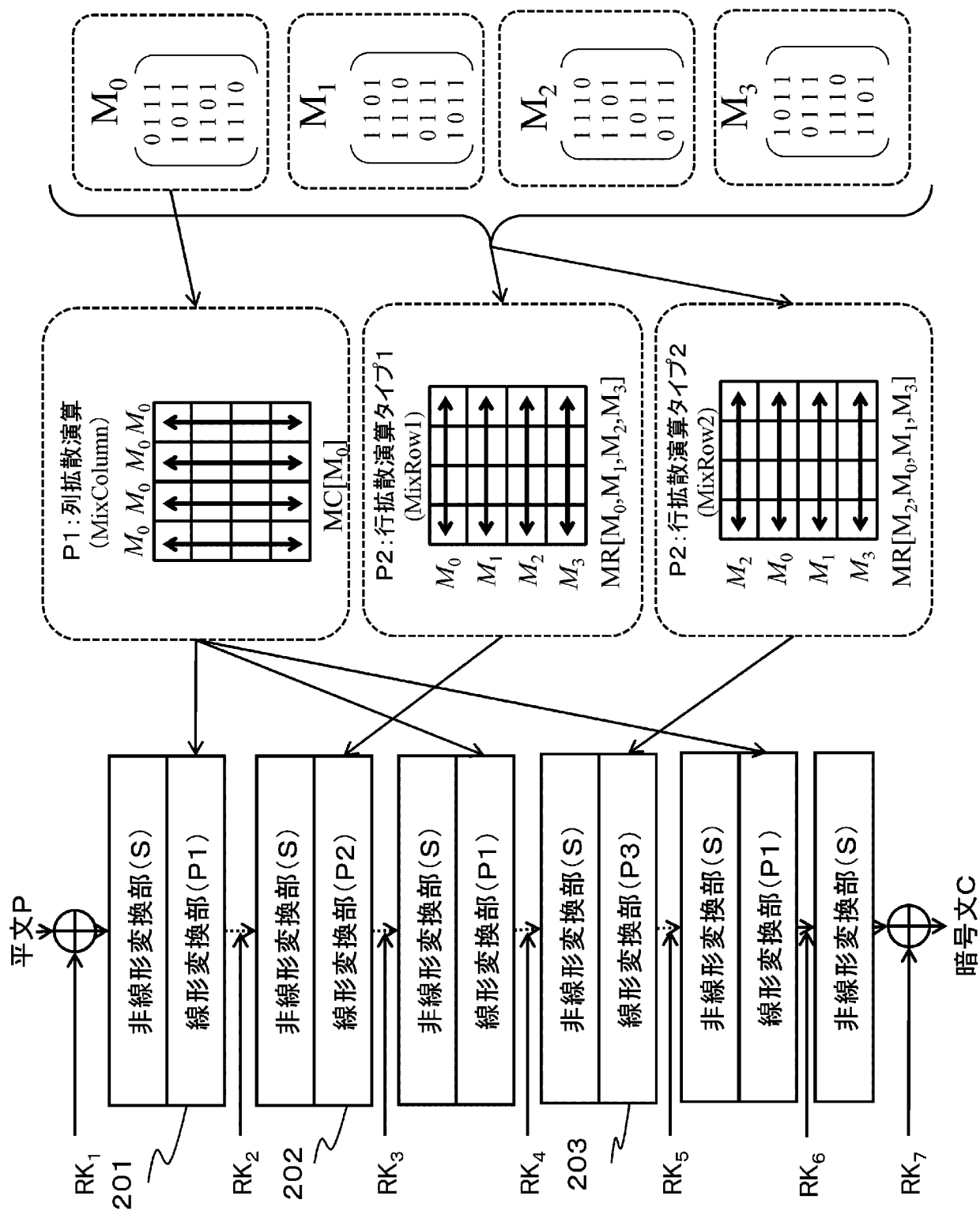
$$\begin{pmatrix} b_0 \\ b_4 \\ b_8 \\ b_{12} \end{pmatrix} = \begin{pmatrix} 1 & 1 & 1 & 0 \\ 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 0 & 1 & 1 & 1 \end{pmatrix} \cdot \begin{pmatrix} a_0 \\ a_4 \\ a_8 \\ a_{12} \end{pmatrix}, \quad \begin{pmatrix} b_1 \\ b_5 \\ b_9 \\ b_{13} \end{pmatrix} = \begin{pmatrix}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

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

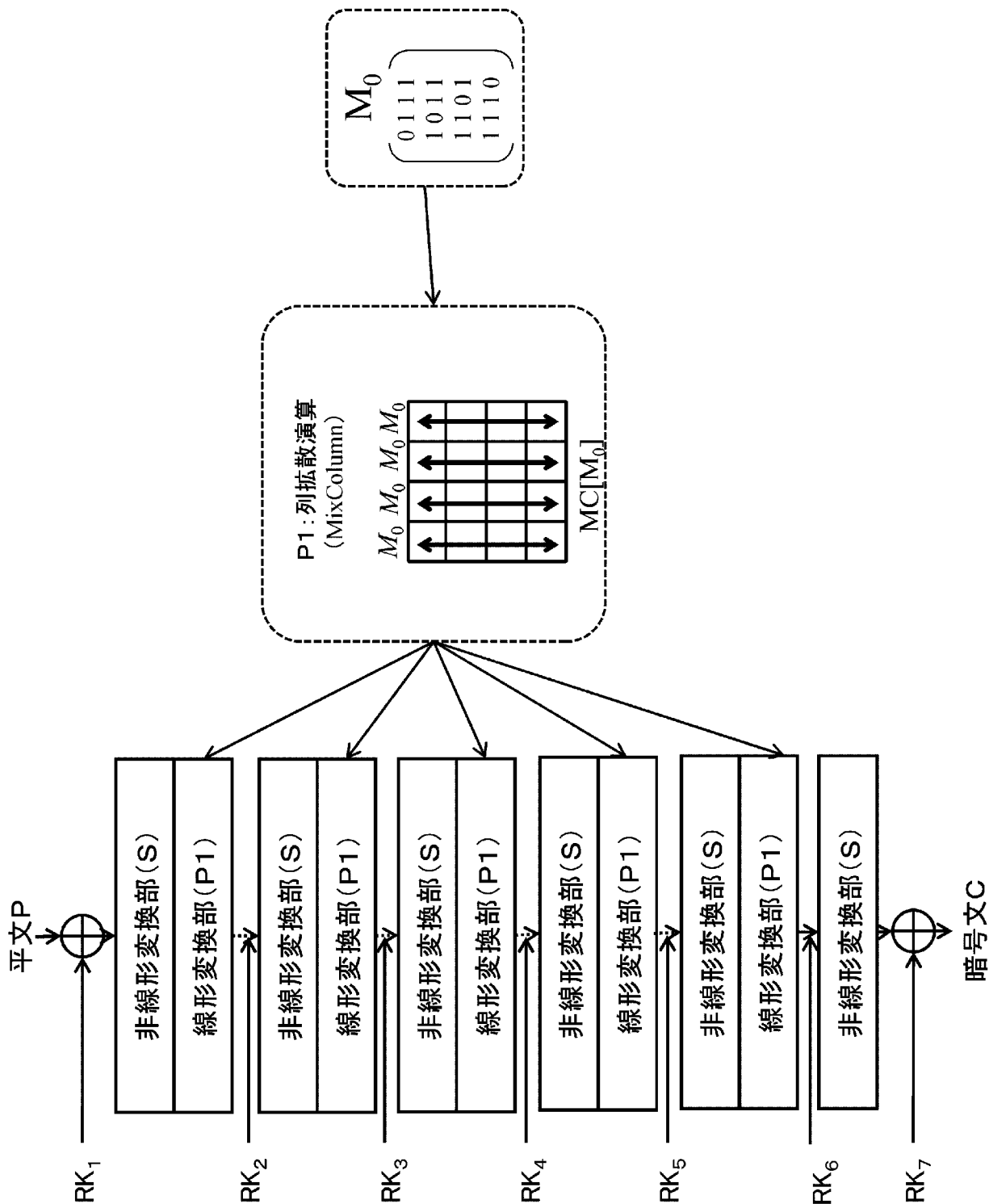
$$\begin{aligned} b_0 &= a_0 + a_4 + a_8, & b_8 &= a_0 + a_8 + a_{12}, \\ b_1 &= a_5 + a_9 + a_{13}, & b_9 &= a_1 + a_5 + a_{13}, \\ b_2 &= a_2 + a_6 + a_{14}, & b_{10} &= a_6 + a_{10} + a_{14}, \\ b_3 &= a_3 + a_{11} + a_{15}, & b_{11} &= a_3 + a_7 + a_{11}, \\ b_4 &= a_0 + a_4 + a_{12}, & b_{12} &=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

(ただし、ここで + は、XOR(排他的論理和)を表す)

[図28]



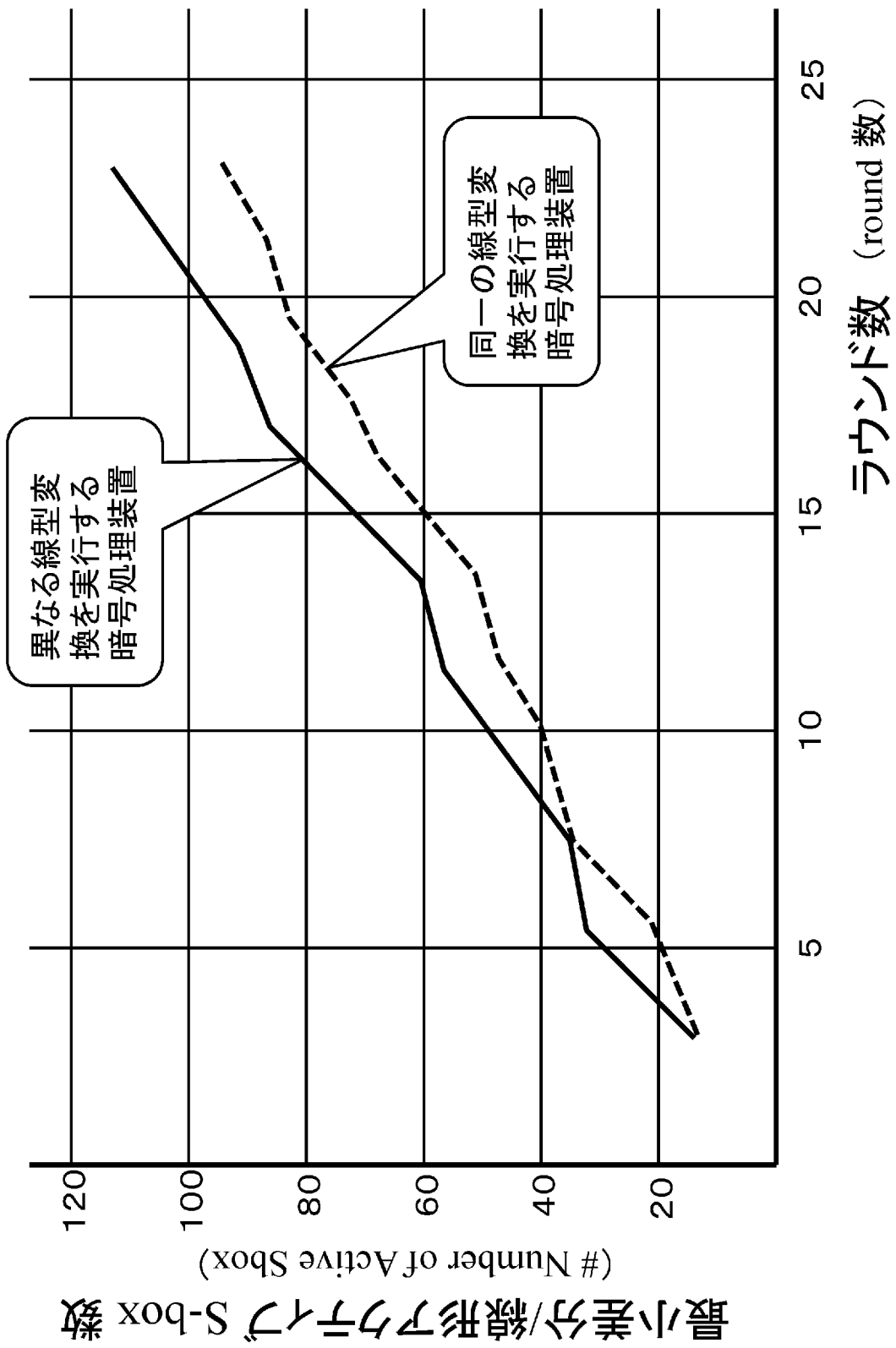
[圖29]



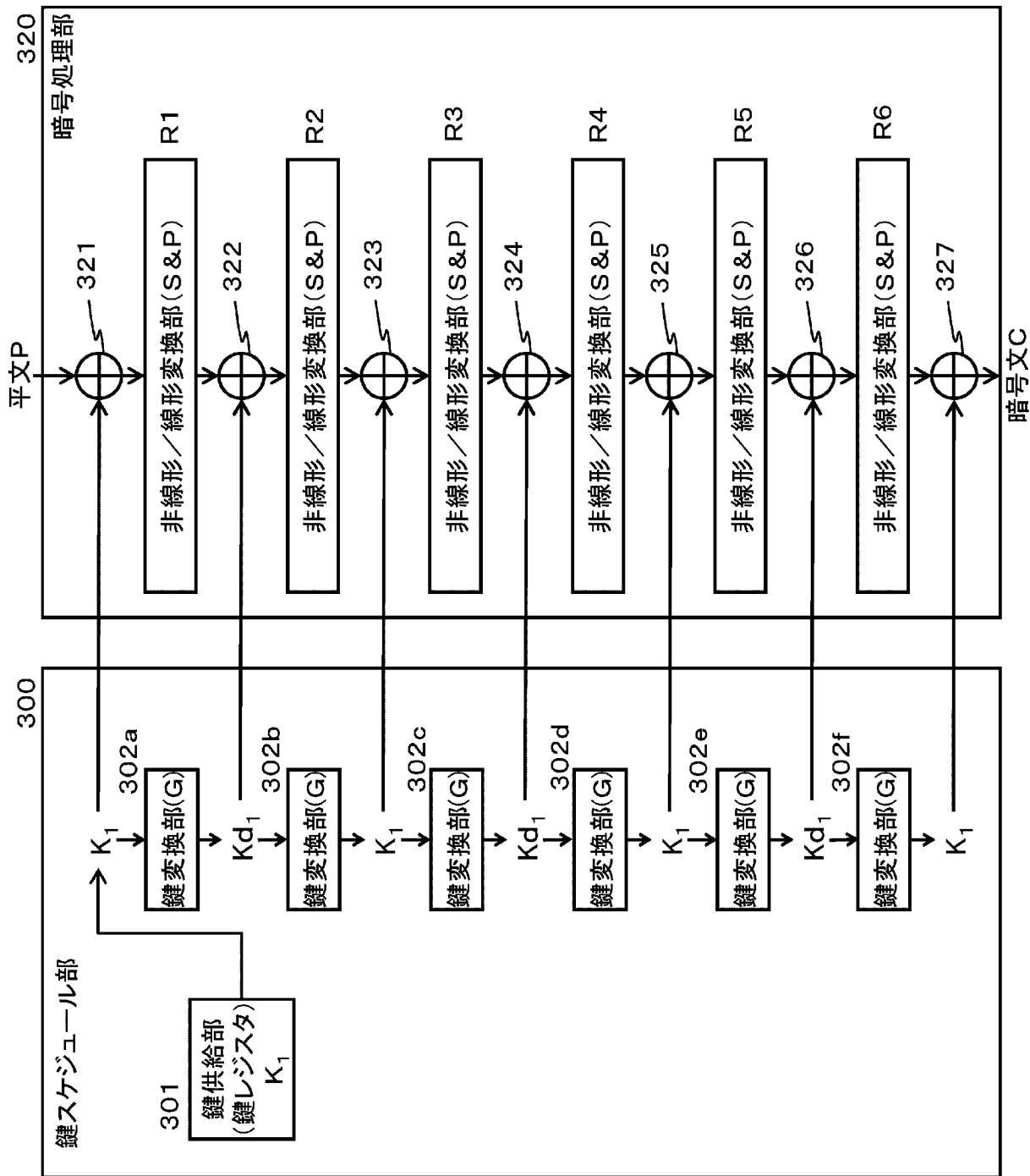
[図30]

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

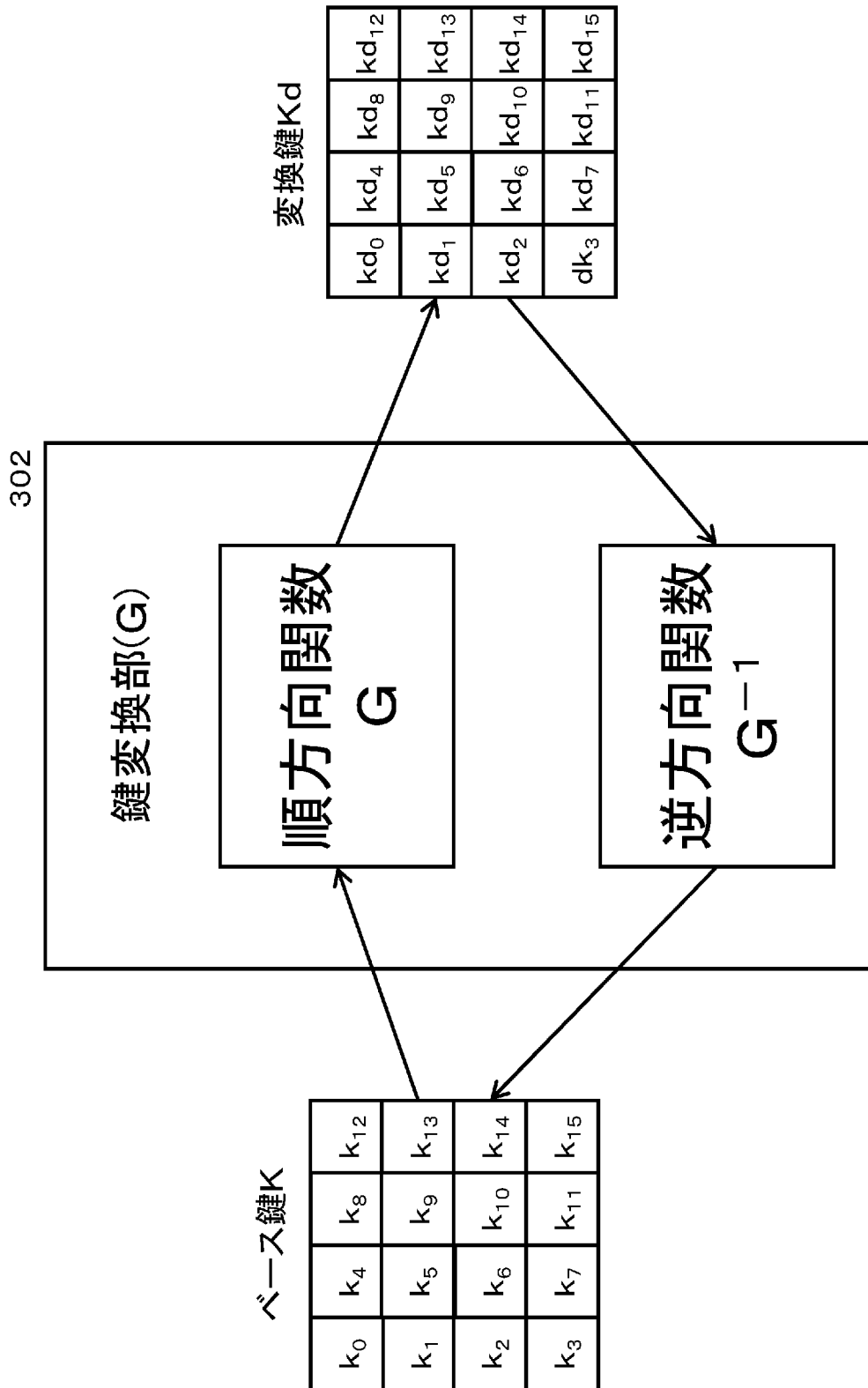
[図31]



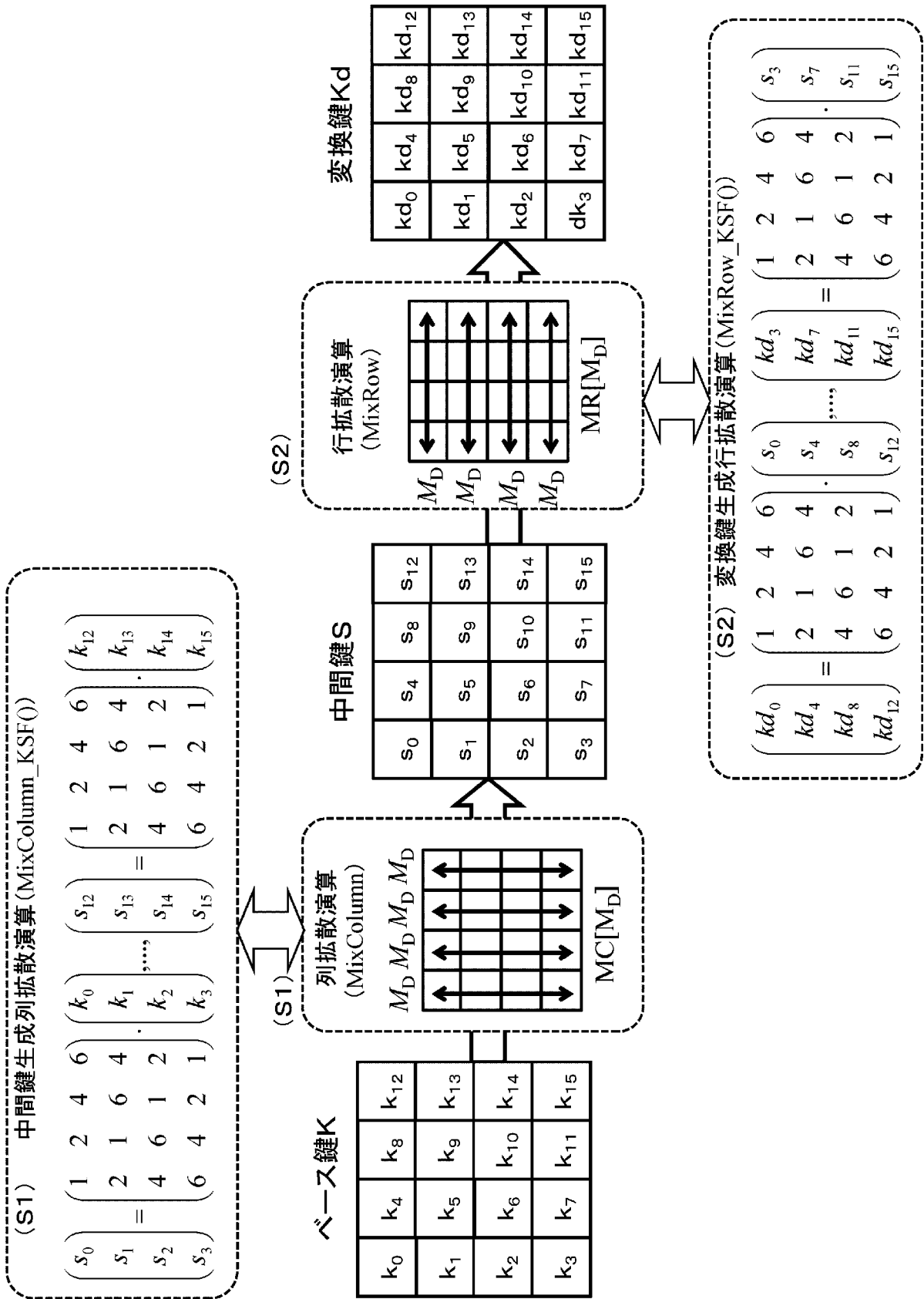
[図32]



