(19)日本国特許庁(JP)　(12)特　許　公　報(B2)　(11)特許番号
特許第3622981号
(P3622981)
(45)発行日　平成17年2月23日(2005.2.23)　(24)登録日　平成16年12月3日(2004.12.3)

(51)Int.Cl.[7]　F I
H03M 13/15　H03M 13/15

請求項の数 5　(全 41 頁)

(21)出願番号　特願平9-514154
(86)(22)出願日　平成8年10月2日(1996.10.2)
(86)国際出願番号　PCT/JP1996/002866
(87)国際公開番号　WO1997/013328
(87)国際公開日　平成9年4月10日(1997.4.10)
審査請求日　平成15年2月13日(2003.2.13)
(31)優先権主張番号　特願平7-255991
(32)優先日　平成7年10月3日(1995.10.3)
(33)優先権主張国　日本国(JP)

(73)特許権者
松下電器産業株式会社
大阪府門真市大字門真１００６番地
(74)代理人
弁理士　青山　葆
(74)代理人
弁理士　河宮　治
(74)代理人
弁理士　石野　正弘
(72)発明者　藪野　寛之
大阪府交野市妙見坂５－５－４０１
(72)発明者　弓場　隆司
京都府宇治市木幡西浦５８－６０６

審査官　藤井　浩

最終頁に続く

(54)【発明の名称】誤り訂正符号化装置及び方法、並びに誤り訂正復号化装置及び方法

(57)【特許請求の範囲】

【請求項１】

2^Nの元の数を有するガロア体GF（2^N）上の元を有するリード・ソロモン符号を用いて、１シンボル当たり自然数Ｎビットの入力データに対する誤り訂正符号を符号化する誤り訂正符号化装置において、

各入力データと上記リード・ソロモン符号の生成多項式の各係数とのガロア体上の複数の積データをそれぞれ予め演算して、上記複数の積データを、各アドレスに対して複数ｂ個の積データを１組として予め記憶する積データ記憶手段と、

それぞれＮ×ｂビットの記憶容量を有する自然数ｍ個の記憶装置からなる第１の記憶手段と、

入力データに応答して、上記積データ記憶手段に記憶された複数の積データを、複数ｂ個の積データを１組として並列に読み出すように上記積データ記憶手段を制御する読み出し制御手段と、

それぞれＮ×ｂビットの第１と第２の入力端子を有し、上記読み出し制御手段によって上記積データ記憶手段から並列に読み出される複数ｂ個の積データが第１の入力端子に入力され、上記第１の入力端子に入力されるデータと、上記第２の入力端子に入力されるデータとの排他的論理和を演算して演算結果のデータを出力する排他的論理和演算手段と、

上記ｍ個の記憶装置に記憶されたデータを１つの記憶装置毎に選択的に順次読み出して出力し、上記選択的に読み出して出力されるＮ×ｂビットのデータのうち上位Ｎ×（ｂ－１）ビットのデータを上記排他的論理和演算手段の第２の入力端子の下位Ｎ×（ｂ－１）ビ

ットに出力するとともに、上記排他的論理和演算手段から出力される演算結果のデータを、上記ｍ個の記憶装置のうちの１つに選択的に順次切り換えて書き込むように、上記第１の記憶手段を制御する第１の選択手段と、
Ｎビットの記憶容量を有し、上記第１の選択手段によって上記ｍ個の記憶装置のうちの１つから選択的に出力されるＮ×ｂビットのデータのうち下位Ｎビットのデータを一時的に記憶して上記排他的論理和演算手段の第２の入力端子の上位Ｎビットに出力する第２の記憶手段と、
上記入力データを上記積データ記憶手段に順次入力することにより、上記第１の記憶手段のｍ個の記憶装置においてパリティデータを生成し、上記入力データに続いて、上記ｍ個の記憶装置を１つの記憶装置毎に選択的に順次切り換えることにより、上記ｍ個の記憶装置において生成される各パリティデータを順次出力する第２の選択手段とを備えたことを特徴とする誤り訂正符号化装置。

【請求項２】
上記読み出し制御手段と、上記第１の選択手段と、上記第２の選択手段とは、別の記憶装置に記憶された所定のプログラムを実行する中央演算制御装置によって構成されたことを特徴とする請求項１記載の誤り訂正符号化装置。

【請求項３】
2^Nの元の数を有するガロア体GF（2^N）上の元を有するリード・ソロモン符号を用いて、１シンボル当たり自然数Ｎビットの入力データに対して符号化された誤り訂正符号を復号化する誤り訂正復号化装置において、
上記入力データと、上記入力データに対するパリティデータとを含む複数の受信シンボルからなり、入力される受信語を各受信シンボル毎に記憶する受信語記憶手段と、
請求項１記載の誤り訂正符号化装置を備え、上記リード・ソロモン符号の生成多項式を用いて、上記入力される受信語に対する剰余を演算して出力する剰余演算手段と、
上記剰余演算手段から出力される剰余に基づいて、上記受信語における誤り位置と、上記誤り位置に対応する誤り数値との組を演算して出力する誤り数値及び誤り位置演算手段と、
上記誤り数値及び誤り位置演算手段から出力される上記受信語における誤り位置に基づいて、上記受信語記憶手段に記憶された上記誤り位置における受信シンボルを上記受信語記憶手段から読み出して出力する読み出し制御手段と、
上記読み出し制御手段から出力される上記誤り位置における受信シンボルと、上記誤り数値及び誤り位置演算手段から出力される、上記誤り位置に対応する誤り数値との排他的論理和を演算して、演算結果のデータを出力する排他的論理和演算手段と、
上記排他的論理和演算手段から出力される演算結果のデータを、上記受信語記憶手段の上記誤り位置に書き込むことにより、上記誤り位置における受信シンボルを訂正する書き込み制御手段とを備えたことを特徴とする誤り訂正復号化装置。