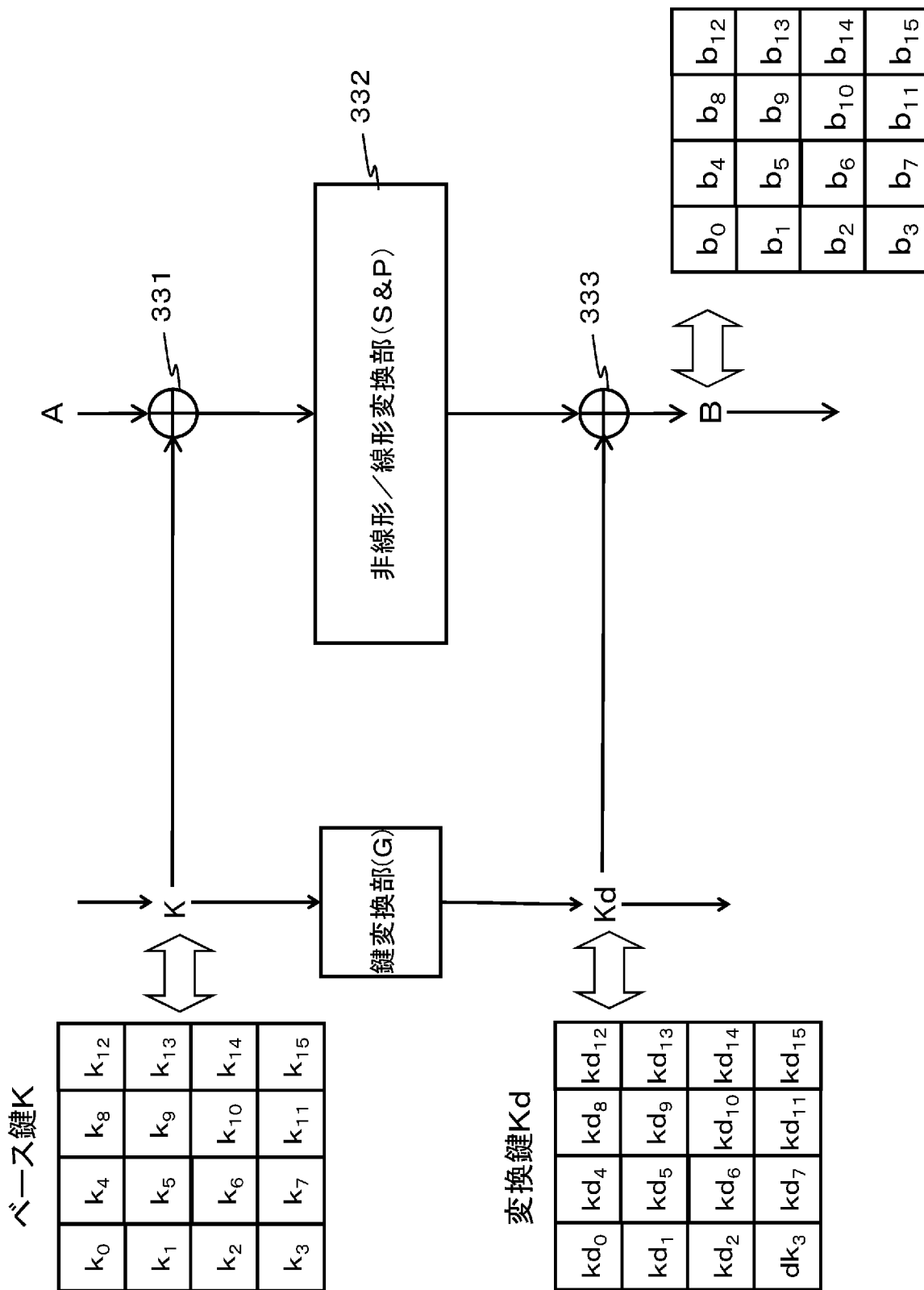
[図33]



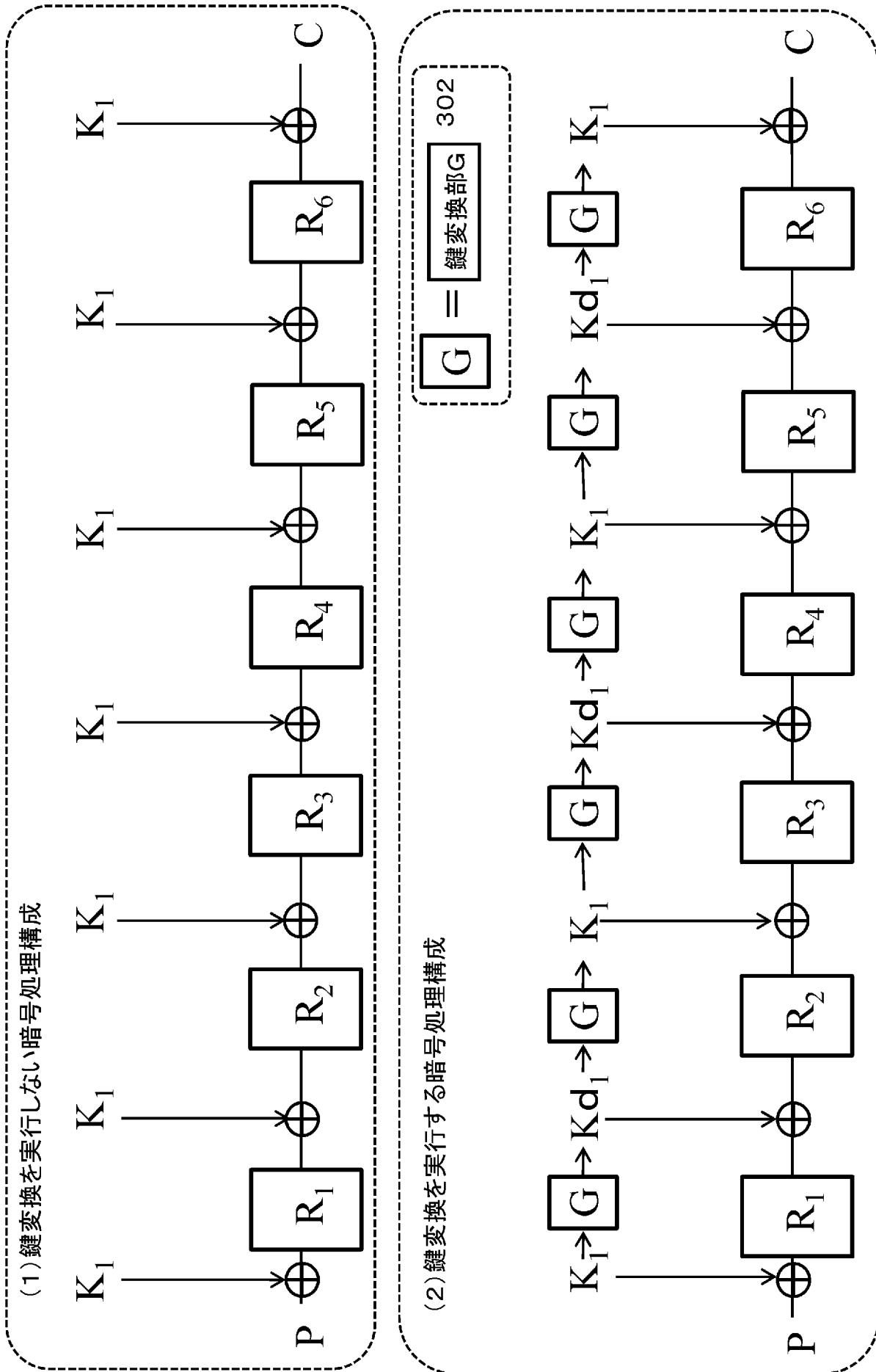
[図34]



[図35]

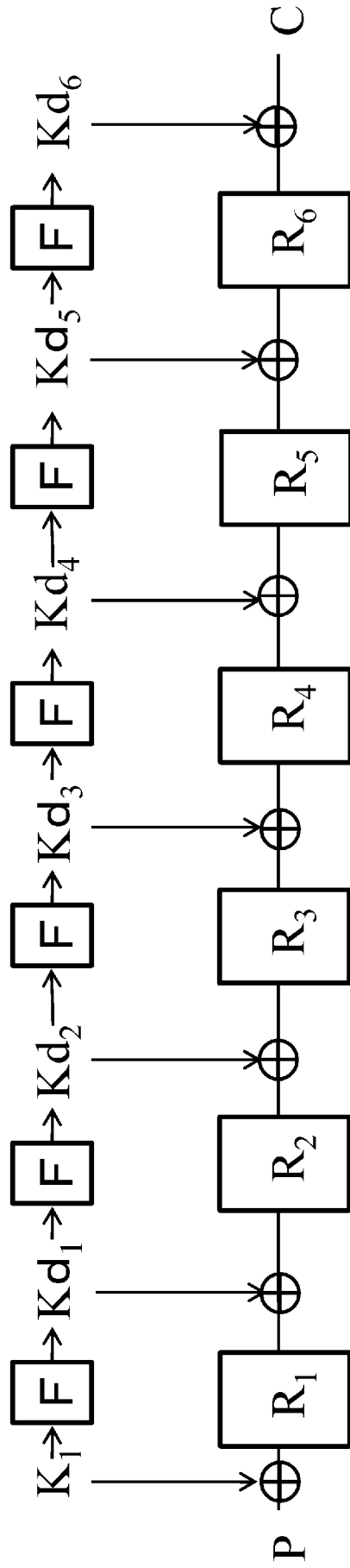


[図36]

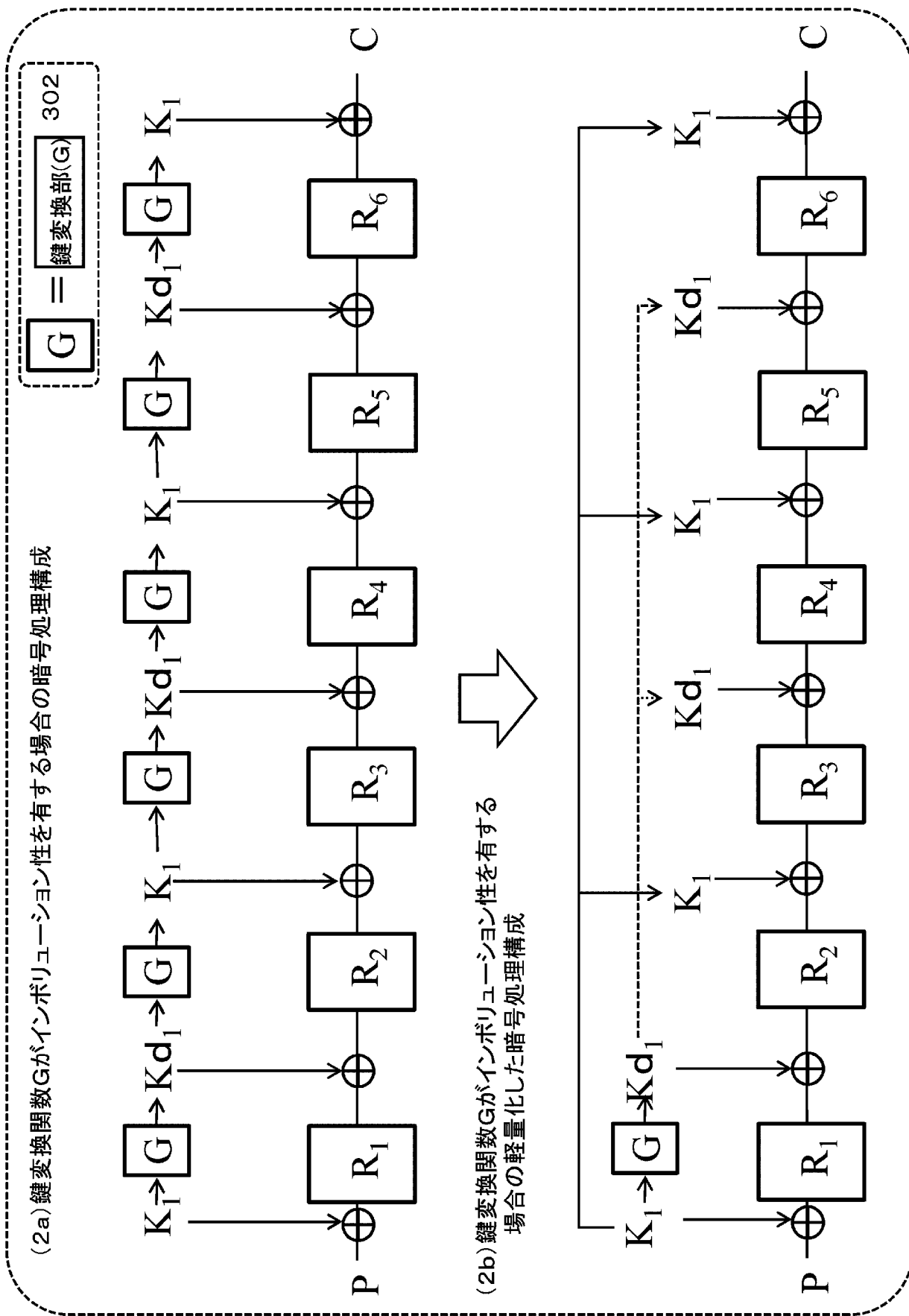


[図37]

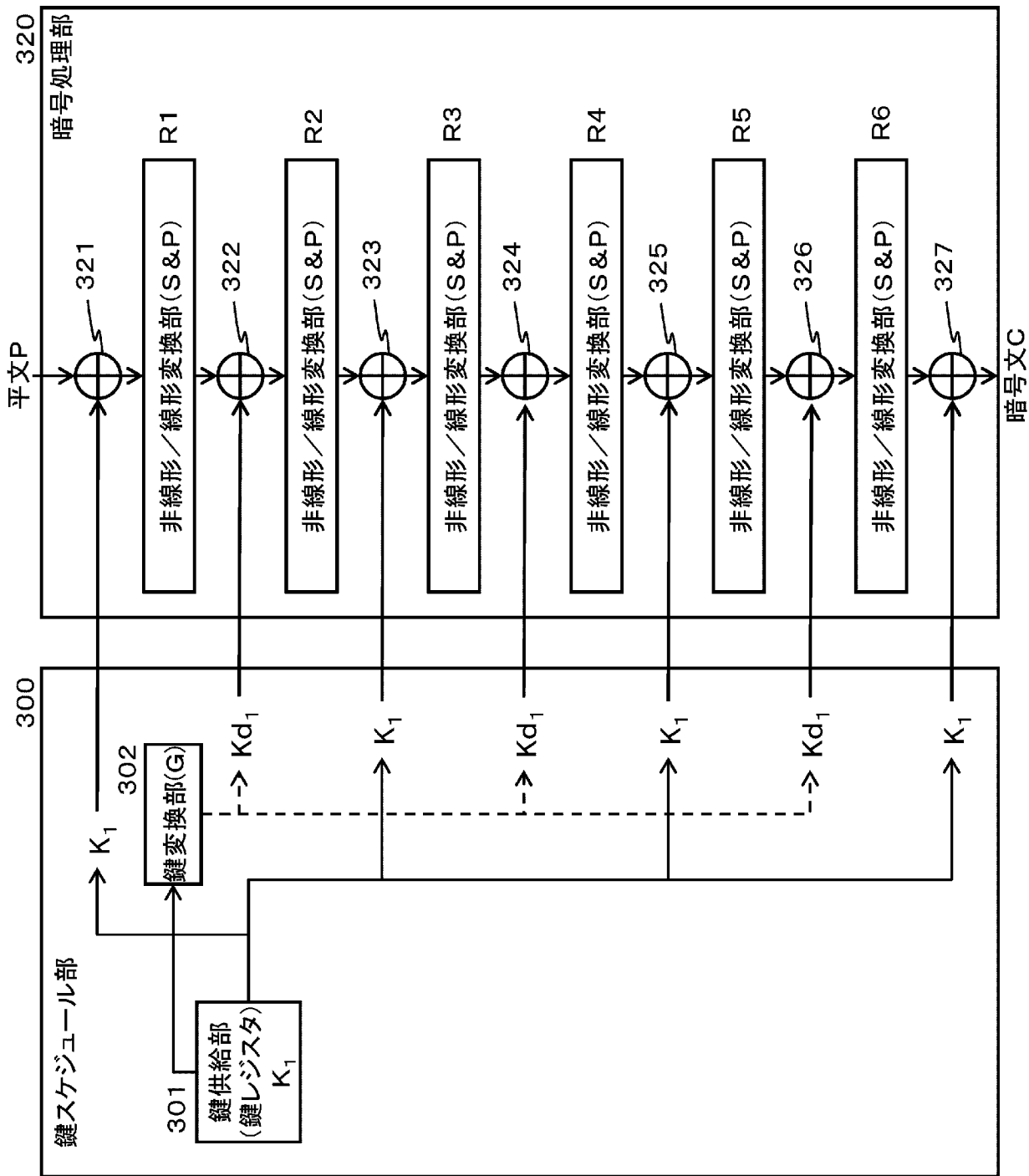
(1) 鍵変換関数Fがインボリューション性なしの場合の暗号処理構成



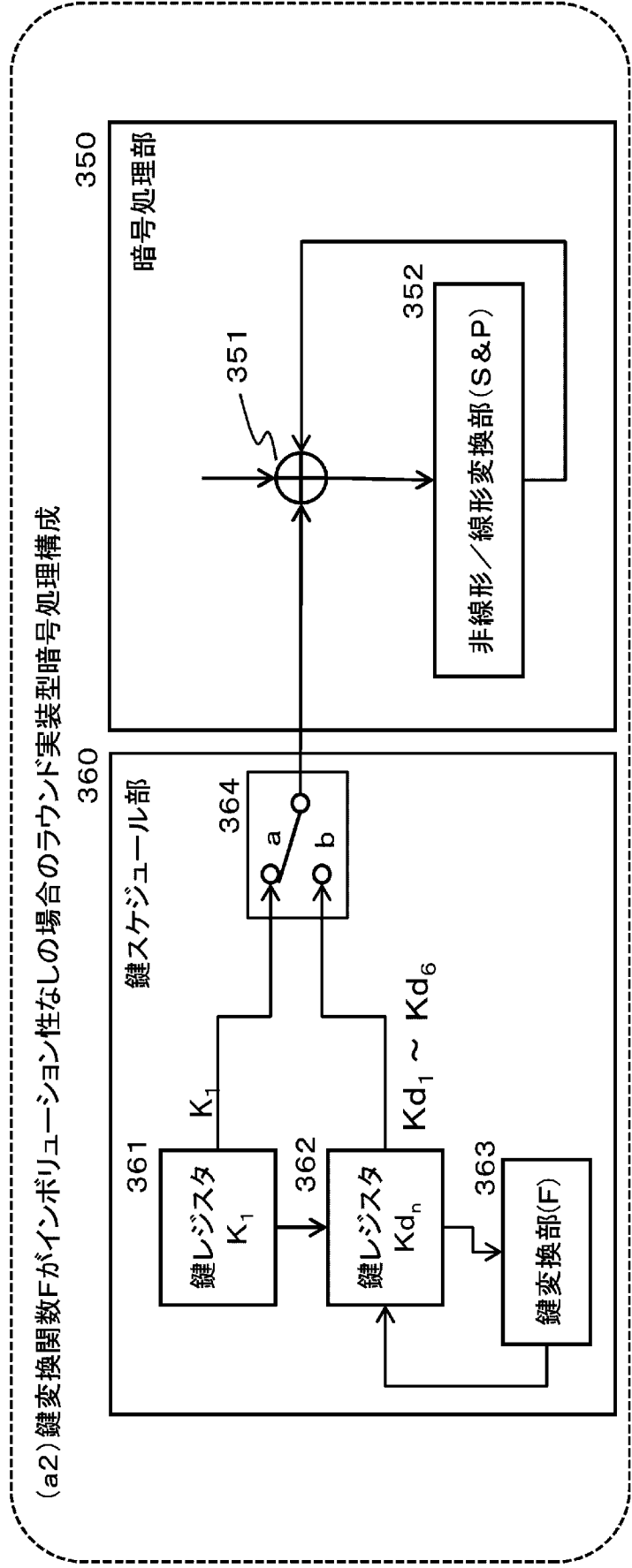
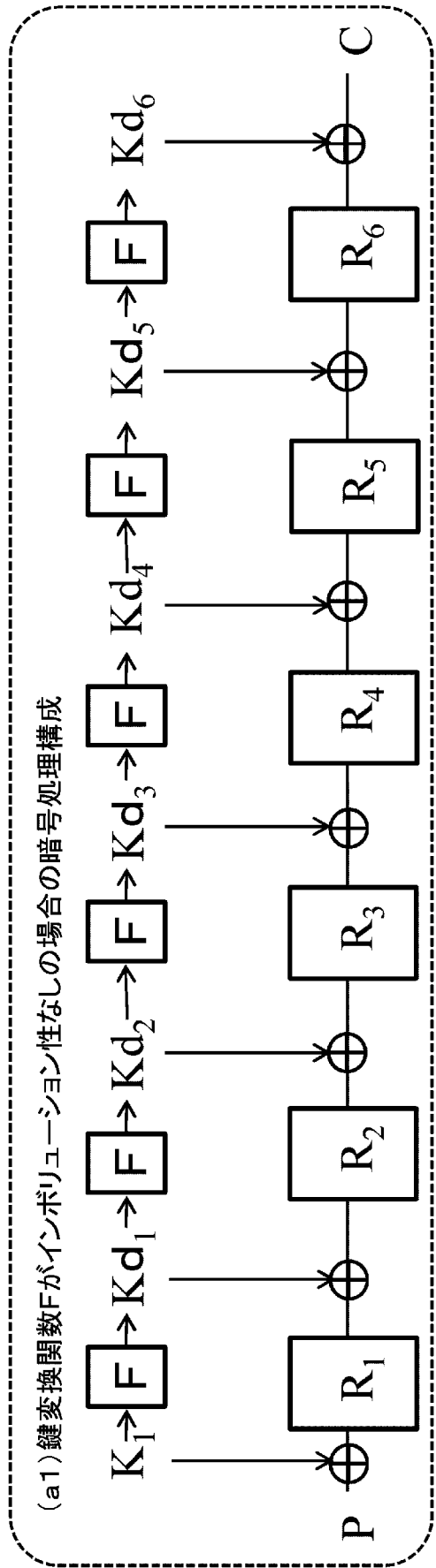
[図38]



[図39]