【請求項４】
2^Nの元の数を有するガロア体GF（2^N）上の元を有するリード・ソロモン符号を用いて、１シンボル当たり自然数Ｎビットの入力データに対する誤り訂正符号を符号化する誤り訂正符号化方法において、
各入力データと上記リード・ソロモン符号の生成多項式の各係数とのガロア体上の複数の積データをそれぞれ予め演算して、上記複数の積データを、各アドレスに対して複数ｂ個の積データを１組として積データ記憶手段に予め記憶するステップと、
入力データに応答して、上記積データ記憶手段に記憶された複数の積データを、複数ｂ個の積データを１組として並列に読み出すように上記積データ記憶手段を制御するステップと、
それぞれＮ×ｂビットの第１と第２の入力端子を有する排他的論理和演算手段を用いて、上記積データ記憶手段から並列に読み出される複数ｂ個の積データが第１の入力端子に入力され、上記第１の入力端子に入力されるデータと、上記第２の入力端子に入力されるデータとの排他的論理和を演算して演算結果のデータを出力するステップと、

それぞれＮ×ｂビットの記憶容量を有するｍ個の記憶装置の第１の記憶手段に記憶されたデータを１つの記憶装置毎に選択的に順次読み出して出力し、上記選択的に読み出して出力されるＮ×ｂビットのデータのうち上記Ｎ×（ｂ－１）ビットのデータを上記排他的論理和演算手段の第２の入力端子の下位Ｎ×（ｂ－１）ビットに出力するとともに、上記排他的論理和演算手段から出力される演算結果のデータを、上記ｍ個の記憶装置のうちの１つに選択的に順次切り換えて書き込むように、上記第１の記憶手段を制御するステップと、

Ｎビットの記憶容量を有する第２の記憶手段を用いて、上記ｍ個の記憶装置のうちの１つから選択的に出力されるＮ×ｂビットのデータのうち下位Ｎビットのデータを一時的に記憶して上記排他的論理和演算手段の第２の入力端子の上位Ｎビットに出力するステップと、

上記入力データを上記積データ記憶手段に順次入力することにより、上記第１の記憶手段のｍ個の記憶装置においてパリティデータを生成し、上記入力データに続いて、上記ｍ個の記憶装置を１つの記憶装置毎に選択的に順次切り換えることにより、上記ｍ個の記憶装置において生成される各パリティデータを順次出力するステップとを含むことを特徴とする誤り訂正符号化方法。

【請求項５】

2^Nの元の数を有するガロア体GF（2^N）上の元を有するリード・ソロモン符号を用いて、１シンボル当たり自然数Ｎビットの入力データに対して符号化された誤り訂正符号を復号化する誤り訂正復号化方法において、

上記入力データと、上記入力データに対するパリティデータとを含む複数の受信シンボルからなり、入力される受信語を各受信シンボル毎に受信語記憶手段に記憶するステップと、

請求項４記載の誤り訂正符号化方法により、上記リード・ソロモン符号の生成多項式を用いて、上記入力される受信語に対する剰余を演算して出力するステップと、

上記出力される剰余に基づいて、上記受信語における誤り位置と、上記誤り位置に対応する誤り数値との組を演算して出力するステップと

上記出力される上記受信語における誤り位置に基づいて、上記受信語記憶手段に記憶された上記誤り位置における受信シンボルを上記受信語記憶手段から読み出して出力するステップと、

上記出力される上記誤り位置における受信シンボルと、上記出力される、上記誤り位置に対応する誤り数値との排他的論理和を演算して、演算結果のデータを出力するステップと、

上記出力される演算結果のデータを、上記受信語記憶手段の上記誤り位置に書き込むことにより、上記誤り位置における受信シンボルを訂正するステップとを含むことを特徴とする誤り訂正復号化方法。

【発明の詳細な説明】

技術分野

本発明は、誤り訂正符号化装置及び方法、並びに誤り訂正復号化装置及び方法に関する。特に、ディジタル記録装置やディジタル通信装置等によって、ディジタルデータの記録又は再生、もしくは、送信又は受信を行う際に、ディジタルデータにおける誤りを訂正する誤り訂正符号を符号化するための誤り訂正符号化装置及び方法、並びに、誤り訂正符号を復号化するための誤り訂正復号化装置及び方法に関するものである。

背景技術

近年、ディジタル記録装置やディジタル通信装置の発達に伴い、ディジタルデータを記録又は再生する場合、もしくは、送信又は受信する場合において、ディジタルデータの誤りを如何に少なくするかは、重要な課題となっている。そこで、ディジタルデータの誤りを訂正するために、誤り訂正符号がディジタルデータを取り扱う各種の装置で用いられている。リード・ソロモン符号も、そのような誤り訂正符号の一種であって、主として、例えば、相変化を利用するPDドライバユニットなどのディジタル記録装置に用いられている。

リード・ソロモン符号は、符号語が、元の数が2^Nであるガロア体GF（2^N）の元から構成され、αをGF（2^N）の原始元としたとき、生成多項式が次式で表される多元の巡回ハミング符号である。

$G(X)=(X-\alpha^0)(X-\alpha^1)\cdots(X-\alpha^{d-2})$ …（１）

ただし、式（１）を含めて、以後、演算は全てガロア体GF（2^N）上で行う。また、ｄは最小符号間距離を表す。

リード・ソロモン符号の符号語は以下のように生成する。

情報後ベクトルＩを次式で表すと、

$I=(i_0,i_1,\cdots,i_{k-1})$ …（２）

情報多項式Ｉ（Ｘ）は

$I(X)=i_0\cdot X^{k-1}+i_1\cdot X^{k-2}+\cdots+i_{k-2}\cdot X+i_{k-1}$ …（３）

と表すことができる。ただし、$i_0,i_1,\cdots,i_{k-1}$はそれぞれ情報シンボルであり、情報の元となるビットデータについて、Ｎビットを１かたまりとして、ガロア体GF（2^N）上の元をベクトル表現したものと対応づける。

そして、符号多項式Ａ（Ｘ）は、情報多項式Ｉ（Ｘ）と生成多項式Ｇ（Ｘ）から次式を用いて求めることができる。

$A(X)=I(X)\cdot G(X)$ …（４）

しかしながら、得られる符号は組織符号にならない。そこで、次のように符号語を作成する。

まず、情報多項式Ｉ（Ｘ）にX^{n-k}をかけ、それをＧ（Ｘ）で割り、その商をＱ（Ｘ）、剰余をＲ（Ｘ）とすると、

$I(X)\cdot X^{n-k}=Q(X)\cdot G(X)+R(X)$ …（５）

と表わすことができる。ここで、

$A(X)=R(X)+I(X)\cdot X^{n-k}$ …（６）

とおくと、式（５）より

$A(X)=Q(X)\cdot G(X)$ …（７）

が成立する。式（７）によって求められたＡ（Ｘ）は生成多項式Ｇ（Ｘ）で割りきれるので、符号多項式になる。符号多項式Ｒ（Ｘ）を

$$R(X)=r_0\cdot X^{n-k-1}+r_1\cdot X^{n-k-2}+\cdots+r_{n-k-2}\cdot X+r_{n-k-1}$$

…（８）

で表わすと、式（６）で表される符号多項式Ａ（Ｘ）は

$A(X)=i_0\cdot X^{n-1}+i_1\cdot X^{n-2}+\cdots$
$+i_{k-2}\cdot X^{n-k+1}+i_{k-1}\cdot X^{n-k}$
$+r_{n-k-1}\cdot X^{n-k-1}+r_{n-k-2}\cdot X^{n-k-2}+\cdots+r_1\cdot X+r_0$ …（９）

と表される。式（９）の符号多項式で表される符号語をベクトル表現で表記すると

$A=i_0,i_1,\cdots,i_{k-1},r_0,r_1,\cdots,r_{n-k-1})$ …（10）

となり、式（10）で表される符号語ベクトルＡは、情報語ベクトルＩをそのまま含んでいるから、組織符号であることがわかる。この場合、符号語ベクトルＡは（n,k）組織符号である。符号語を作成する際に、情報語ベクトルに付加するベクトルＲ、つまり

$R=r_0,r_1,\cdots,r_{n-k-1})$ …（11）

はパリティ語ベクトルである。

以上のように生成する符号を、リード・ソロモン符号RS（n,k,d＝ｎ－ｋ＋１）と書く。

図12にリード・ソロモン符号を用いた従来技術の誤り訂正符号化装置の一例を示す。この回路は、ガロア体GF（2^N）の係数を有する多項式の除算を行うものである。図12において、誤り訂正符号化装置は、

（ａ）８ビット排他的論理和演算器195乃至206と、

（ｂ）ガロア体上の係数k_{12}乃至k_1を有する係数乗算器171乃至182と、

（ｃ）８ビットラッチ183乃至194と、

（ｄ）リセット時に８ビットラッチ183乃至194の内容を０にクリアして初期化する初期値

設定回路207とを備える。
当該誤り訂正符号化装置において、入力データは、排他的論理和演算器206の第１の入力端子に入力され、排他的論理和演算器206の出力端子から出力されるデータは、係数乗算器171を介してラッチ183に入力されるとともに、各係数乗算器172乃至182を介してそれぞれ排他的論理和演算器195乃至205に入力される。さらに、ラッチ183乃至194と、排他的論理和演算器195乃至206とは、交互に配置されかつ、データがラッチ183からラッチ194に向かって伝送されるように継続接続される。
次に、図12の誤り訂正符号化装置を用いて実際に８ビットを１シンボルとしたリード・ソロモン符号RS（32,20,d＝13）を符号化する場合を説明する。
ただし、原始多項式ｍ（Ｘ）と、原始元αと、生成多項式Ｇ（Ｘ）はそれぞれ以下のように定義する。