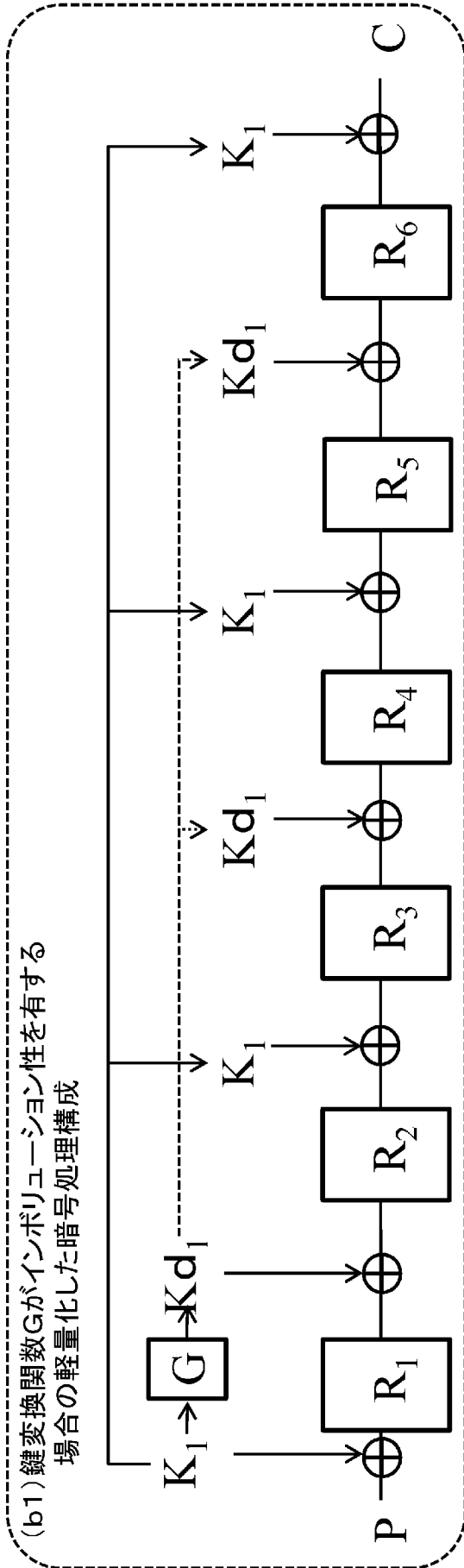


[図40]

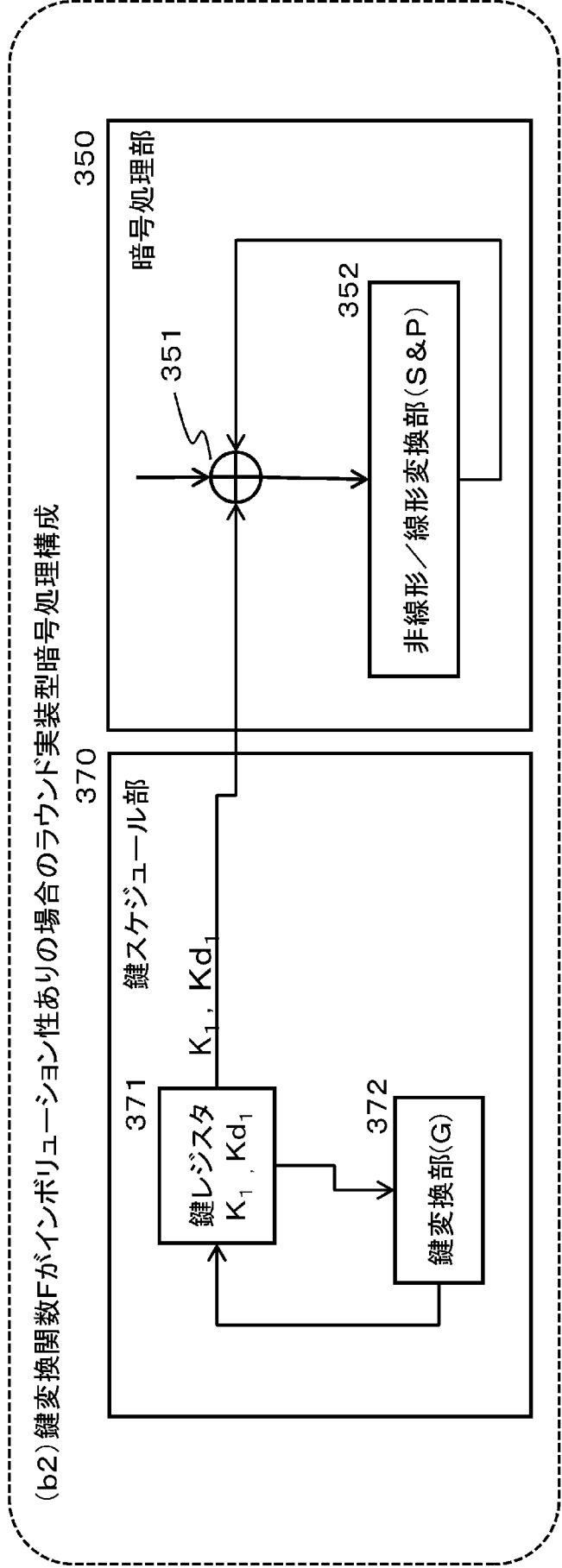


[図41]

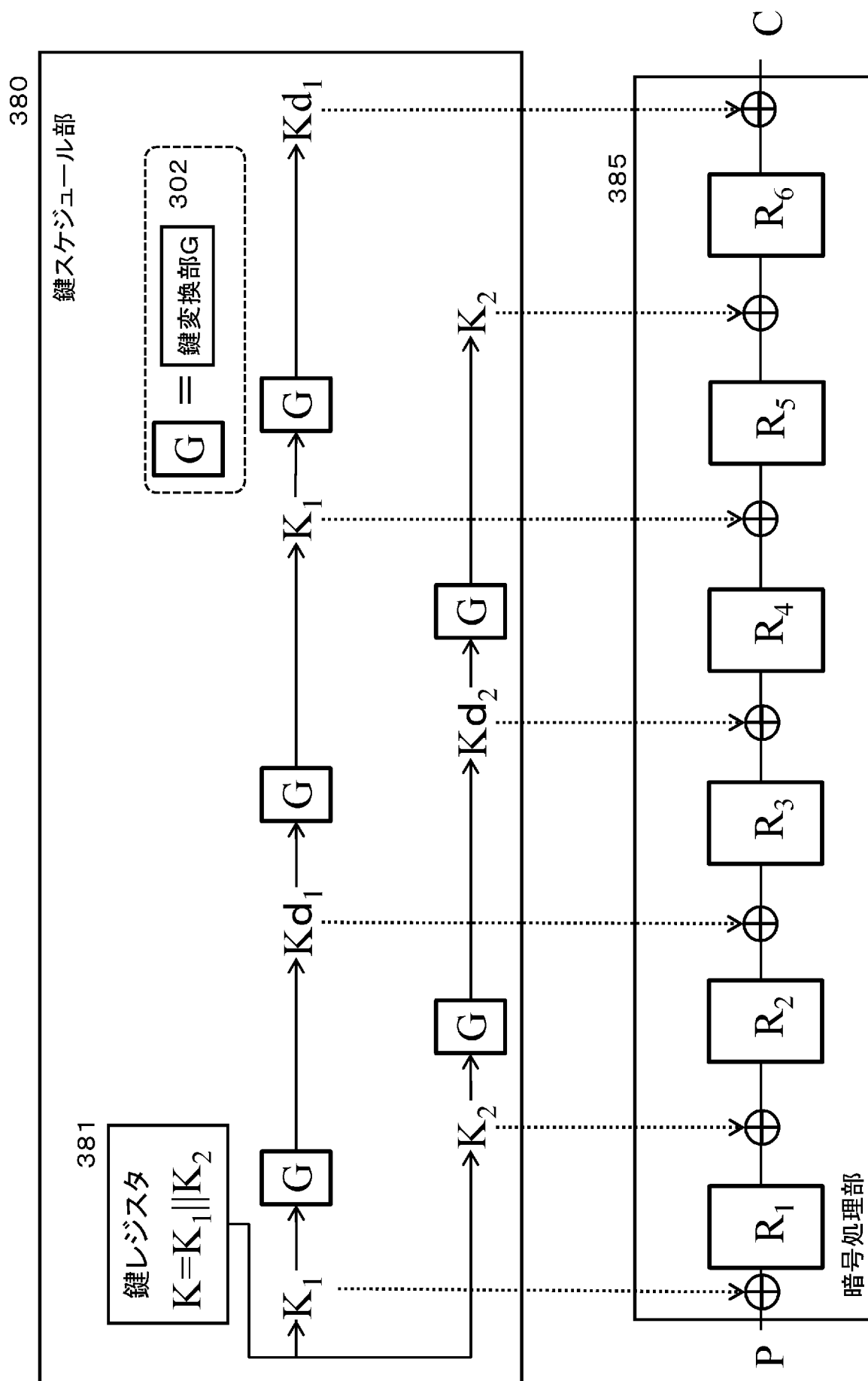
(b1) 鍵変換関数Gがインボリューション性を有する場合の軽量化した暗号処理構成



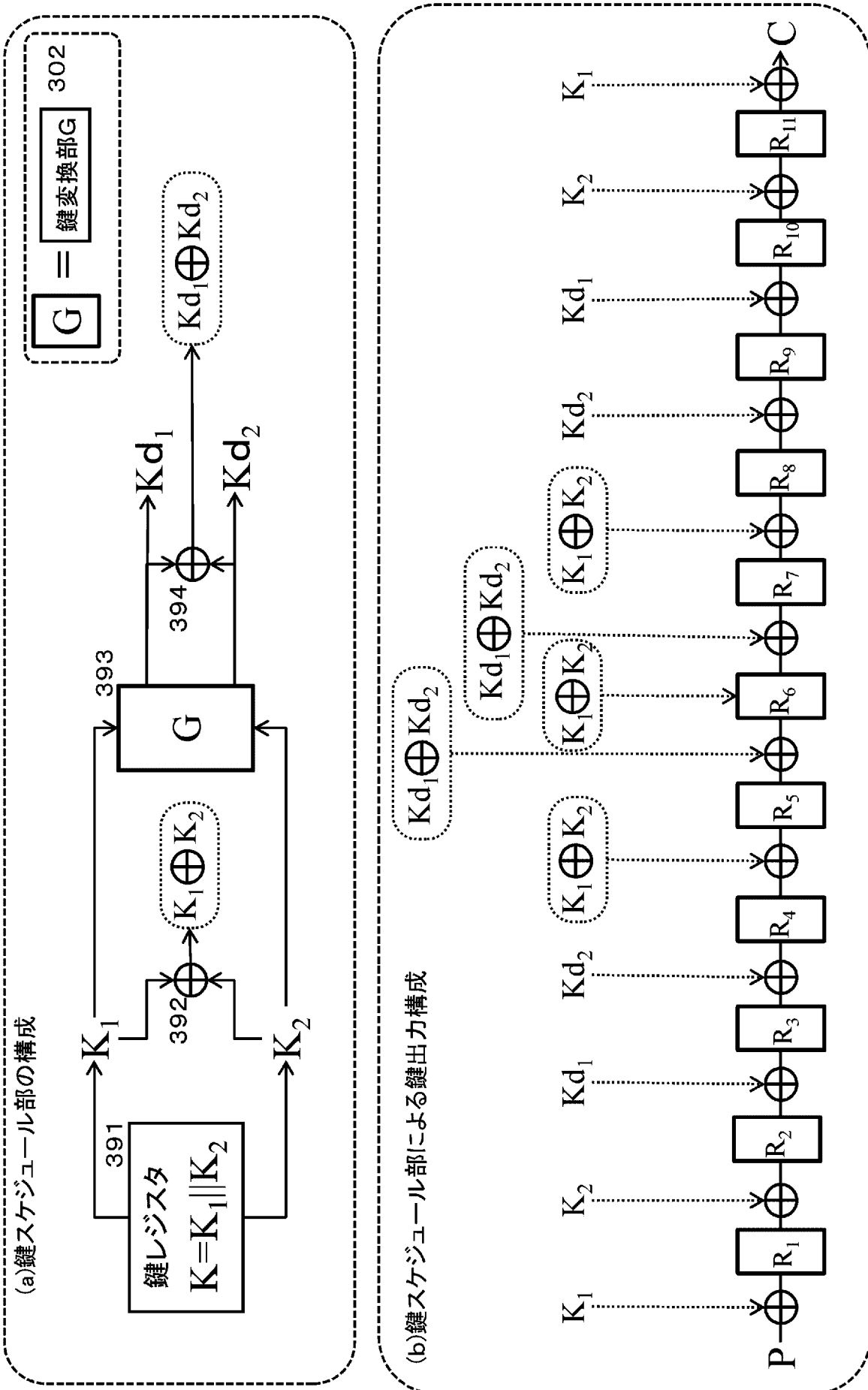
(b2) 鍵変換関数Fがインボリューション性ありの場合のラウンド実装型暗号処理構成



[図42]

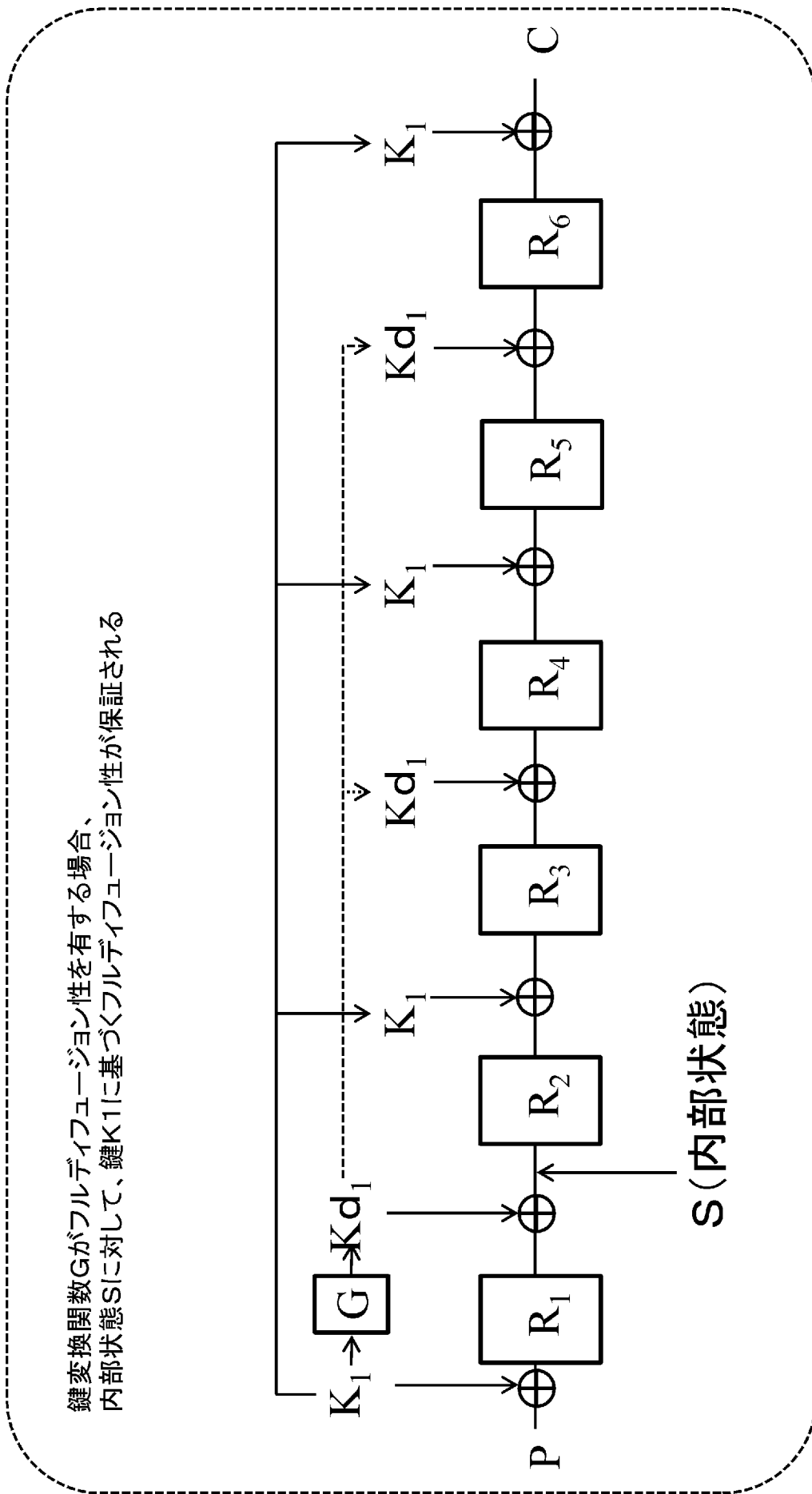


[図43]

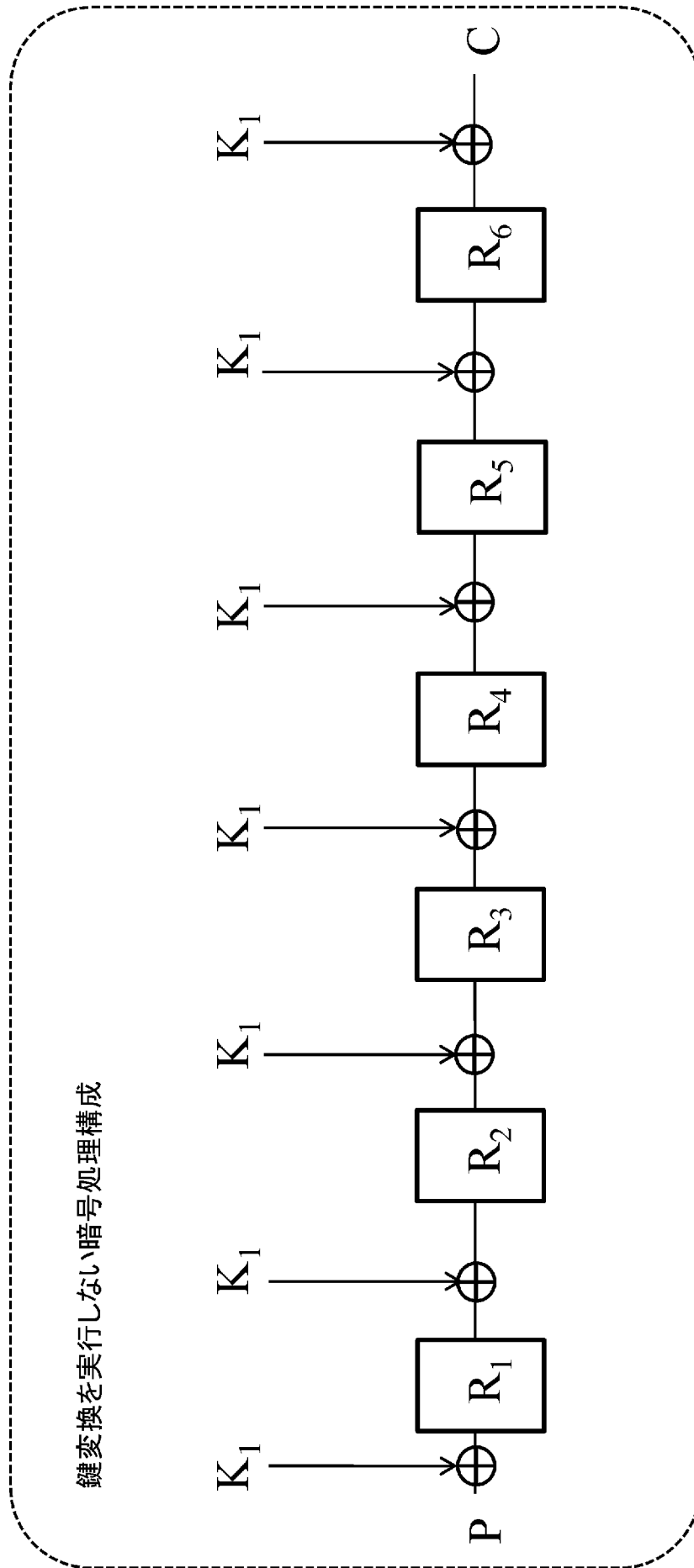


[図44]

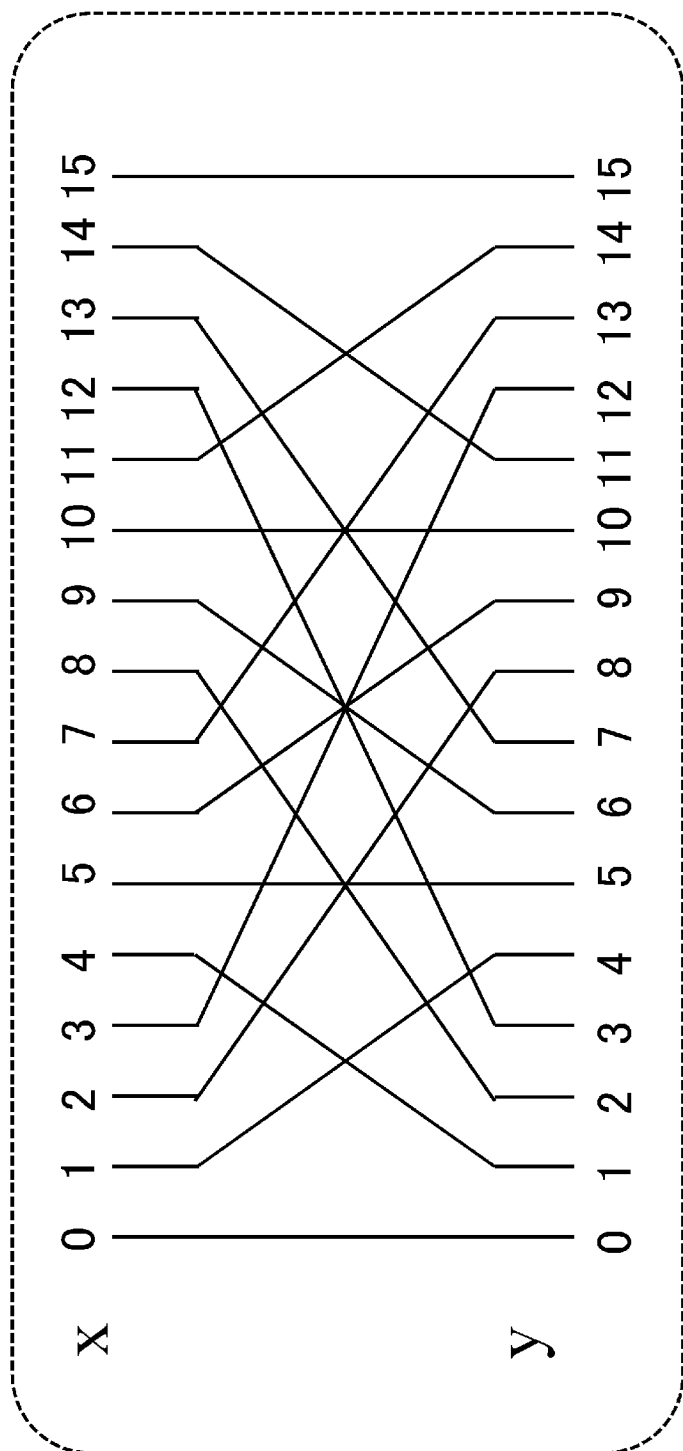
鍵変換関数Gがフルディフュージョン性を有する場合、
内部状態Sに対して、鍵K1に基づくフルディフュージョン性が保証される



[図45]



[図46]

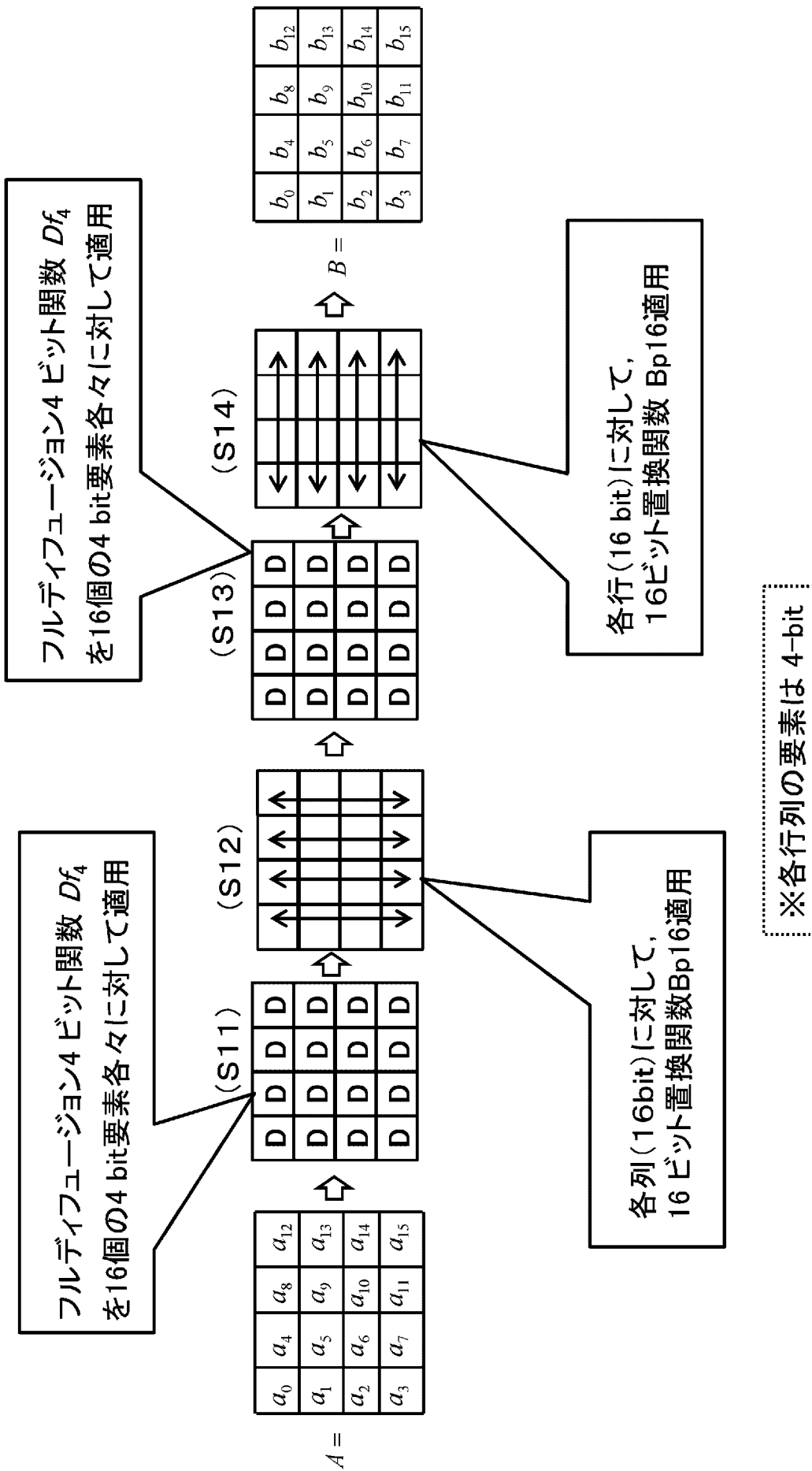


16ビット ビット置換: Bp16

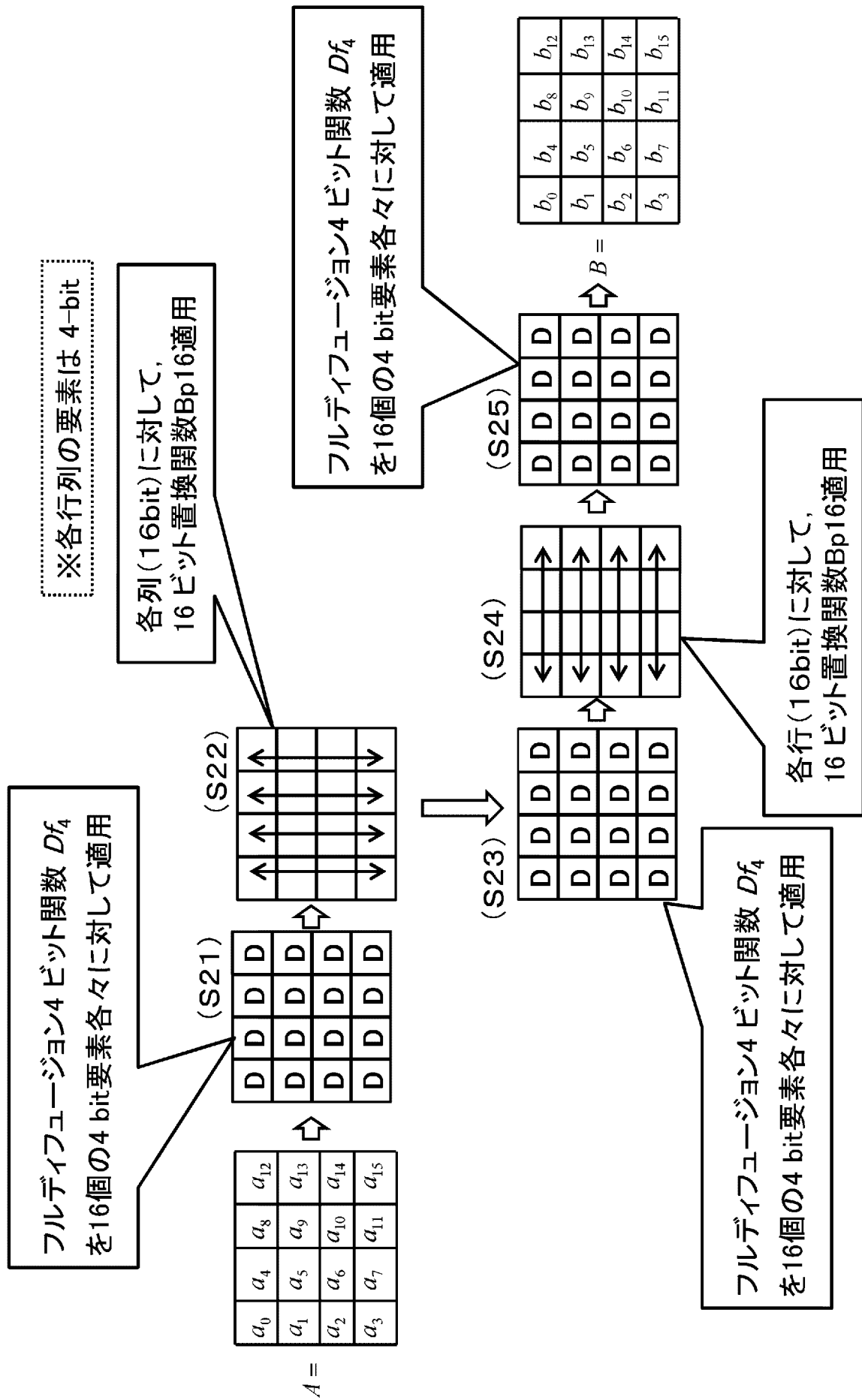
- 入力 : x_0, x_1, \dots, x_{15} (各1 bit)
- 出力 : y_0, y_1, \dots, y_{15} (各1 bit)
- 関数 : $y_0 = x_0, y_1 = x_4, y_2 = x_8, y_3 = x_{12}$
 $y_4 = x_1, y_5 = x_5, y_6 = x_9, y_7 = x_{13}$
 $y_8 = x_2, y_9 = x_6, y_{10} = x_{10}, y_{11} = x_{14}$
 $y_{12} = x_3, y_{13} = x_7, y_{14} = x_{11}, y_{15} = x_{15}$

-備考: インボリューション性も持つ

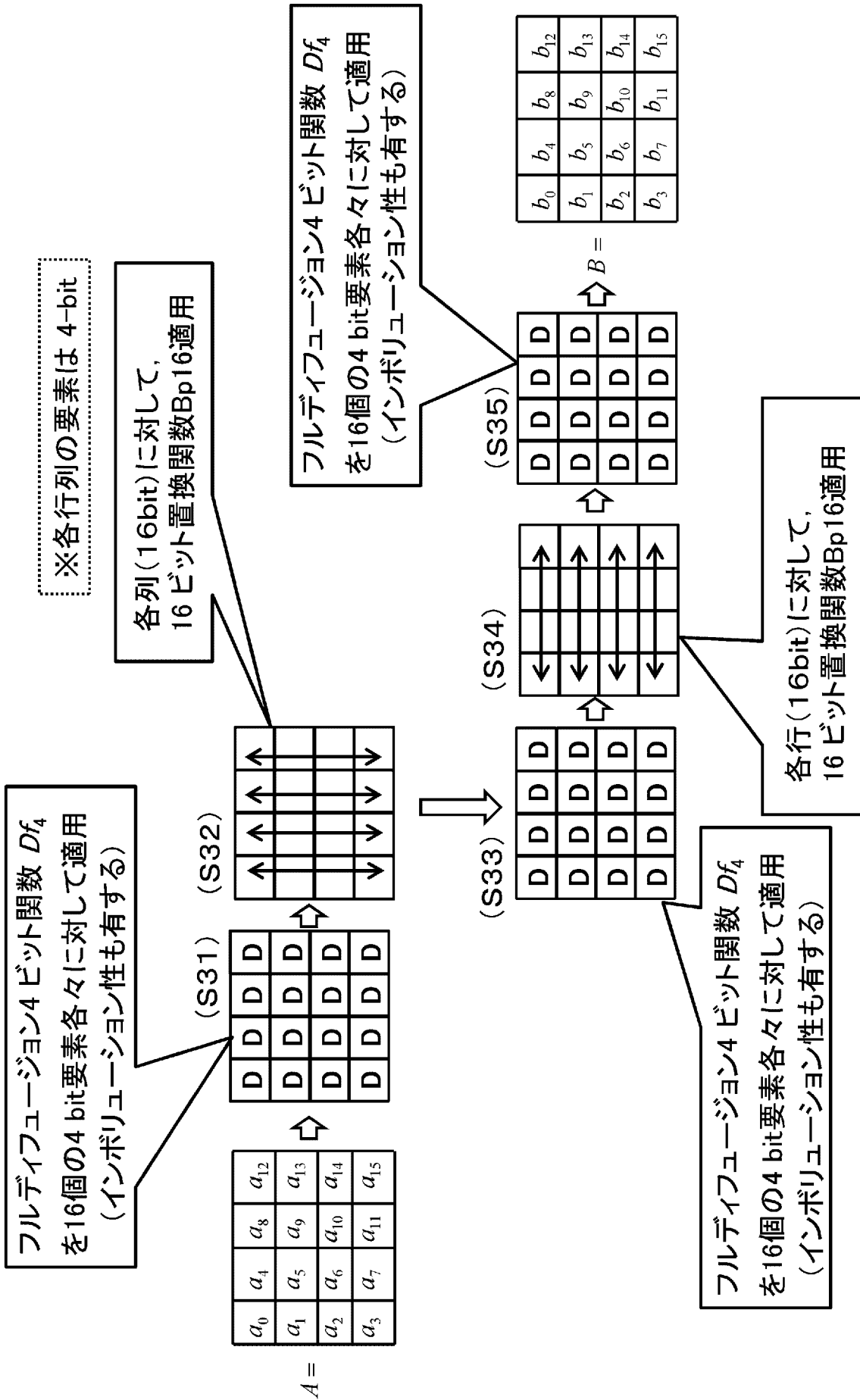
[図47]



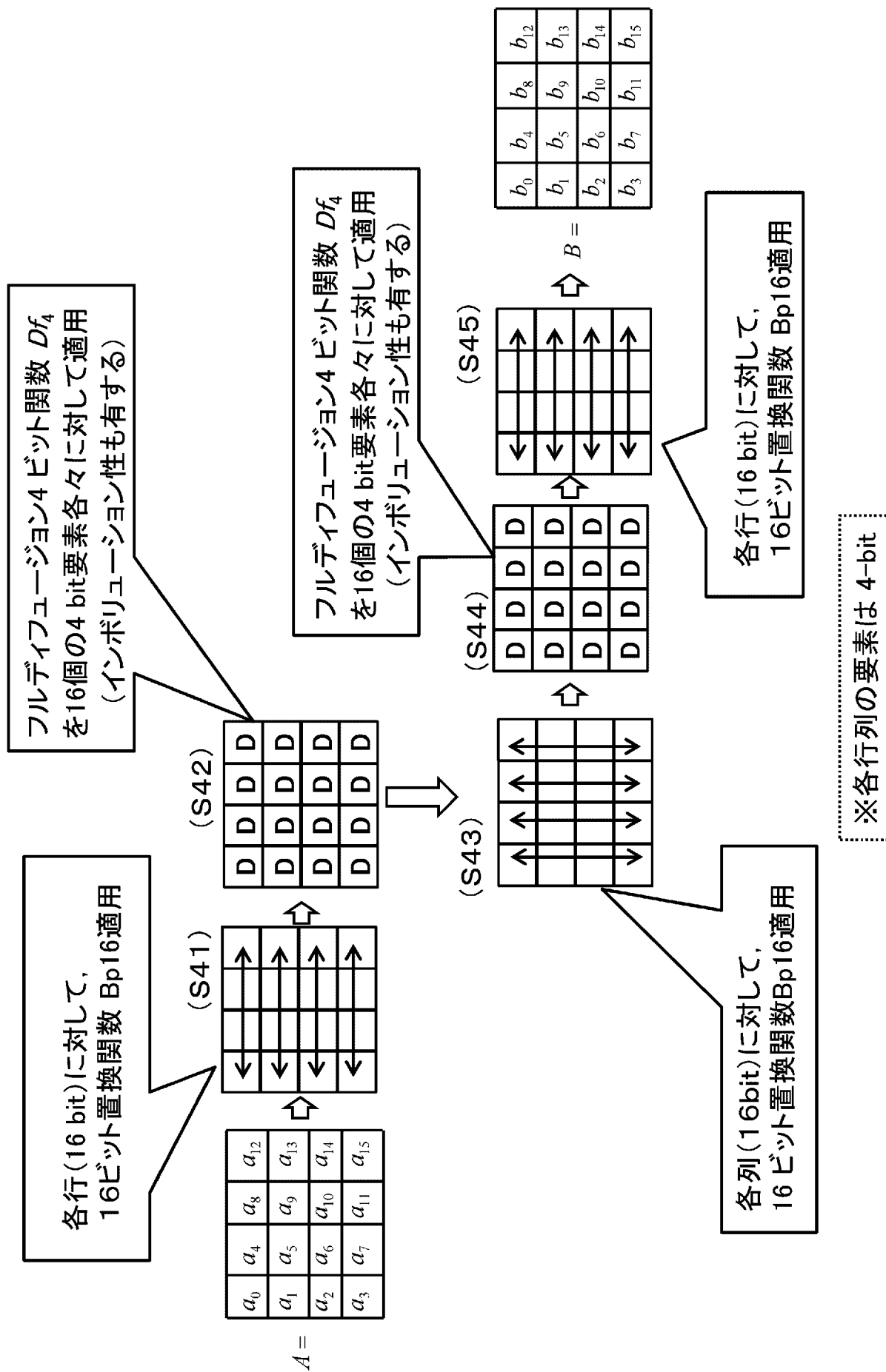
[図48]



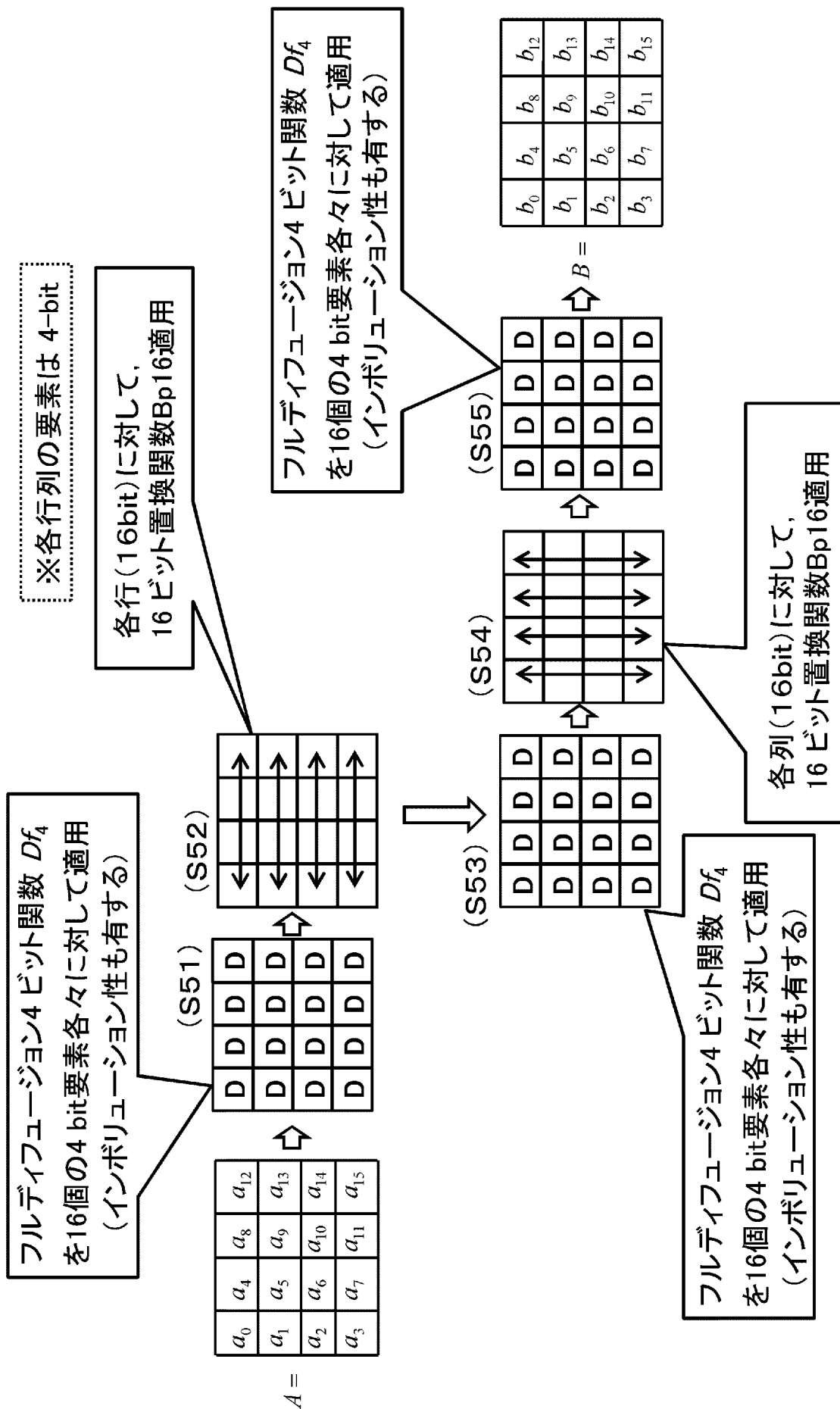
[図49]



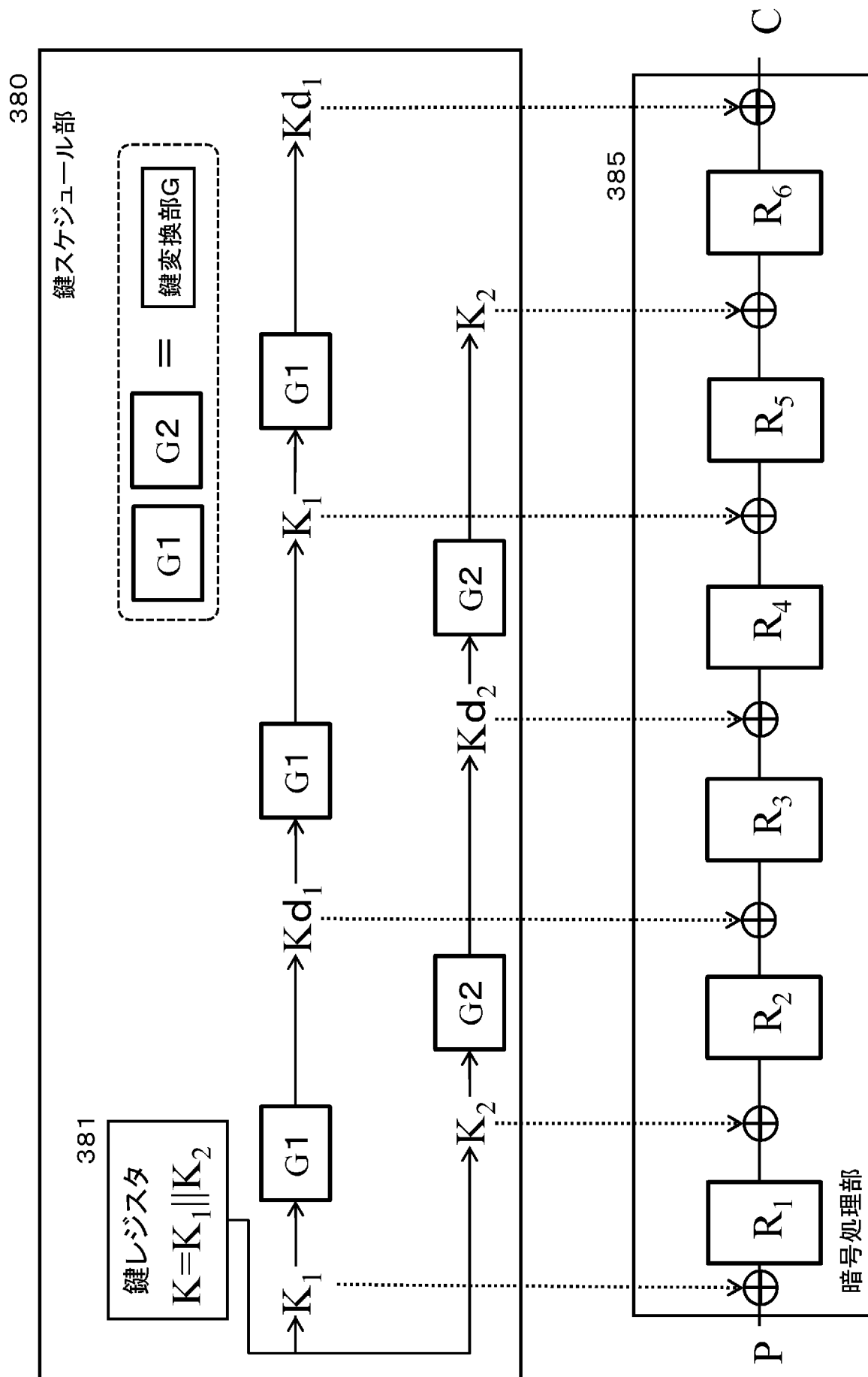
[図50]