$$m(X) = X^8 + X^4 + X^3 + X^2 + 1 \quad \cdots (12)$$

$$\alpha = (00000010) \quad \cdots (13)$$

$$G(X) = \prod_{i=0}^{11} (X - \alpha^i) = k_1 \cdot X^{11} + k_2 \cdot X^{10} + \cdots + k_{11} \cdot X + k_{12} \quad \cdots (14)$$

ただし、式（14）のk_1からk_{12}は、生成多項式Ｇ（Ｘ）を展開して、Ｘの降べきの順に並べたときの、X^{11}からX^0にかかる係数を表す。
第１の入力データである第１の情報シンボルi_{000}が排他的論理和演算器206に入力されると、第１の情報シンボルi_{000}と、８ビットラッチ194の出力データである00Hとの排他的論理和が演算され、演算結果のデータがガロア体係数乗算器171乃至182に入力される。ここで、00HのＨは、16進数表示を表わし、以下同様である。この場合において、ガロア体係数乗算器171乃至182に入力されるデータをd_{000}とすると、第１の情報シンボルi_{000}とデータ00Hとの排他的論理和であるデータd_{000}は、次式で表わすことができる。

$$d_{000} = i_{000} \quad \cdots (15)$$

これは、図13において、列番号R1でかつステップS101の３行分にある演算に対応する。次に、ガロア体係数乗算器171乃至182が、入力データd_{000}とそれぞれの係数k_{12}乃至k_1とのガロア体上の積を出力する。これらの積は、図13における列番号R13からR2までででかつステップS101の２行目にあるデータに相当する。
次に、ガロア体係数乗算器171乃至182の出力データがそれぞれ８ビットラッチ183乃至194に格納される。ここで、各ラッチ183乃至194に格納されるデータをそれぞれp_{000}からp_{011}とすると、これらのデータ値は、図13において列番号R13からR2までででかつステップS101の３行分にある演算を行った結果に対応する。ただし、図13における加算記号は排他的論理和演算を表し、以下、演算EORは排他的論理和演算を表す。
次に、第２の入力データである第２の情報シンボルi_{001}が排他的論理和演算器206に入力されると、第２の情報シンボルi_{001}と、８ビットラッチ194の出力データであるp_{011}との排他的論理和が演算され、演算結果のデータがガロア体係数乗算器171乃至182に入力される。この場合、ガロア体係数乗算器171乃至182に入力されるデータをd_{001}とすると、第２の情報シンボルi_{001}とデータp_{011}との排他的論理和であるデータd_{001}は次式で表わすことができる。

$$d_{001} = i_{001} + p_{011} \quad \cdots (16)$$

これは、図13において、列番号R2でかつステップS101の３行分にある演算に対応する。次に、ガロア体係数乗算器171乃至182は、入力データd_{001}と、各係数k_{12}乃至k_1とのガロア体上の積を出力する。これらの積は、図13における列番号R14からR3までででかつステップS102の２行目にあるデータに対応する。
次に、ガロア体係数乗算器171乃至182の出力データがそれぞれ８ビットラッチ183乃至194に格納される。各ラッチ183乃至194に格納されるデータをp_{012}からp_{023}とすると、これらのデータ値は、図13において列番号R14からR3までででかつステップS102の３行分にある演

算を行った結果に対応する。
以下、同様に、情報シンボルi_{002}から情報シンボルi_{019}を排他的論理和演算器206に入力していくと、図13に示す演算が続けられ、最後に、情報語（$i_{000},i_{001},\cdots,i_{019}$）を生成多項式（式（14））で割った余りであるパリティ語（$p_{228},p_{229},\cdots,p_{239}$）がそれぞれ、８ビットラッチ183乃至194に格納される。今まで入力した情報シンボル$i_{000},i_{001},\cdots,i_{019}$の続きに、最後に得られたパリティシンボル$p_{228},p_{229},\cdots,p_{239}$を付加すると、符号語（$i_{000},i_{001},\cdots,i_{019},p_{228},p_{229},\cdots,p_{239}$）が完成する。
しかしながら、図12に示すような誤り訂正符号化装置では、ガロア体上の係数乗算器171乃至182はそれぞれ複雑な回路を有しているため、最小符号間距離の長い符号を符号化する場合、大規模な回路となる。また、図12に示すような誤り訂正符号化装置では、装置の構成を変えずに最小符号間距離を変更することは、困難である。装置の構成を変えずに最小符号間距離の変更を行うためには、装置の構成を変えずにガロア体上の係数乗算器の係数を変更可能にすることと、装置の構成を変えずに除算回路のループを変更できる回路にする必要があり、それらを実現するには、さらに複雑な回路になる。
本発明の第１の目的は以上の問題点を解決し、実用に値する符号化速度を持ちながら、従来技術に比較して小規模な回路構成で実現可能であるとともに、装置の構成を変えずに最小符号間距離ｄを自由に変更可能な誤り訂正符号化装置及び方法を提供することにある。
本発明の第２の目的は、実用に値する復号化速度を持ちながら、従来技術に比較して小規模な回路構成で実現可能であるとともに、装置の構成を変えずに最小符号間距離ｄを自由に変更可能な誤り訂正復号化装置及び方法を提供することにある。

発明の開示

本発明に係る誤り訂正符号化装置によれば、2^Nの元の数を有するガロア体GF（2^N）上の元を有するリード・ソロモン符号を用いて、１シンボル当たり自然数Ｎビットの入力データに対する誤り訂正符号を符号化する誤り訂正符号化装置において、
各入力データと上記リード・ソロモン符号の生成多項式の各係数とのガロア体上の複数の積データをそれぞれ予め演算して、上記複数の積データを、各アドレスに対して複数ｂ個の積データを１組として予め記憶する積データ記憶手段と、
それぞれＮ×ｂビットの記憶容量を有する自然数ｍ個の記憶装置からなる第１の記憶手段と、
入力データに応答して、上記積データ記憶手段に記憶された複数の積データを、複数ｂ個の積データを１組として並列に読み出すように上記積データ記憶手段を制御する読み出し制御手段と、
それぞれＮ×ｂビットの第１と第２の入力端子を有し、上記読み出し制御手段によって上記積データ記憶手段から並列に読み出される複数ｂ個の積データが第１の入力端子に入力され、上記第１の入力端子に入力されるデータと、上記第２の入力端子に入力されるデータとの排他的論理和を演算して演算結果のデータを出力する排他的論理和演算手段と、
上記ｍ個の記憶装置に記憶されたデータを１つの記憶装置毎に選択的に順次読み出して出力し、上記選択的に読み出して出力されるＮ×ｂビットのデータのうち上位Ｎ×（ｂ－１）ビットのデータを上記排他的論理和演算手段の第２の入力端子の下位Ｎ×（ｂ－１）ビットに出力するとともに、上記排他的論理和演算手段から出力される演算結果のデータを、上記ｍ個の記憶装置のうちの１つに選択的に順次切り換えて書き込むように、上記第１の記憶手段を制御する第１の選択手段と、
Ｎビットの記憶容量を有し、上記第１の選択手段によって上記ｍ個の記憶装置のうちの１つから選択的に出力されるＮ×ｂビットのデータのうち下位Ｎビットのデータを一時的に記憶して上記排他的論理和演算手段の第２の入力端子の上位Ｎビットに出力する第２の記憶手段と、
上記入力データを上記積データ記憶手段に順次入力することにより、上記第１の記憶手段のｍ個の記憶装置においてパリティデータを生成し、上記入力データに続いて、上記ｍ個の記憶装置を１つの記憶装置毎に選択的に順次切り換えることにより、上記ｍ個の記憶装置において生成される各パリティデータを順次出力する第２の選択手段とを備えたことを