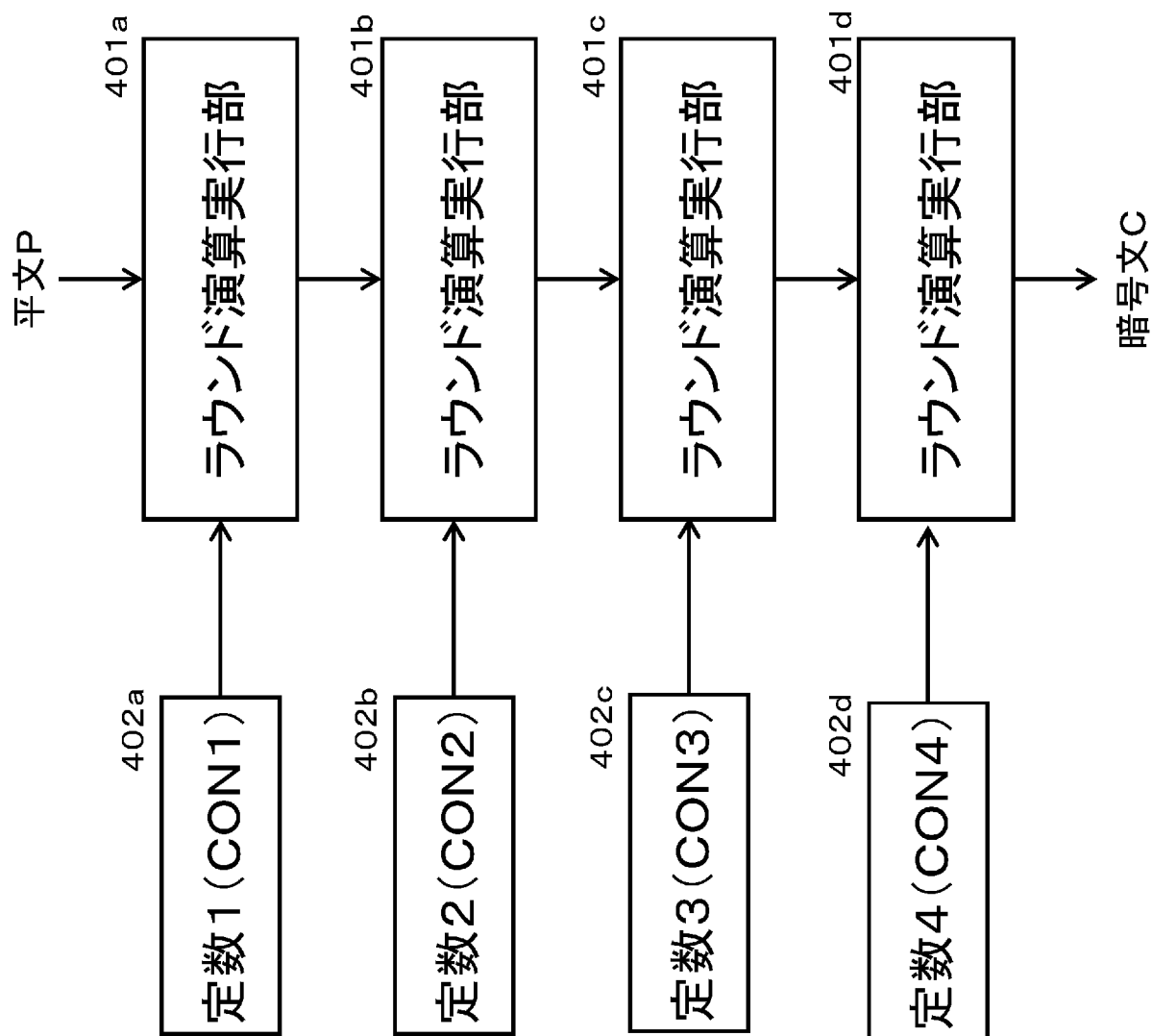
[図51]



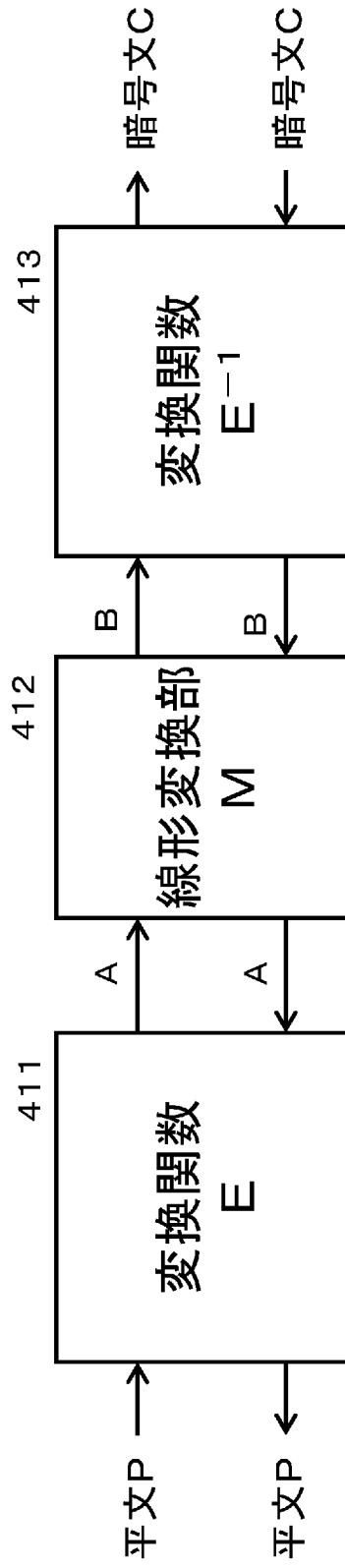
[図52]



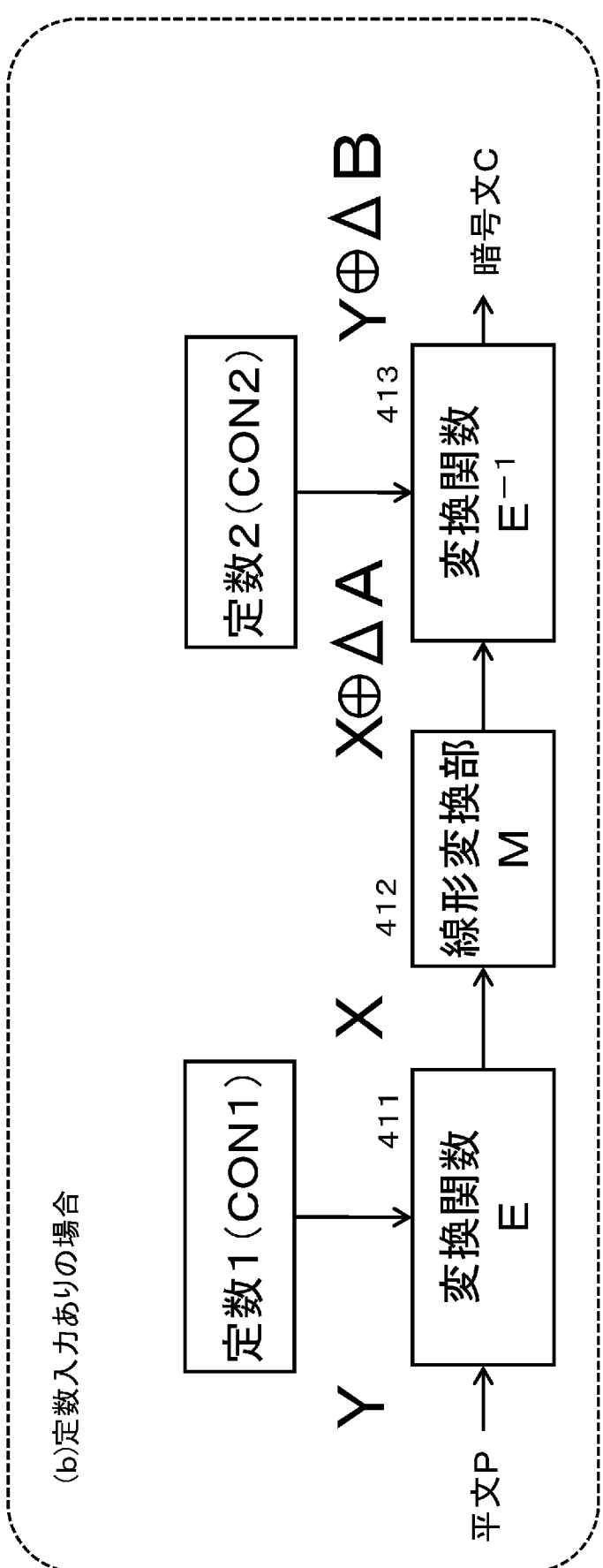
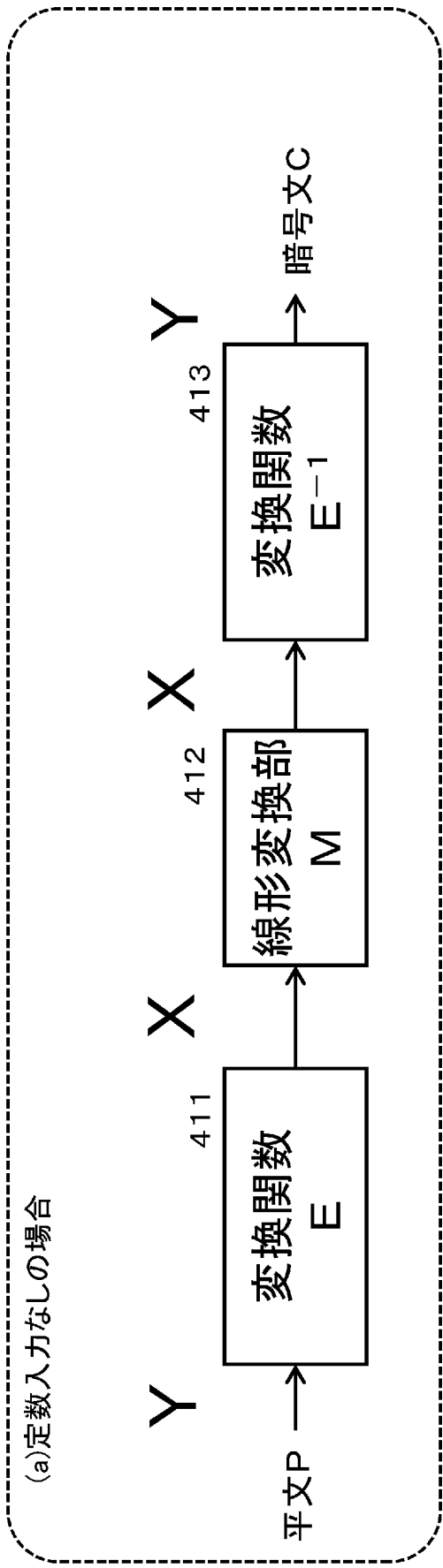
[図53]



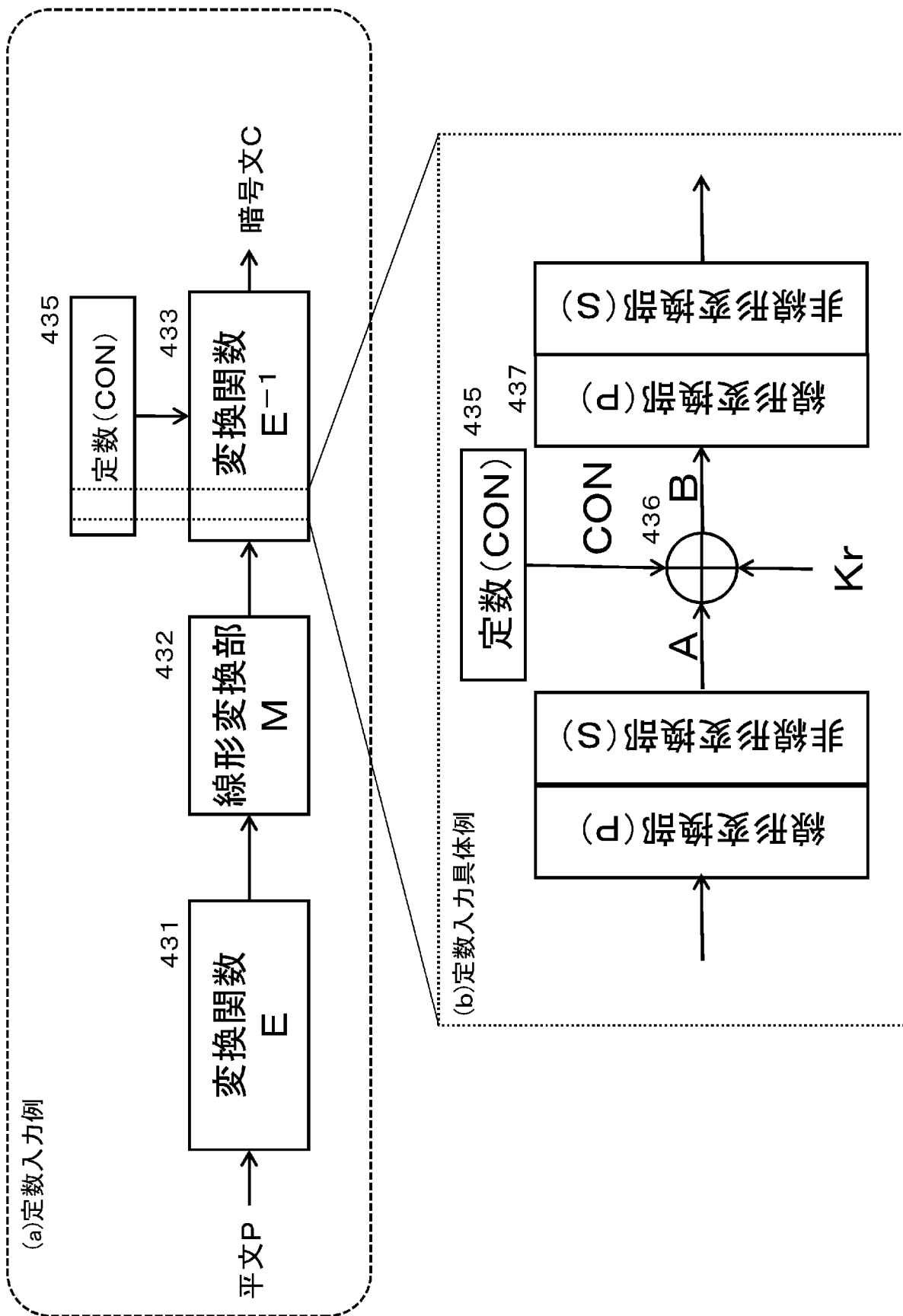
[図54]



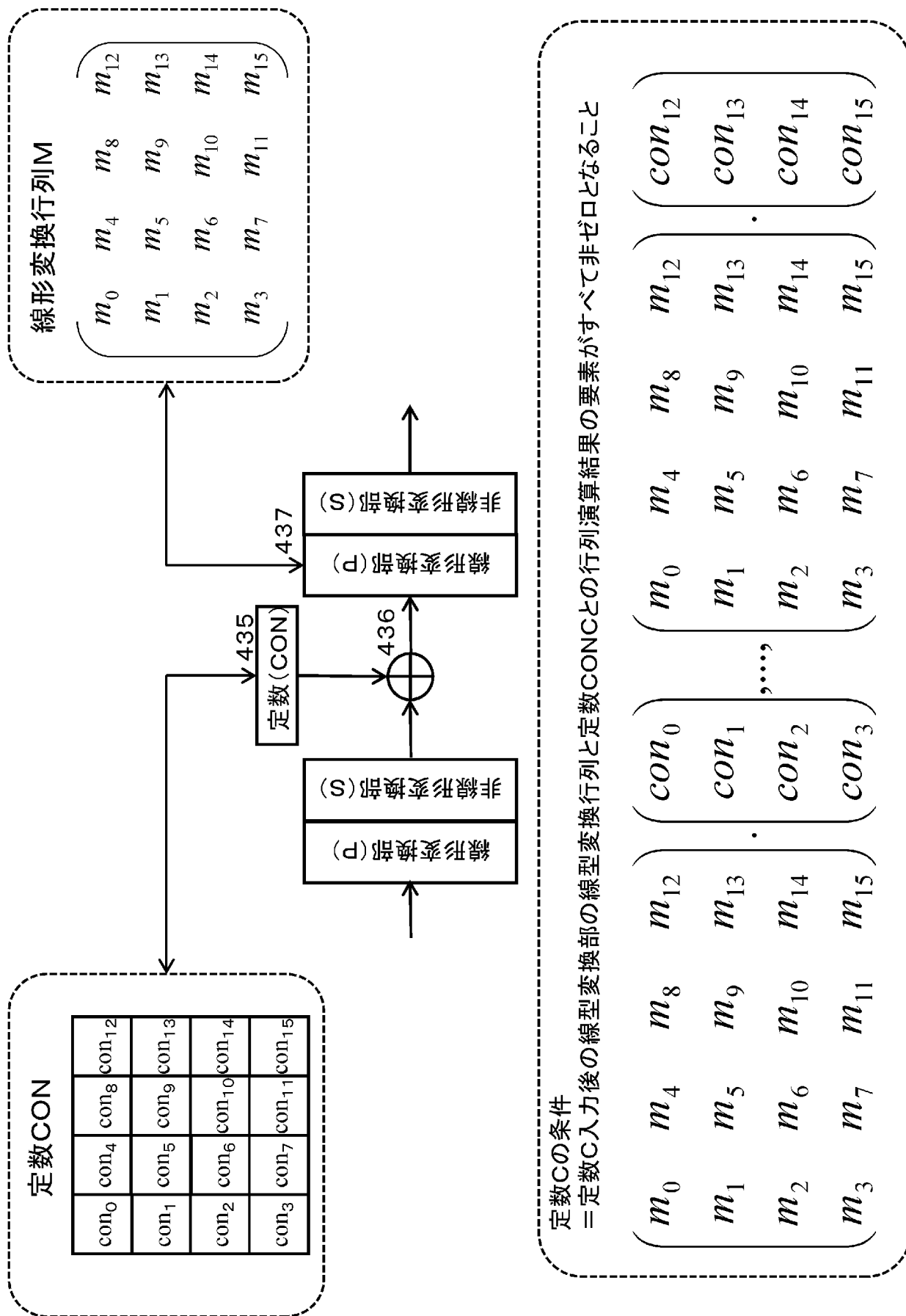
[図55]



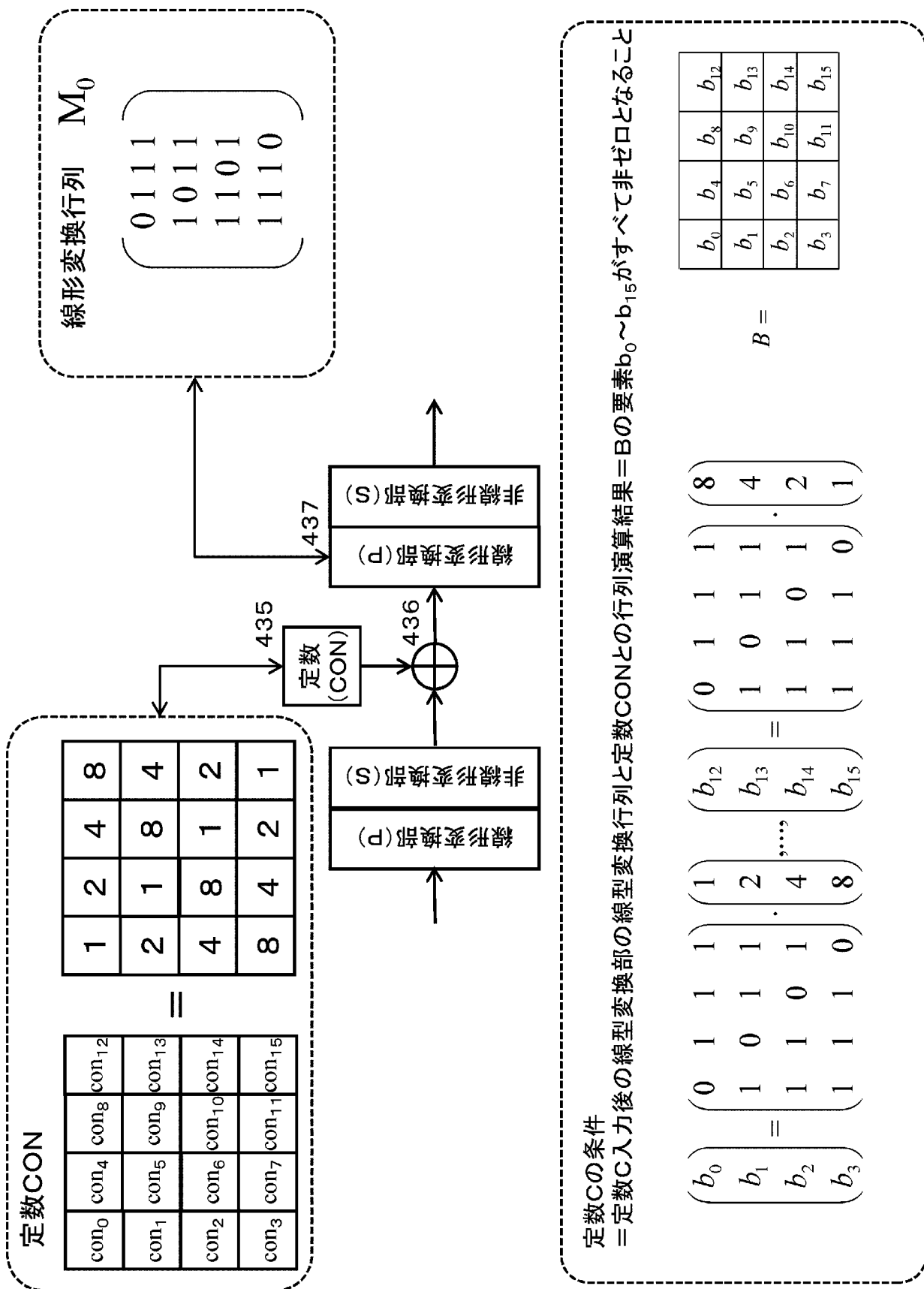
[図56]



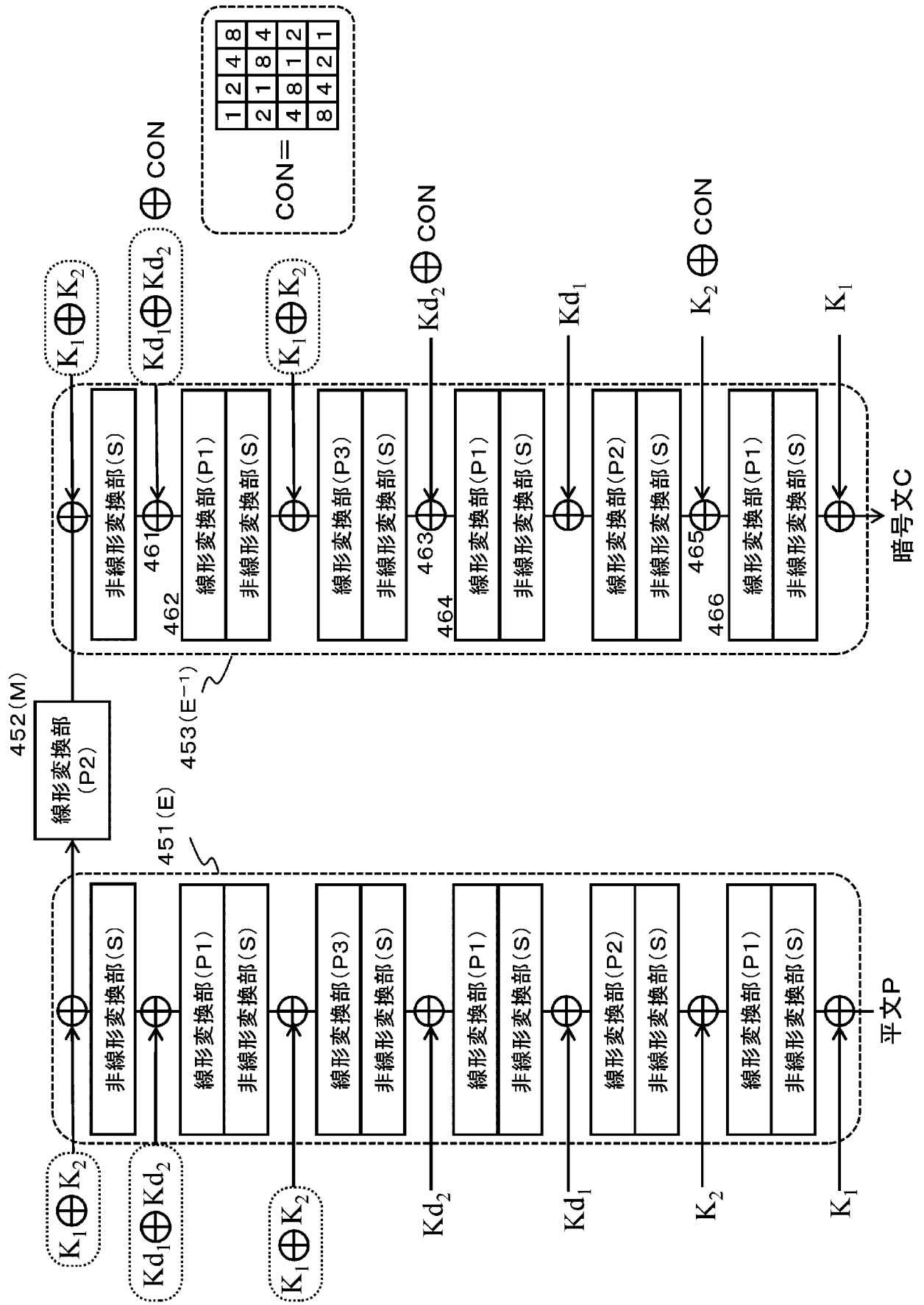
[図57]