特徴とする。
また、上記誤り訂正符号化装置において、好ましくは、上記読み出し制御手段と、上記第１の選択手段と、上記第２の選択手段とは、別の記憶装置に記憶された所定のプログラムを実行する中央演算制御装置によって構成される。
また、本発明に係る誤り訂正復号化装置によれば、2^Nの元の数を有するガロア体GF（2^N）上の元を有するリード・ソロモン符号を用いて、１シンボル当たり自然数Ｎビットの入力データに対して符号化された誤り訂正符号を復号化する誤り訂正復号化装置において、
上記入力データと、上記入力データに対するパリティデータとを含む複数の受信シンボルからなり、入力される受信語を各受信シンボル毎に記憶する受信語記憶手段と、
上記誤り訂正符号化装置を備え、上記リード・ソロモン符号の生成多項式を用いて、上記入力される受信語に対する剰余を演算して出力する剰余演算手段と、
上記剰余演算手段から出力される剰余に基づいて、上記受信語における誤り位置と、上記誤り位置に対応する誤り数値との組を演算して出力する誤り数値及び誤り位置演算手段と、
上記誤り数値及び誤り位置演算手段から出力される上記受信語における誤り位置に基づいて、上記受信語記憶手段に記憶された上記誤り位置における受信シンボルを上記受信語記憶手段から読み出して出力する読み出し制御手段と、
上記読み出し制御手段から出力される上記誤り位置における受信シンボルと、上記誤り数値及び誤り位置演算手段から出力される、上記誤り位置に対応する誤り数値との排他的論理和を演算して、演算結果のデータを出力する排他的論理和演算手段と、
上記排他的論理和演算手段から出力される演算結果のデータを、上記受信語記憶手段の上記誤り位置に書き込むことにより、上記誤り位置における受信シンボルを訂正する書き込み制御手段とを備えたことを特徴とする。
さらに、本発明に係る誤り訂正符号化方法によれば、2^Nの元の数を有するガロア体GF（2^N）上の元を有するリード・ソロモン符号を用いて、１シンボル当たり自然数Ｎビットの入力データに対する誤り訂正符号を符号化する誤り訂正符号化方法において、
各入力データを上記リード・ソロモン符号の生成多項式の各係数とのガロア体上の複数の積データをそれぞれ予め演算して、上記複数の積データを、各アドレスに対して複数ｂ個の積データを１組として積データ記憶手段に予め記憶するステップと、
入力データに応答して、上記積データ記憶手段に記憶された複数の積データを、複数ｂ個の積データを１組として並列に読み出すように上記積データ記憶手段を制御するステップと、
それぞれＮ×ｂビットの第１と第２の入力端子を有する排他的論理和演算手段を用いて、上記積データ記憶手段から並列に読み出される複数ｂ個の積データが第１の入力端子に入力され、上記第１の入力端子に入力されるデータと、上記第２の入力端子に入力されるデータとの排他的論理和を演算して演算結果のデータを出力するステップと、
それぞれＮ×ｂビットの記憶容量を有するｍ個の記憶装置の第１の記憶手段に記憶されたデータを１つの記憶装置毎に選択的に順次読み出して出力し、上記選択的に読み出して出力されるＮ×ｂビットのデータのうち上記Ｎ×（ｂ－１）ビットのデータを上記排他的論理和演算手段の第２の入力端子の下位Ｎ×（ｂ－１）ビットに出力するとともに、上記排他的論理和演算手段から出力される演算結果のデータを、上記ｍ個の記憶装置のうちの１つに選択的に順次切り換えて書き込むように、上記第１の記憶手段を制御するステップと、
Ｎビットの記憶容量を有する第２の記憶手段を用いて、上記ｍ個の記憶装置のうちの１つから選択的に出力されるＮ×ｂビットのデータのうち下位Ｎビットのデータを一時的に記憶して上記排他的論理和演算手段の第２の入力端子の上位Ｎビットに出力するステップと、
上記入力データを上記積データ記憶手段に順次入力することにより、上記第１の記憶手段のｍ個の記憶装置においてパリティデータを生成し、上記入力データに続いて、上記ｍ個の記憶装置を１つの記憶装置毎に選択的に順次切り換えることにより、上記ｍ個の記憶装

置において生成される各パリティデータを順次出力するステップとを含むことを特徴とする。
またさらに、本発明に係る誤り訂正復号化方法によれば、2^Nの元の数を有するガロア体GF（2^N）上の元を有するリード・ソロモン符号を用いて、１シンボル当たり自然数Ｎビットの入力データに対して符号化された誤り訂正符号を復号化する誤り訂正復号化方法において、
上記入力データと、上記入力データに対するパリティデータとを含む複数の受信シンボルからなり、入力される受信語を各受信シンボル毎に受信語記憶手段に記憶するステップと、
上記誤り訂正符号化方法により、上記リード・ソロモン符号の生成多項式を用いて、上記入力される受信語に対する剰余を演算して出力するステップと、
上記出力される剰余に基づいて、上記受信語における誤り位置と、上記誤り位置に対応する誤り数値との組を演算して出力するステップと
上記出力される上記受信語における誤り位置に基づいて、上記受信語記憶手段に記憶された上記誤り位置における受信シンボルを上記受信語記憶手段から読み出して出力するステップと、
上記出力される上記誤り位置における受信シンボルと、上記出力される、上記誤り位置に対応する誤り数値との排他的論理和を演算して、演算結果のデータを出力するステップと、
上記出力される演算結果のデータを、上記受信語記憶手段の上記誤り位置に書き込むことにより、上記誤り位置における受信シンボルを訂正するステップとを含むことを特徴とする。

【図面の簡単な説明】

図１は、本発明に係る第１の実施形態の誤り訂正符号化装置のブロック図である。
図２は、図１の誤り訂正符号化装置の動作を示すタイミングチャートである。
図３は、本発明に係る第２の実施形態の誤り訂正符号化装置のブロック図である。
図４は、図３の誤り訂正符号化装置においてｍ＝３としたときのブロック図である。
図５は、図３の誤り訂正符号化装置のCPU61によって実行される符号化処理を示すフローチャートである。
図６は、図５のサブルーチンである初期化処理（ステップS1）を示すフローチャートである。
図７は、図５のサブルーチンである符号語生成処理（ステップS2）を示すフローチャートである。
図８は、図７のサブルーチン処理P1（ステップS7）を示すフローチャートである。
図９は、図７及び図８のサブルーチン処理P2（ステップS23,S35）を示すフローチャートである。
図10は、図８のサブルーチン処理P3（ステップS37）を示すフローチャートである。
図11は、図７のサブルーチン処理P4（ステップS24）を示すフローチャートである。
図12は、従来技術の誤り訂正符号化装置のブロック図である。
図13は、図12の誤り訂正符号化装置においてリード・ソロモン符号RS（n,n－12,d＝13）の計算方法を説明するための図である。
図14は、図12の誤り訂正符号化装置においてリード・ソロモン符号RS（n,n－5,d＝６）の計算方法を説明するための図である。
図15は、本発明に係る第３の実施形態の誤り訂正復号化装置のブロック図である。