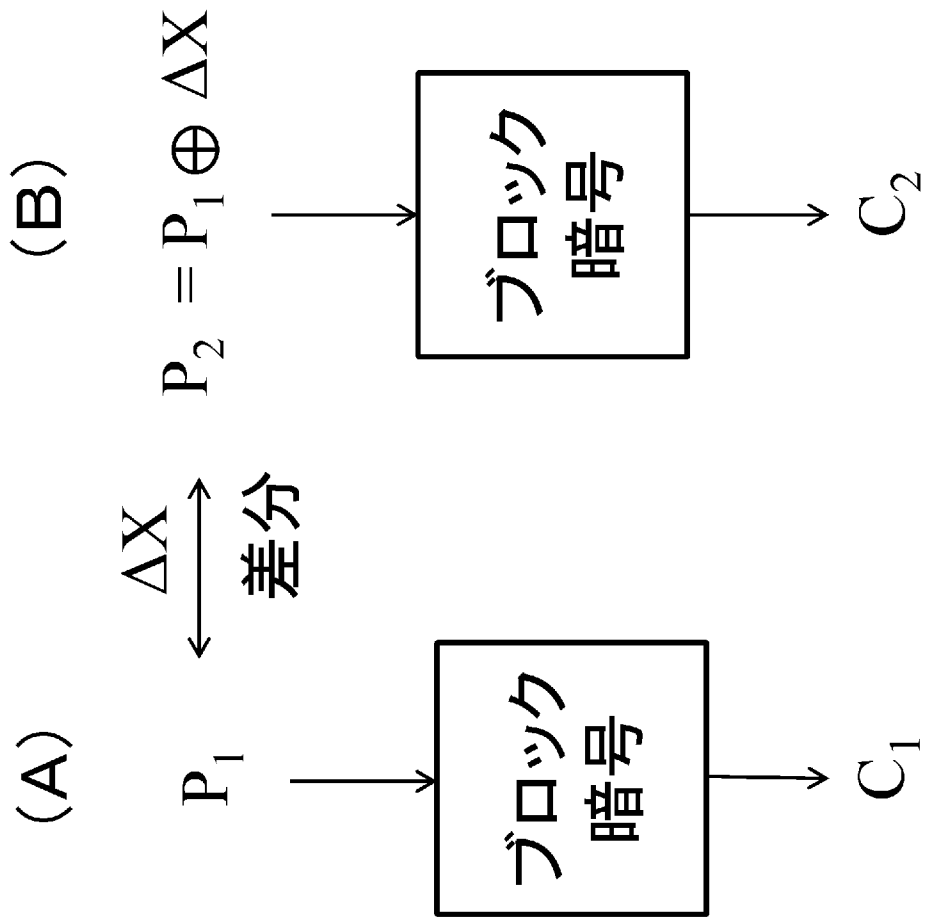
[図58]



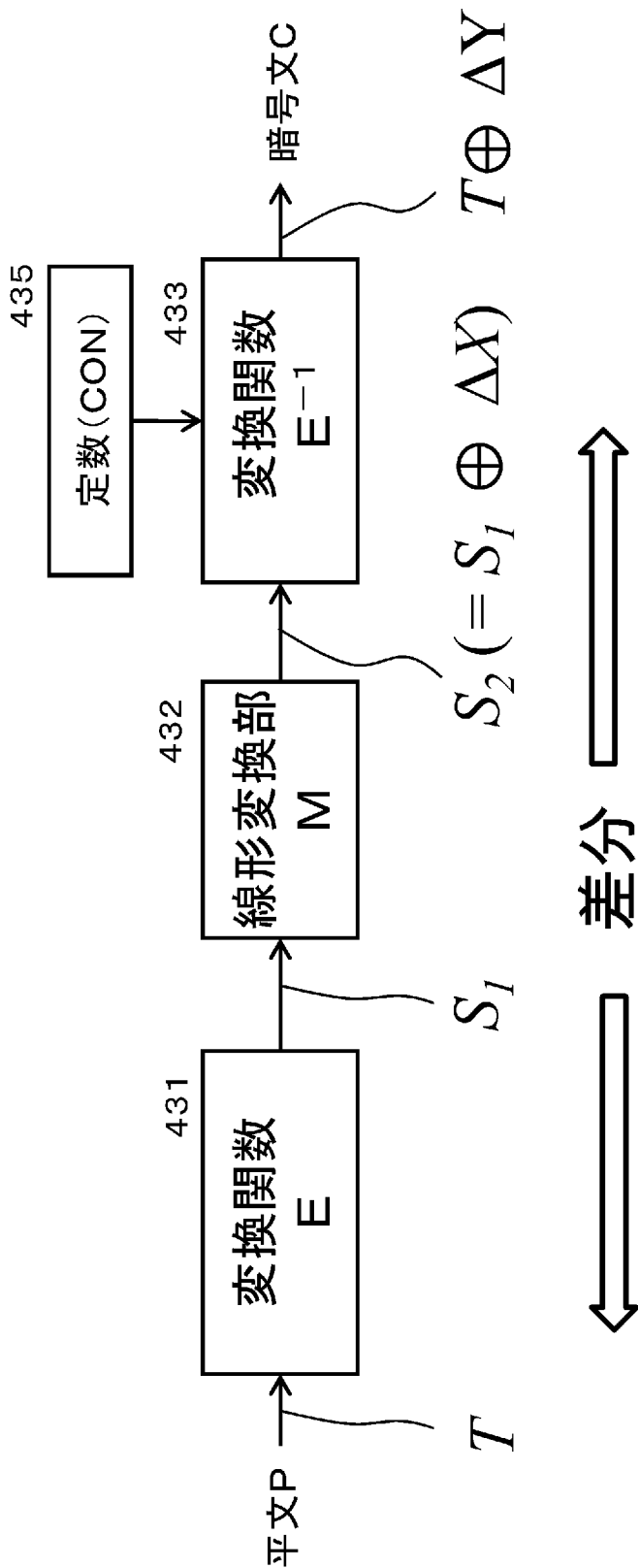
[図59]



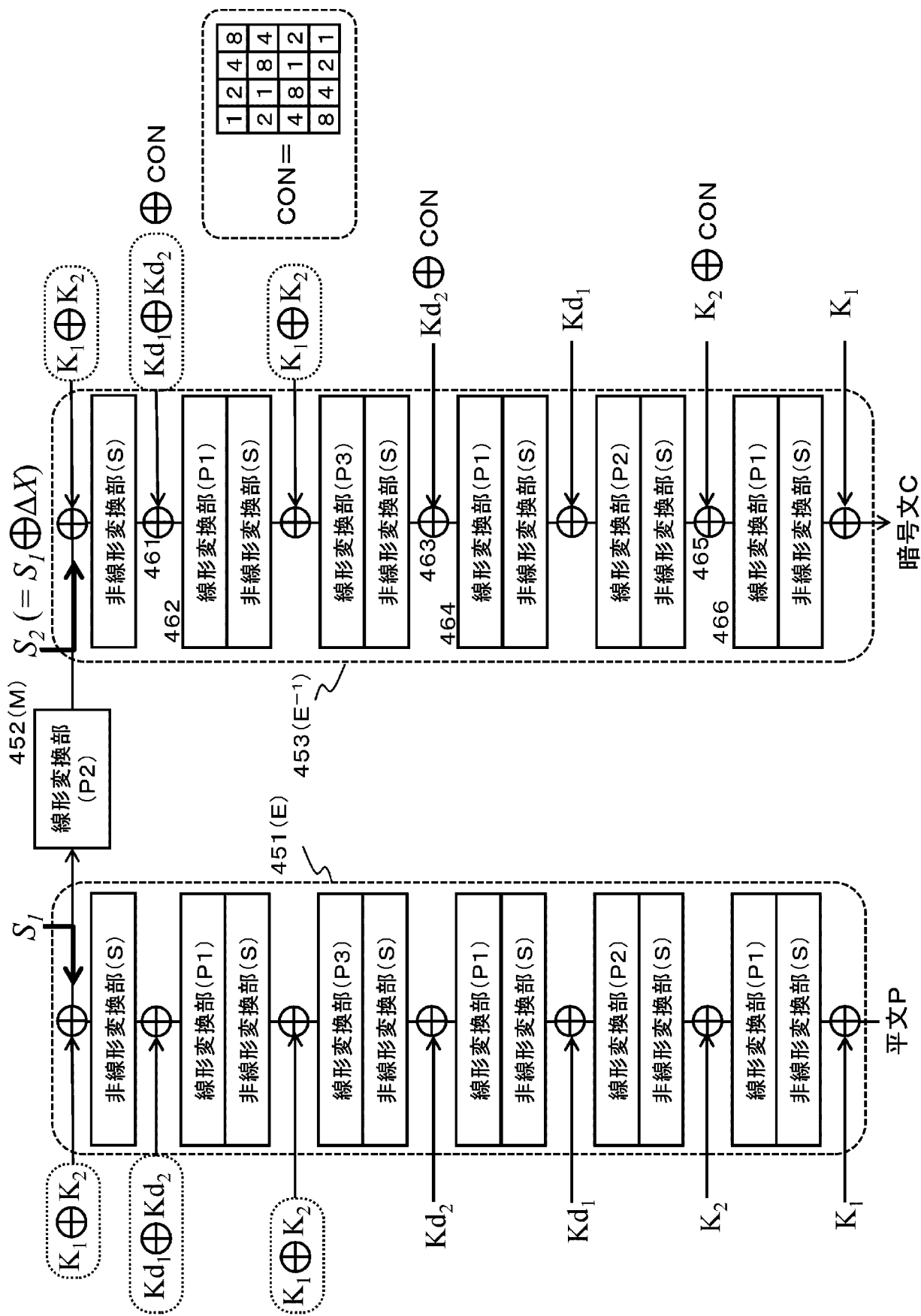
[図60]



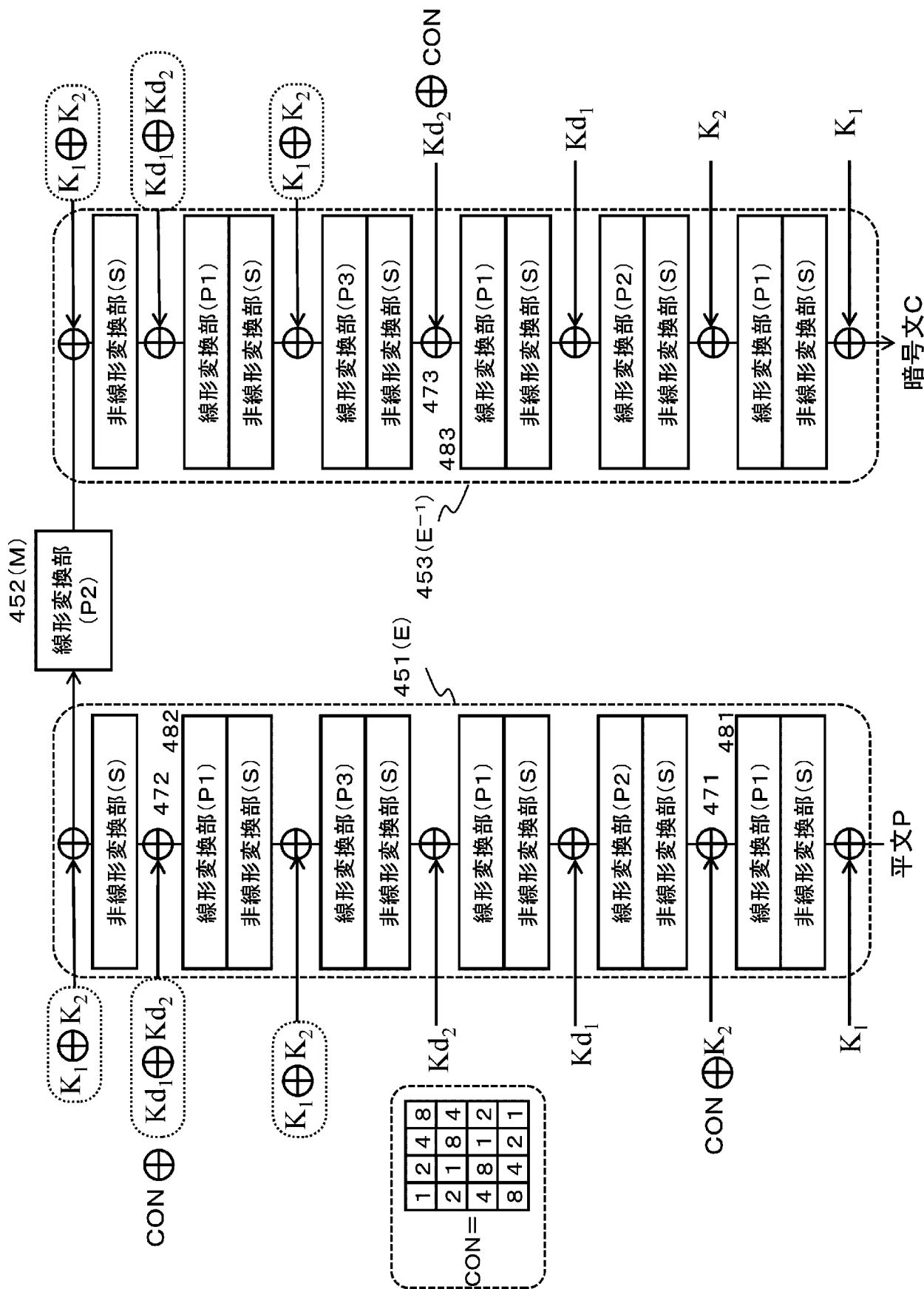
[図61]



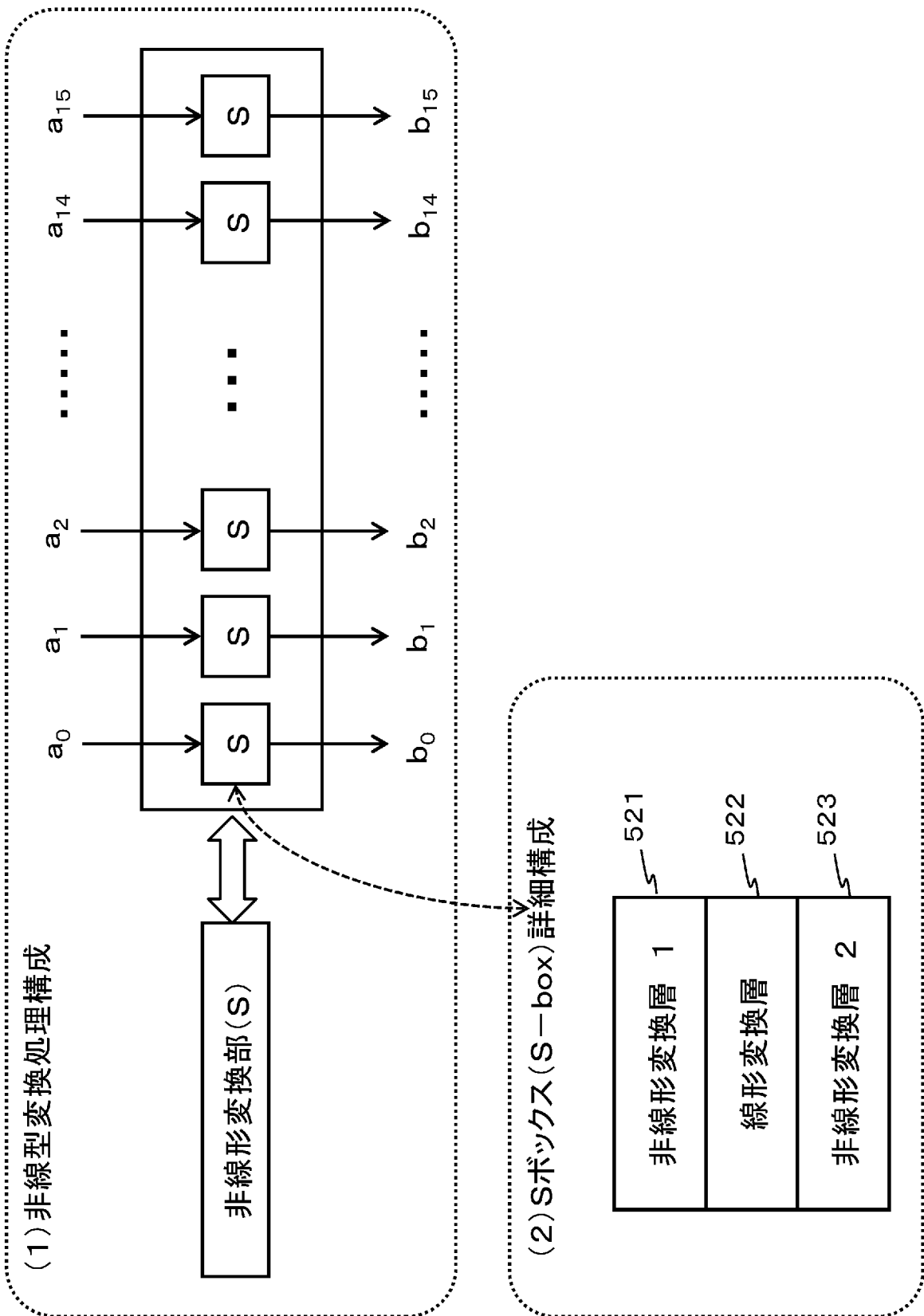
[図62]



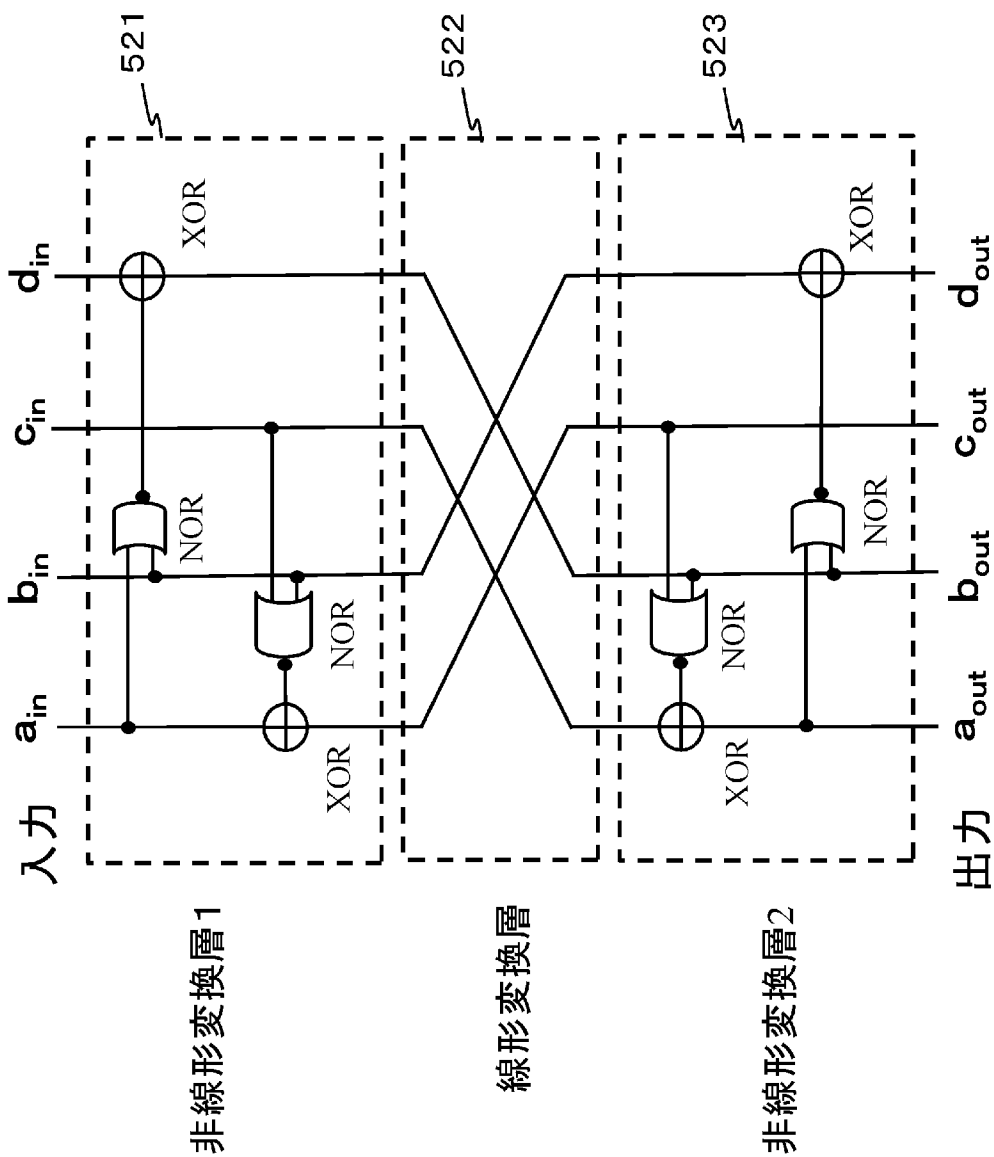
[図63]



[図64]