発明を実施するための最良の形態

以下、本発明に係る実施形態について図面を参照して説明する。

＜第１の実施形態＞

本発明に係る第１の実施形態について、図面を参照しながら説明する。第１の実施形態では、８ビットを１シンボルとしたリード・ソロモン符号RS（32,20,d＝13）を符号化する場合を説明する。ただし、原始多項式ｍ（Ｘ）、原始元α、生成多項式Ｇ（Ｘ）はそれぞ

れ式（12）、式（13）、式（14）のように定義する。
図１は、本発明に係る第１の実施形態の誤り訂正符号化装置の構成を示すブロック図である。この第１の実施形態の誤り訂正符号化装置は、
（ａ）８ビットラッチ12,27と、
（ｂ）４進カウンタ13と、
（ｃ）ROM（読み出し専用メモリ）14と、
（ｄ）８ビット排他的論理和演算器11,15,16,17,18と、
（ｅ）32ビットバス選択器19,23と、
（ｆ）32ビットラッチ20,21,22からなるキューメモリ30と、
（ｇ）入力データ“1"をデコードしてパルス信号を出力するデコーダ24と、
（ｈ）入力データ“2"をデコードしてパルス信号を出力するデコーダ25と、
（ｉ）入力データ“3"をデコードしてパルス信号を出力するデコーダ26と、
（ｊ）入力データ“0"をデコードしてパルス信号を出力するデコーダ28と、
（ｋ）２ビットラッチ29と、
（ｌ）３個のシンボル遅延回路301乃至303からなる遅延回路300と、
（ｍ）８ビット出力データ選択器310とを備える。
図１の誤り訂正符号化装置において、８ビットの入力データは、排他的論理和演算器11の第１の入力端子に入力されるとともに、出力データ選択器310のｄ接点を介して出力データとして出力される。排他的論理和演算器18の出力端子からの出力データは、排他的論理和演算器11の第２の入力端子に入力される。一方、クロックCLKは、４進カウンタ13と、ラッチ29及び27に入力される。４進カウンタ13は、入力されるクロックCLKを計数し、かつ計数値が４となると０にリセットし、その計数値の２ビットのデータを、ラッチ29を介して上位２ビットのアドレスとしてROM14のアドレス端子に出力するとともに、選択信号としてバス選択器19,23及びデコーダ24,25,26,28に出力する。ここで、２ビットラッチ29は、４進カウンタ13の計数値の２ビットデータを、クロックCLKの立ち下がりのタイミングでラッチする。
デコーダ28は、４進カウンタ13の出力データが“0"の値を示したとき、クロックCLKの半周期の時間だけ遅れてクロックCLKの半周期の幅を有するパルス信号をラッチ27に出力する。また、デコーダ24は、４進カウンタ13の出力データが“1"の値を示したとき、クロックCLKの半周期の時間だけ遅れてクロックCLKの半周期の幅を有するパルス信号をラッチ20に出力する。さらに、デコーダ25は、４進カウンタ13の出力データが“2"の値を示したとき、クロックCLKの半周期の時間だけ遅れてクロックCLKの半周期の幅を有するパルス信号をラッチ21に出力する。またさらに、デコーダ26は、４進カウンタ13の出力データが“3"の値を示したとき、クロックCLKの半周期の時間だけ遅れてクロックCLKの半周期の幅を有するパルス信号をラッチ22に出力する。
ここで、８ビットラッチ12は、排他的論理和演算器11からの８ビットデータを、デコーダ26からのパルス信号のクロックCLKの立ち上がりのタイミングでラッチしてROM14に下位８ビットのアドレスとして出力する。ROM14には、ベクトル表現の入力シンボルに対して所定の係数をガロア体上で乗算した結果がベクトル表現で予め格納されている。入力データはアドレスの下位８ビットとして与え、アドレスの上位２ビットは係数を切り替えるために用いる。ROM14のデータ端子からの32ビットの出力データは、４個の排他的論理和演算器15乃至18の第１の入力端子に入力され、その出力端子から出力される32ビットのデータは、32ビットバス選択器19の入力端子に入力される。また、排他的論理和演算器18から出力される８ビットの出力データは排他的論理和演算器11の第２の入力端子に入力される。
ここで、32ビットバス選択器19,23はそれぞれ、８ビットバス４本分を連動して４回路に切り替えるバス選択器である。32ビットバス選択器19,23は、選択信号が“0"のとき最下位置のｄ接点に切り換えられて、バス選択器19の入力端子及びバス選択器23の出力端子はそれぞれオープン状態となる。また、32ビットバス選択器19,23は、選択信号が“1"のとき最上位置のａ接点に切り換えられて、バス選択器19の入力端子は32ビットラッチ20の入力バスに接続されるとともに、バス選択器23は32ビットラッチ20の出力バスに接続される