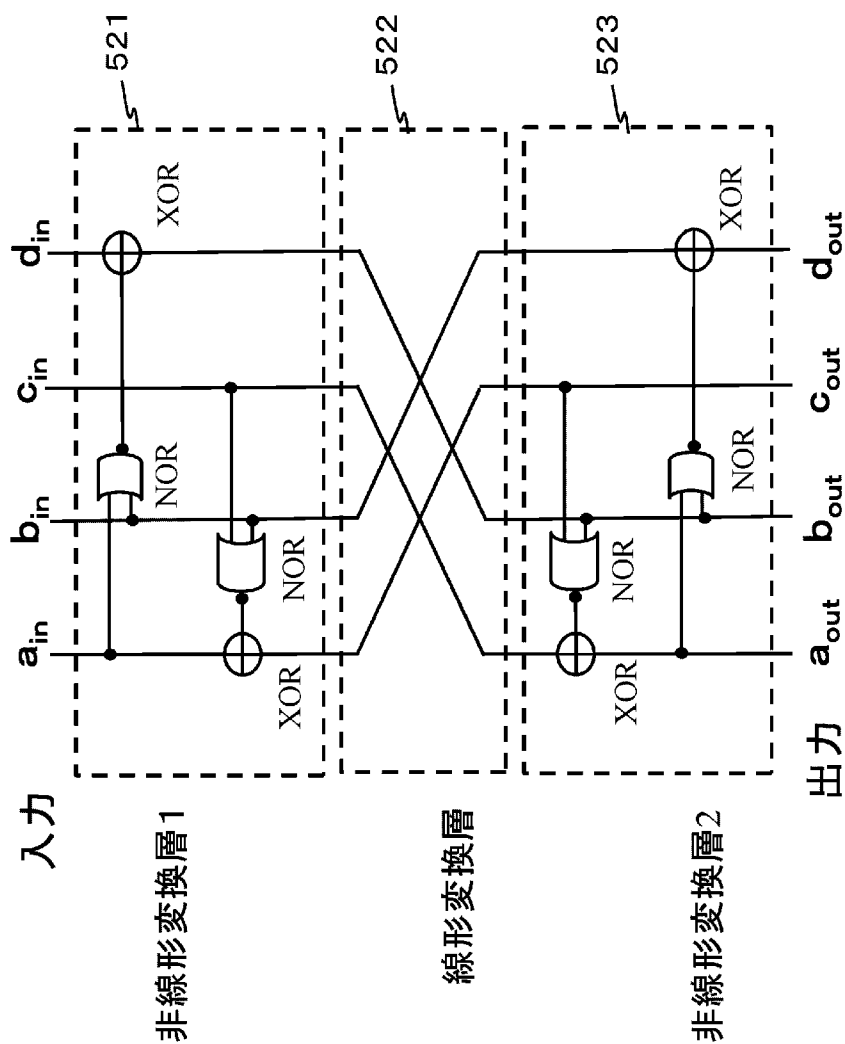


[圖65]



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]



\oplus : 排他的論理和 (XOR)
 \sim : () の値の否定 (NOT)

$$a_{out} = c_{in} \oplus \sim((d_{in} \oplus \sim(a_{in} | b_{in}))) | (a_{in} \oplus \sim(b_{in} | c_{in})))$$

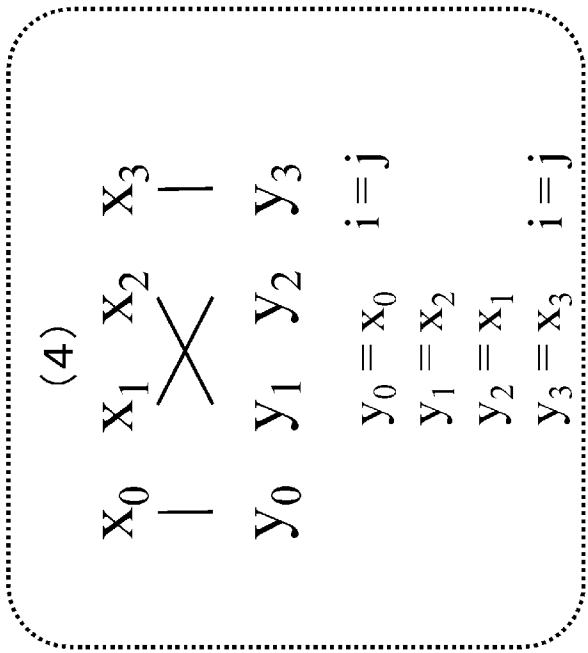
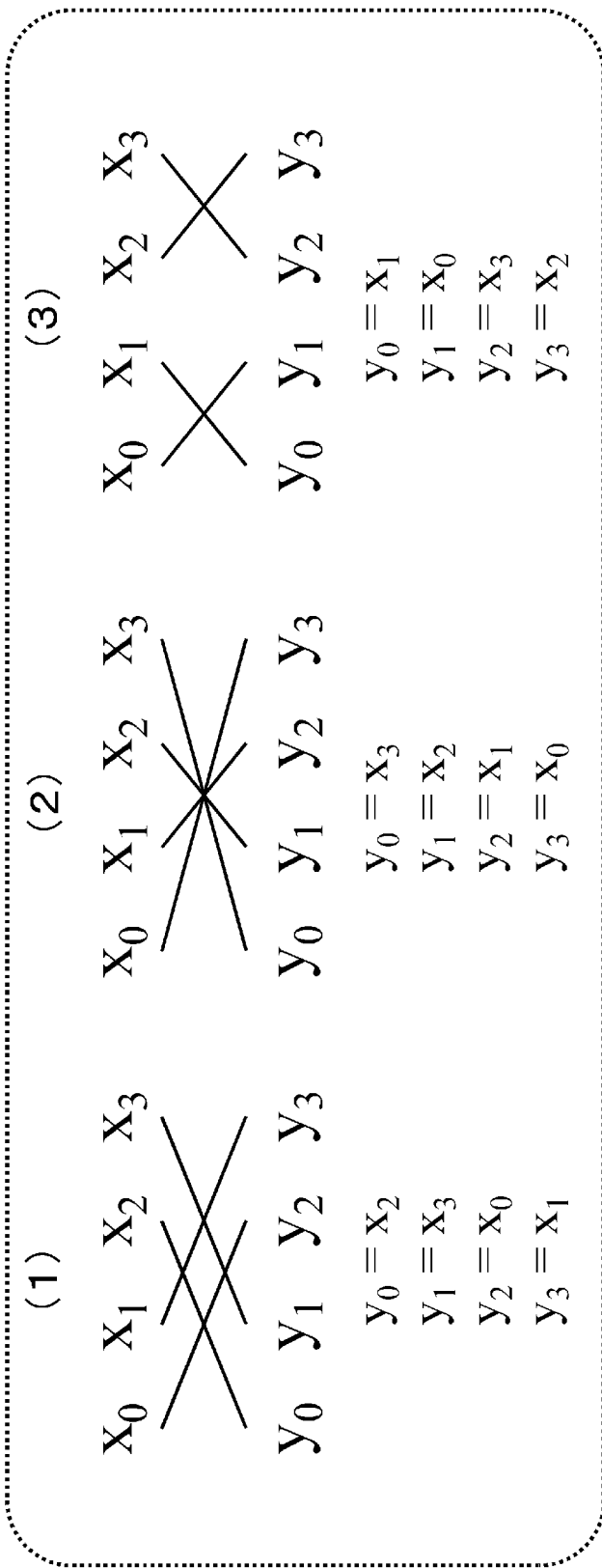
$$b_{out} = d_{in} \oplus \sim(a_{in} | b_{in})$$

$$c_{out} = a_{in} \oplus \sim(b_{in} | c_{in})$$

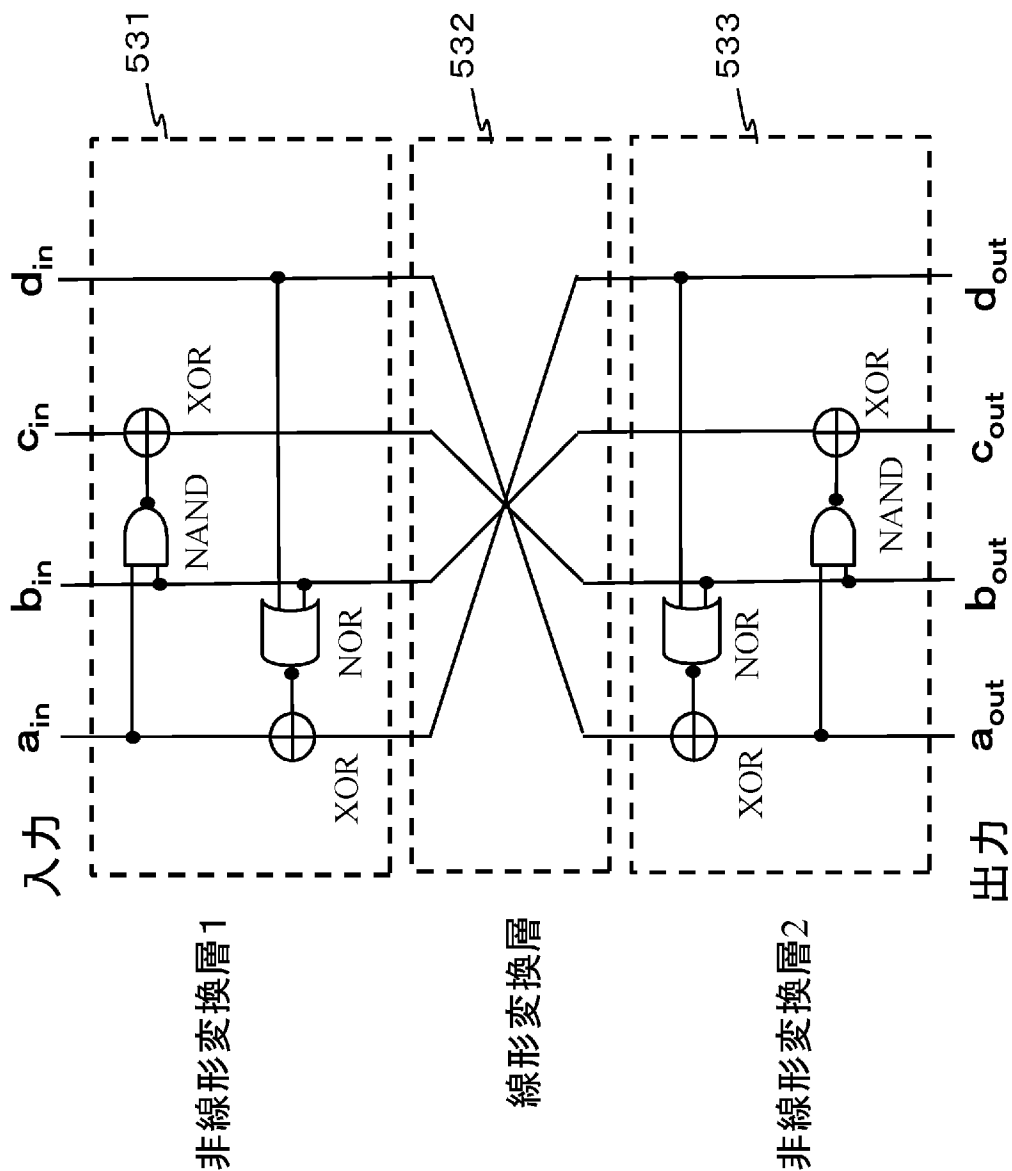
$$d_{out} = b_{in} \oplus \sim(b_{out} | a_{out})$$

$$= b_{in} \oplus \sim((d_{in} \oplus \sim(a_{in} | b_{in}))) | (c_{in} \oplus \sim((d_{in} \oplus \sim(a_{in} | b_{in}))) | (a_{in} \oplus \sim(b_{in} | c_{in}))))))$$

[図67]

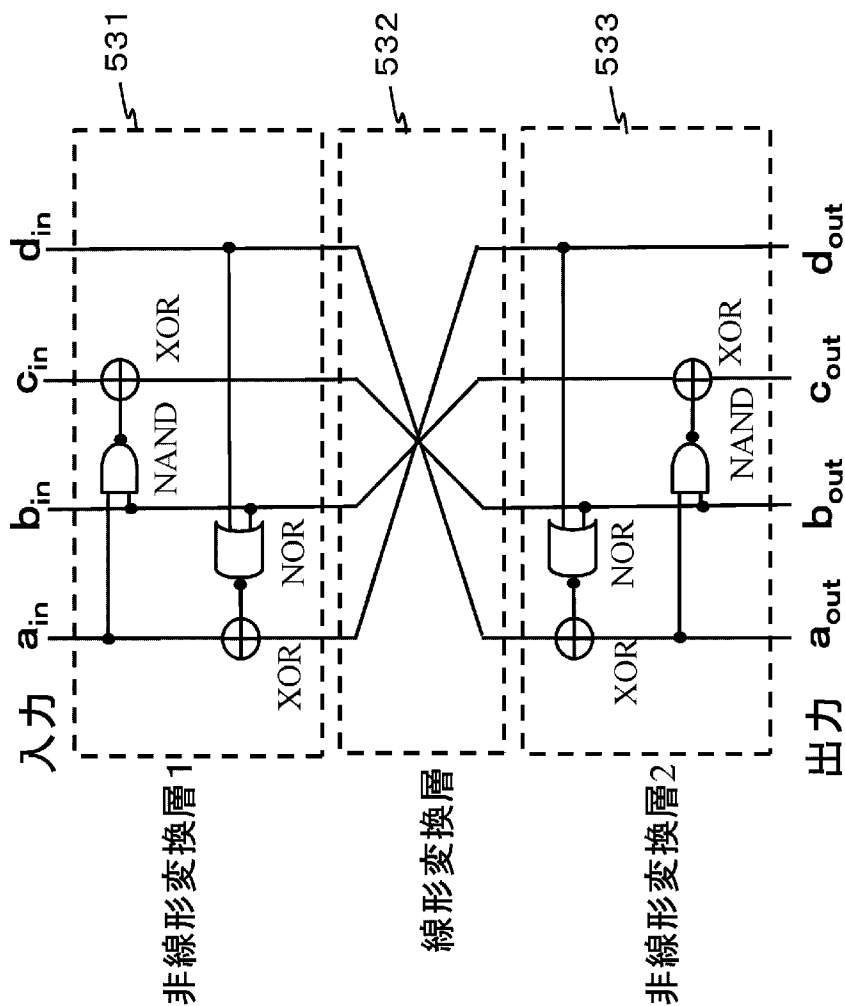


[圖68]



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

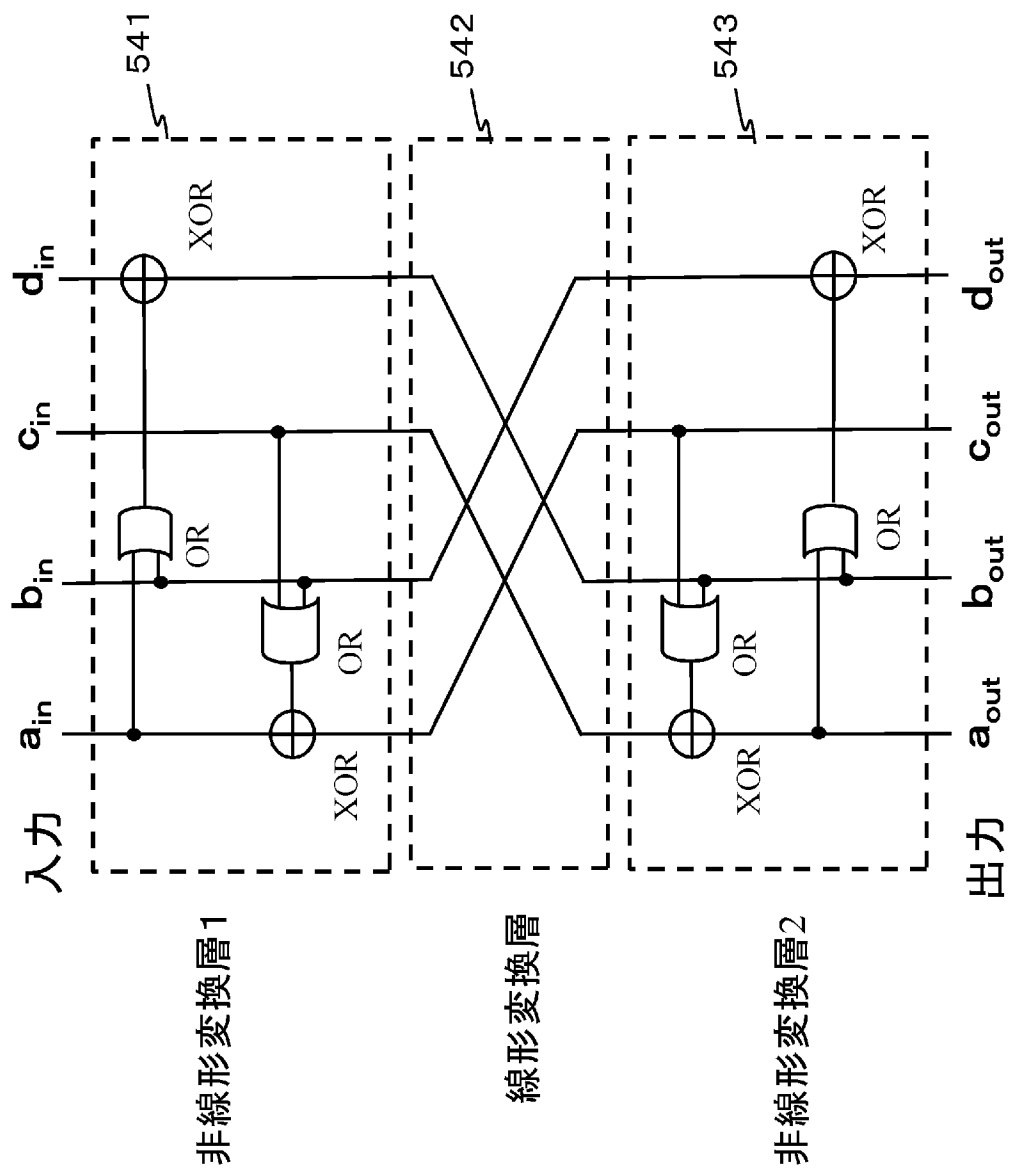
[図69]



$$\begin{aligned}
 a_{out} &= d_{in} \oplus \sim(b_{out} | d_{out}) = d_{in} \oplus \sim((c_{in} \oplus (a_{in} \& b_{in})) | (a_{in} \oplus \sim(b_{in} | d_{in}))) \\
 b_{out} &= c_{in} \oplus \sim(a_{in} \& b_{in}) \\
 c_{out} &= b_{in} \oplus (a_{out} \& b_{out}) \\
 &= b_{in} \oplus ((d_{in} \oplus \sim((c_{in} \oplus (a_{in} \& b_{in})) | (a_{in} \oplus \sim(b_{in} | d_{in})))) \& (c_{in} \oplus \sim(a_{in} \& b_{in}))) \\
 d_{out} &= a_{in} \oplus \sim(b_{in} | d_{in})
 \end{aligned}$$

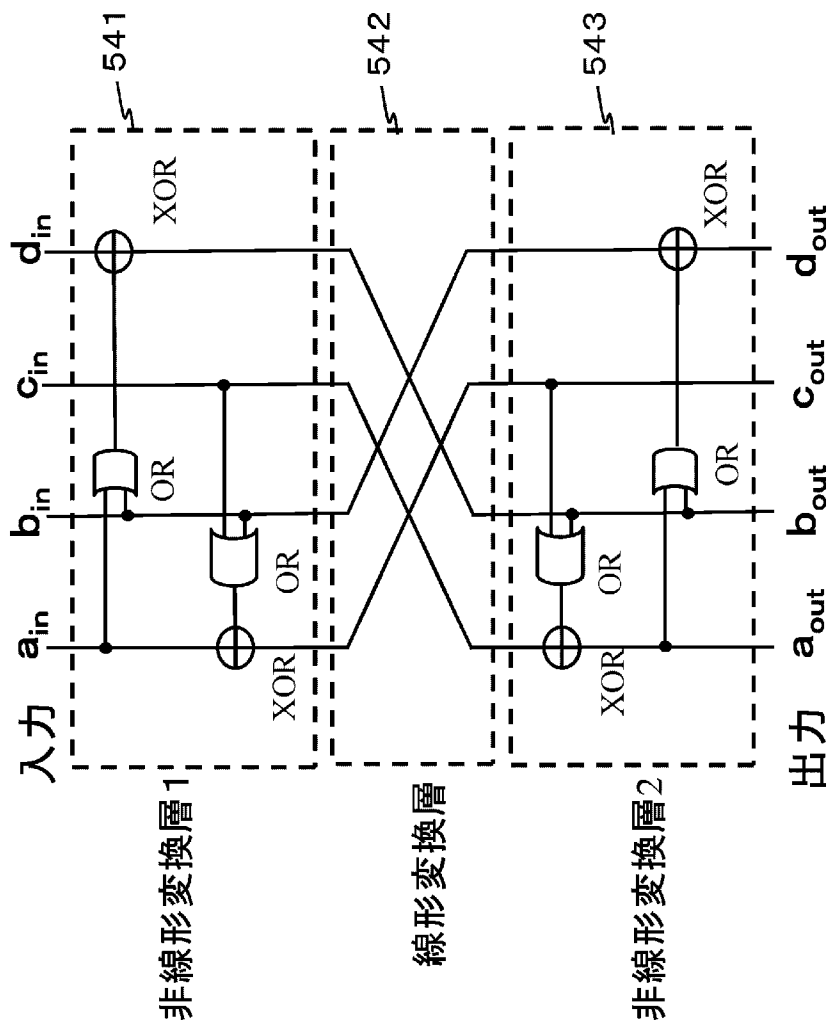
\oplus : 排他的論理和 (XOR) \sim : () の値の否定 (NOT)

[圖70]



輸入值(in)	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
輸出值(out)	0	13	2	7	14	10	6	3	15	11	5	9	12	1	4	8

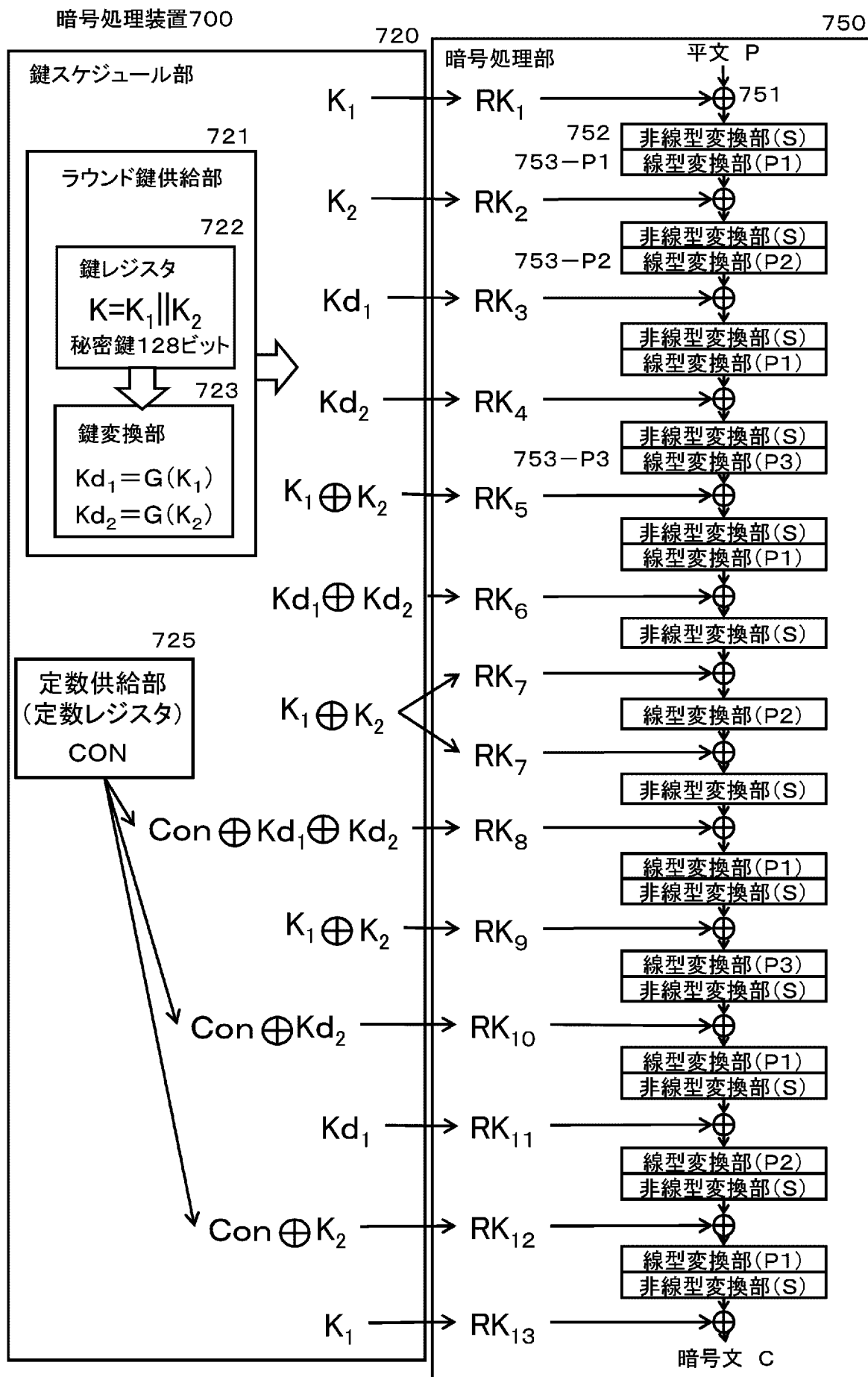
[圖71]



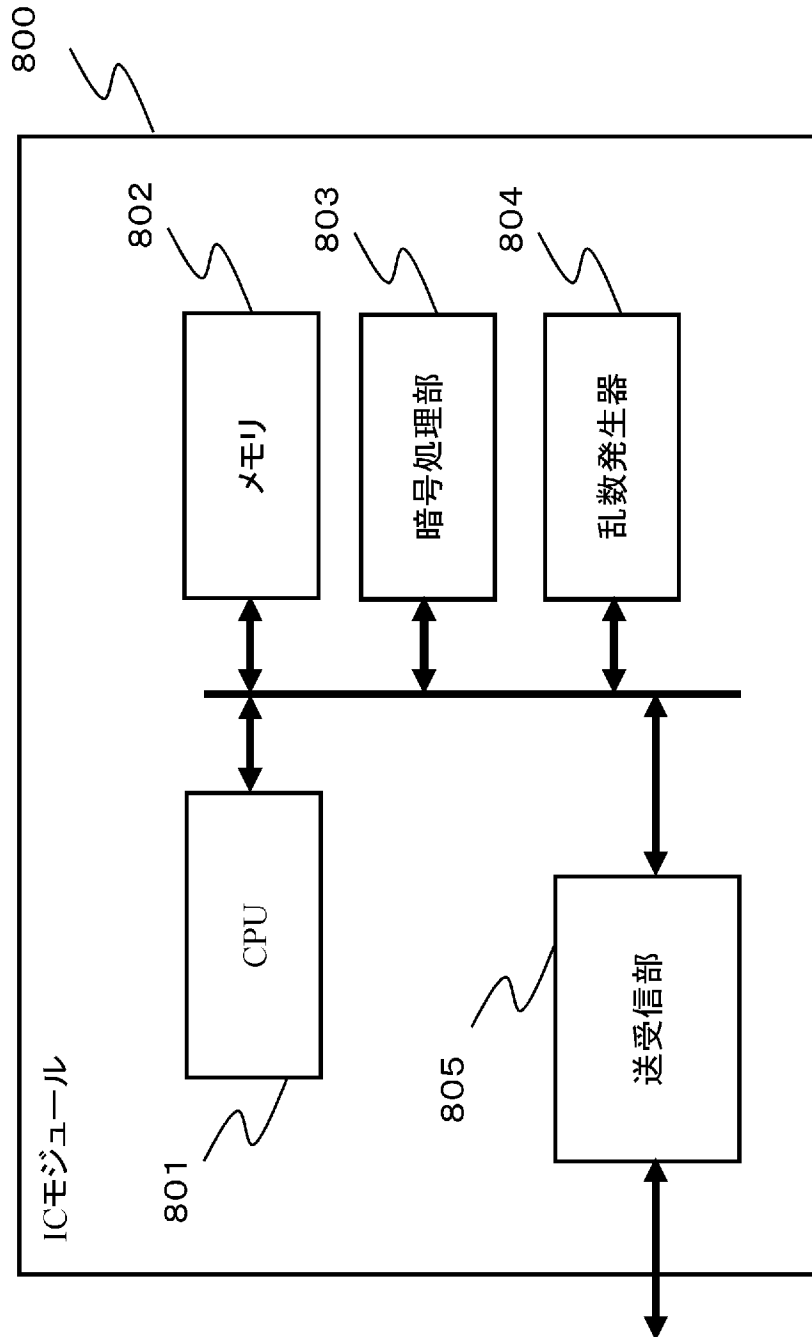
$$\begin{aligned}
 a_{out} &= c_{in} \oplus ((d_{in} \oplus (a_{in} | b_{in})) | (a_{in} \oplus (b_{in} | c_{in}))) \\
 b_{out} &= d_{in} \oplus (a_{in} | b_{in}) \\
 c_{out} &= a_{in} \oplus (b_{in} | c_{in}) \\
 d_{out} &= b_{in} \oplus (b_{out} | a_{out}) \\
 &= b_{in} \oplus (((d_{in} \oplus (a_{in} | b_{in})) | (c_{in} \oplus ((d_{in} \oplus (a_{in} | b_{in})) | (a_{in} \oplus (b_{in} | c_{in}))))))
 \end{aligned}$$

\oplus :排他的論理和(XOR)

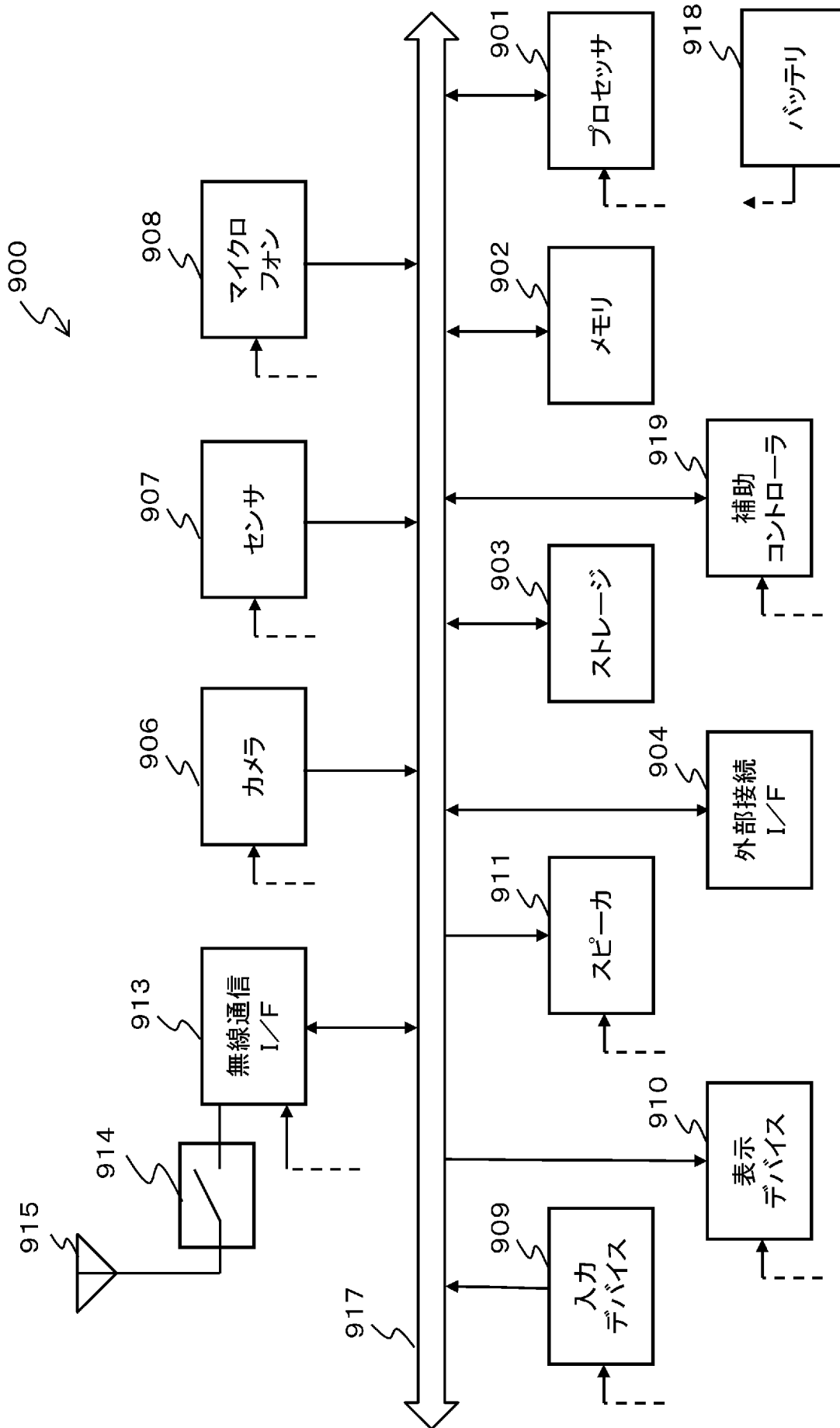
[図72]



[図73]



[図74]



INTERNATIONAL SEARCH REPORT

International application No.

PCT/JP2015/055280

A. CLASSIFICATION OF SUBJECT MATTER

G09C1/00(2006.01)i, H04L9/06(2006.01)i

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)

G09C1/00, H04L9/06

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Jitsuyo Shinan Koho	1922-1996	Jitsuyo Shinan Toroku Koho	1996-2015
Kokai Jitsuyo Shinan Koho	1971-2015	Toroku Jitsuyo Shinan Koho	1994-2015

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)

IEEE Xplore, JSTPlus(JDreamIII), block cipher, involution, diffusion

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
Y A	JP 2000-66586 A (Toshiba Corp.), 03 March 2000 (03.03.2000), paragraphs [0022] to [0069]	1, 6-12, 17 2-5, 13-16, 18
Y	JP 2012-70048 A (Toshiba Corp.), 05 April 2012 (05.04.2012), paragraphs [0030] to [0032]; fig. 8	1, 6-12, 17
Y	WO 2008/026625 A1 (Sony Corp.), 06 March 2008 (06.03.2008), paragraphs [0130] to [0148]; fig. 24 to 26	6, 7
Y	JP 2002-23622 A (Toshiba Corp.), 23 January 2002 (23.01.2002), paragraphs [0035] to [0045], [0057] to [0068], [0088]	9

Further documents are listed in the continuation of Box C.

See patent family annex.

* Special categories of cited documents:

“A” document defining the general state of the art which is not considered to be of particular relevance

“E” earlier application or patent but published on or after the international filing date

“L” document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)

“O” document referring to an oral disclosure, use, exhibition or other means

“P” document published prior to the international filing date but later than the priority date claimed

“T” later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

“X” document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

“Y” document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art

“&” document member of the same patent family

Date of the actual completion of the international search
22 May 2015 (22.05.15)

Date of mailing of the international search report
02 June 2015 (02.06.15)

Name and mailing address of the ISA/
Japan Patent Office
3-4-3, Kasumigaseki, Chiyoda-ku,
Tokyo 100-8915, Japan

Authorized officer

Telephone No.

INTERNATIONAL SEARCH REPORT

International application No.

PCT/JP2015/055280

C (Continuation). DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
Y	WO 2009/087972 A1 (NEC Corp., NEC Software Hokuriku, Ltd.), 16 July 2009 (16.07.2009), paragraphs [0088] to [0095]; fig. 12, 13	10
Y	JP 2005-107078 A (Sony Corp.), 21 April 2005 (21.04.2005), paragraphs [0103] to [0121]; fig. 9	11
Y	Wang, C. and Heys, H. M., AN ULTRA COMPACT BLOCK CIPHER FOR SERIALIZED ARCHITECTURE IMPLEMENTATIONS, 2009 Canadian Conference on Electrical and Computer Engineering, p.1085-1090, 2009, especially 2.CIPHER SPECIFICATION	12
Y A	Barreto, P. S.L.M. and Rijmen, V., The ANUBIS Block Cipher, First Open NESSIE Workshop [online], 2000.11, [retrieved on 2015.05.21] Retrieved from the Internet: <URL:https://www.cosic.esat.kuleuven.be/nessie/workshop/submissions/anubis.zip, anubis/B/Anubis.pdf> especially 3 Description of the ANUBIS primitive	12 2-5, 13-16, 18
A	Borghoff, J. et al., PRINCE - A Low-latency Block Cipher for Pervasive Computing Applications, Cryptology ePrint Archive [online], Report 2012/529, 2012.09.13, [retrieved on 2015.05.21] Retrieved from the Internet: <URL: https://eprint.iacr.org/2012/529/20120913:093817> especially 2.CIPHER SPECIFICATION	1-18

INTERNATIONAL SEARCH REPORT
Information on patent family members

International application No.

PCT/JP2015/055280

JP 2000-66586 A	2000.03.03	US 6819764 B1	2004.11.16
		EP 982895 A2	2000.03.01
JP 2012-70048 A	2012.04.05	US 2012/0069997 A1	2012.03.22
WO 2008/026625 A1	2008.03.06	JP 2008-58831 A	2008.03.13
		US 2010/0008498 A1	2010.01.14
		EP 2058784 A1	2009.05.13
		CN 101512619 A	2009.08.19
		TW 200828196 A	2008.07.01
JP 2002-23622 A	2002.01.23	US 2002/0021802 A1	2002.02.21
		EP 1172965 A2	2002.01.16
		KR 10-2002-0006475 A	2002.01.19
		CN 1349182 A	2002.05.15
WO 2009/087972 A1	2009.07.16	US 2011/0110519 A1	2011.05.12
JP 2005-107078 A	2005.04.21	US 2005/0111659 A1	2005.05.26

INTERNATIONAL SEARCH REPORT

International application No.

PCT/JP2015/055280

Box No. II Observations where certain claims were found unsearchable (Continuation of item 2 of first sheet)

This international search report has not been established in respect of certain claims under Article 17(2)(a) for the following reasons:

1. Claims Nos.:
because they relate to subject matter not required to be searched by this Authority, namely:

2. Claims Nos.:
because they relate to parts of the international application that do not comply with the prescribed requirements to such an extent that no meaningful international search can be carried out, specifically:

3. Claims Nos.:
because they are dependent claims and are not drafted in accordance with the second and third sentences of Rule 6.4(a).

Box No. III Observations where unity of invention is lacking (Continuation of item 3 of first sheet)

This International Searching Authority found multiple inventions in this international application, as follows:

The inventions according to the claims of the application are divided into the following two groups.

1. Claims 1-12, 17

An encryption processing device wherein a conversion function used by a key scheduling unit has an involution property

2. Claims 13-16, 18

An encryption processing device wherein a conversion function used by a key scheduling unit has a full-diffusion property

1. As all required additional search fees were timely paid by the applicant, this international search report covers all searchable claims.
2. As all searchable claims could be searched without effort justifying additional fees, this Authority did not invite payment of additional fees.
3. As only some of the required additional search fees were timely paid by the applicant, this international search report covers only those claims for which fees were paid, specifically claims Nos.:

4. No required additional search fees were timely paid by the applicant. Consequently, this international search report is restricted to the invention first mentioned in the claims; it is covered by claims Nos.:

Remark on Protest

- The additional search fees were accompanied by the applicant's protest and, where applicable, the payment of a protest fee.
- The additional search fees were accompanied by the applicant's protest but the applicable protest fee was not paid within the time limit specified in the invitation.
- No protest accompanied the payment of additional search fees.

A. 発明の属する分野の分類 (国際特許分類 (IPC)) Int.Cl. G09C1/00(2006.01)i, H04L9/06(2006.01)i		
B. 調査を行った分野 調査を行った最小限資料 (国際特許分類 (IPC)) Int.Cl. G09C1/00, H04L9/06		
最小限資料以外の資料で調査を行った分野に含まれるもの 日本国実用新案公報 1922-1996年 日本国公開実用新案公報 1971-2015年 日本国実用新案登録公報 1996-2015年 日本国登録実用新案公報 1994-2015年		
国際調査で使用した電子データベース (データベースの名称、調査に使用した用語) IEEE Xplore, JSTPlus(JDreamIII) Block cipher, key schedule, involution, diffusion		
C. 関連すると認められる文献		
引用文献の カテゴリー*	引用文献名 及び一部の箇所が関連するときは、その関連する箇所の表示	関連する 請求項の番号
Y A	JP 2000-66586 A (株式会社東芝) 2000.03.03, [0022]-[0069]	1,6-12, 17 2-5, 13-16, 18
Y	JP 2012-70048 A (株式会社東芝) 2012.04.05, [0030]-[0032], 図 8	1,6-12, 17
Y	WO 2008/026625 A1 (ソニー株式会社) 2008.03.06, [0130]-[0148], 図 24-26	6, 7
Y	JP 2002-23622 A (株式会社東芝) 2002.01.23, [0035]-[0045], [0057]-[0068], [0088]	9
<input checked="" type="checkbox"/> C 欄の続きにも文献が列挙されている。 <input checked="" type="checkbox"/> パテントファミリーに関する別紙を参照。		
* 引用文献のカテゴリー 「A」 特に関連のある文献ではなく、一般的技術水準を示すもの 「E」 国際出願日前の出願または特許であるが、国際出願日以後に公表されたもの 「L」 優先権主張に疑義を提起する文献又は他の文献の発行日若しくは他の特別な理由を確立するために引用する文献 (理由を付す) 「O」 口頭による開示、使用、展示等に言及する文献 「P」 国際出願日前で、かつ優先権の主張の基礎となる出願日の後に公表された文献 「T」 国際出願日又は優先日後に公表された文献であって出願と矛盾するものではなく、発明の原理又は理論の理解のために引用するもの 「X」 特に関連のある文献であって、当該文献のみで発明の新規性又は進歩性がないと考えられるもの 「Y」 特に関連のある文献であって、当該文献と他の1以上の文献との、当業者にとって自明である組合せによって進歩性がないと考えられるもの 「&」 同一パテントファミリー文献		
国際調査を完了した日 22.05.2015	国際調査報告の発送日 02.06.2015	
国際調査機関の名称及びあて先 日本国特許庁 (ISA/J P) 郵便番号100-8915 東京都千代田区霞が関三丁目4番3号	特許庁審査官 (権限のある職員) 中里 裕正 電話番号 03-3581-1101 内線 3546	5 S 9364

C (続き) . 関連すると認められる文献		
引用文献の カテゴリー*	引用文献名 及び一部の箇所が関連するときは、その関連する箇所の表示	関連する 請求項の番号
Y	WO 2009/087972 A1 (日本電気株式会社, 北陸日本電気ソフトウェア株式会社) 2009.07.16, [0088]-[0095], 図 12, 13	10
Y	JP 2005-107078 A (ソニー株式会社) 2005.04.21, [0103]-[0121], 図 9	11
Y	Wang, C. and Heys, H. M., AN ULTRA COMPACT BLOCK CIPHER FOR SERIALIZED ARCHITECTURE IMPLEMENTATIONS, 2009 Canadian Conference on Electrical and Computer Engineering, p.1085-1090, 2009, especially 2.CIPHER SPECIFICATION	12
Y A	Barreto, P. S.L.M. and Rijmen, V., The ANUBIS Block Cipher, First Open NESSIE Workshop [online], 2000.11, [retrieved on 2015.05.21] Retrieved from the Internet: <URL:https://www.cosic.esat.kuleuven.be/nessie/workshop/submissions/a nubis.zip, anubis/B/Anubis.pdf> especially 3 Description of the ANUBIS primitive	12 2-5, 13-16, 18
A	Borghoff, J. et al., PRINCE - A Low-latency Block Cipher for Pervasive Computing Applications, Cryptology ePrint Archive [online], Report 2012/529, 2012.09.13, [retrieved on 2015.05.21] Retrieved from the Internet: <URL: https://eprint.iacr.org/2012/529/20120913:093817> especially 2.CIPHER SPECIFICATION	1-18

第II欄 請求の範囲の一部の調査ができないときの意見（第1ページの2の続き）

法第8条第3項（PCT17条(2)(a)）の規定により、この国際調査報告は次の理由により請求の範囲の一部について作成しなかった。

1. 請求項 _____ は、この国際調査機関が調査をすることを要しない対象に係るものである。つまり、

2. 請求項 _____ は、有意義な国際調査をすることができる程度まで所定の要件を満たしていない国際出願の部分に係るものである。つまり、

3. 請求項 _____ は、従属請求の範囲であってPCT規則6.4(a)の第2文及び第3文の規定に従って記載されていない。

第III欄 発明の単一性が欠如しているときの意見（第1ページの3の続き）

次に述べるようにこの国際出願に二以上の発明があるところの国際調査機関は認めた。
本願請求項に係る発明は、以下の二群に区分される。

1. 請求項1-12, 17
鍵スケジュール部における変換関数がインポリューション性を有する暗号処理装置

2. 請求項13-16, 18
鍵スケジュール部における変換関数がフルディフュージョン性を有する暗号処理装置

1. 出願人が必要な追加調査手数料をすべて期間内に納付したので、この国際調査報告は、すべての調査可能な請求項について作成した。

2. 追加調査手数料を要求するまでもなく、すべての調査可能な請求項について調査することができたので、追加調査手数料の納付を求めなかった。

3. 出願人が必要な追加調査手数料を一部のみしか期間内に納付しなかったため、この国際調査報告は、手数料の納付のあった次の請求項のみについて作成した。

4. 出願人が必要な追加調査手数料を期間内に納付しなかったため、この国際調査報告は、請求の範囲の最初に記載されている発明に係る次の請求項について作成した。

追加調査手数料の異議の申立てに関する注意

- 追加調査手数料及び、該当する場合には、異議申立手数料の納付と共に、出願人から異議申立てがあった。
- 追加調査手数料の納付と共に出願人から異議申立てがあったが、異議申立手数料が納付命令書に示した期間内に支払われなかった。
- 追加調査手数料の納付はあったが、異議申立てはなかった。

JP 2000-66586 A	2000. 03. 03	US 6819764 B1 EP 982895 A2	2004. 11. 16 2000. 03. 01
JP 2012-70048 A	2012. 04. 05	US 2012/0069997 A1	2012. 03. 22
WO 2008/026625 A1	2008. 03. 06	JP 2008-58831 A US 2010/0008498 A1 EP 2058784 A1 CN 101512619 A TW 200828196 A	2008. 03. 13 2010. 01. 14 2009. 05. 13 2009. 08. 19 2008. 07. 01
JP 2002-23622 A	2002. 01. 23	US 2002/0021802 A1 EP 1172965 A2 KR 10-2002-0006475 A CN 1349182 A	2002. 02. 21 2002. 01. 16 2002. 01. 19 2002. 05. 15
WO 2009/087972 A1	2009. 07. 16,	US 2011/0110519 A1	2011. 05. 12
JP 2005-107078 A	2005. 04. 21	US 2005/0111659 A1	2005. 05. 26