。さらに、32ビットバス選択器19,23は、選択信号が“2"のとき最上から２番目の位置のｂ接点に切り換えられて、バス選択器19の入力端子は32ビットラッチ21の入力バスに接続されるとともに、バス選択器23は32ビットラッチ21の出力バスに接続される。またさらに、32ビットバス選択器19,23は、選択信号が“3"のとき最上から３番目の位置のｃ接点に切り換えられて、バス選択器19の入力端子は32ビットラッチ22の入力バスに接続されるとともに、バス選択器23は32ビットラッチ22の出力バスに接続される。
バス選択器23から出力される32ビットの出力データのうち上位24ビットのデータは、排他的論理和演算器17,18,19の第２の入力端子に入力され、バス選択器23から出力される32ビットの出力データのうち下位８ビットのデータは、８ビットラッチ27を介して、排他的論理和演算器15の第２の入力端子に入力される。ここで、８ビットラッチ27は、入力される８ビットのデータを、クロックCLKの立ち上がりのタイミングでラッチした後、排他的論理和演算器15の第２の入力端子に出力し、デコーダ28からのパルス信号に応答して、ラッチしているデータを０にクリアする。
32ビットラッチ20から出力される32ビットの出力データはシンボル遅延回路301に入力され、シンボル遅延回路301は、入力される32ビットのデータのうち
（ａ）上位８ビットデータを、バス選択器310のａ接点から出力される情報シンボルの後に、遅延することなく８ビットバス選択器310のｂ接点を介して出力データとして出力し（以下、この出力データの出力タイミングを基準出力タイミングという。）、
（ｂ）次の上位８ビットデータを上記基準出力タイミングから１シンボル（＝１バイト＝８ビット）だけ遅延した後、８ビットバス選択器310のｂ接点を介して出力データとして出力し、
（ｃ）次の上位８ビットデータを上記基準出力タイミングから２シンボルだけ遅延した後、８ビットバス選択器310のｂ接点を介して出力データとして出力し、
（ｄ）次の下位８ビットデータを上記基準出力タイミングから３シンボルだけ遅延した後、８ビットバス選択器310のｂ接点を介して出力データとして出力する。
また、32ビットラッチ21から出力される32ビットの出力データはシンボル遅延回路302に入力され、シンボル遅延回路302は、入力される32ビットのデータのうち
（ａ）上位８ビットデータを上記基準出力タイミングから４シンボルだけ遅延した後、８ビットバス選択器310のｃ接点を介して出力データとして出力し、
（ｂ）次の上位８ビットデータを上記基準出力タイミングから５シンボルだけ遅延した後、８ビットバス選択器310のｃ接点を介して出力データとして出力し、
（ｃ）次の上位８ビットデータを上記基準出力タイミングから６シンボルだけ遅延した後、８ビットバス選択器310のｃ接点を介して出力データとして出力し、
（ｄ）次の上位８ビットデータを上記基準出力タイミングから７シンボルだけ遅延した後、８ビットバス選択器310のｃ接点を介して出力データとして出力する。
さらに、32ビットラッチ22から出力される32ビットの出力データはシンボル遅延回路303に入力され、シンボル遅延回路303は、入力される32ビットのデータのうち
（ａ）上位８ビットデータを上記基準出力タイミングから８シンボルだけ遅延した後、８ビットバス選択器310のｄ接点を介して出力データとして出力し、
（ｂ）次の上位８ビットデータを上記基準出力タイミングから９シンボルだけ遅延した後、８ビットバス選択器310のｄ接点を介して出力データとして出力し、
（ｃ）次の上位８ビットデータを上記基準出力タイミングから10シンボルだけ遅延した後、８ビットバス選択器310のｄ接点を介して出力データとして出力し、
（ｄ）次の上位８ビットデータを上記基準出力タイミングから11シンボルだけ遅延した後、８ビットバス選択器310のｄ接点を介して出力データとして出力する。
従って、当該誤り訂正符号化装置に入力される、例えば20シンボルの情報シンボルに続いて、誤り訂正符号化されラッチ20に格納された32ビット＝４シンボルのパリティ語と、誤り訂正符号化されラッチ21に格納された32ビット＝４シンボルのパリティ語と、誤り訂正符号化されラッチ22に格納された32ビット＝４シンボルのパリティ語とが出力データとしてバス選択器310から出力される。

図２は、図１の誤り訂正符号化装置の動作を示すタイミングチャートである。入力データは８ビットで、クロックCLKの４個のパルスにつき１度入力される。また、４進カウンタ13の初期値は２に設定され、８ビットラッチ12,27、32ビットラッチ20,21,22の初期値は０に設定され、２ビットラッチ29の初期値は０に設定される。

表１及び表２は、ROM14に記憶されるデータの内容である。表１及び表２において、k_1乃至k_{12}は式（14）に示す生成多項式Ｇ（Ｘ）の係数であり、αは式（13）に示すガロア体GF（2^8）の原始元であり、積の記号はガロア体上の積を表す。k_m,α^n（ここで、ｍは１から12までの自然数であり、ｎは０以上の整数である。）はどちらもガロア体GF（2^8）の元である。従って、k_mとα^nとの積もガロア体GF（2^8）の元であり、ベクトル表現を用いると８ビットの数値に対応するので、その積の数値データはROM14に予め格納される。

表１及び表２において、アドレスＨは、ROM14の上位２ビットのアドレスであり、アドレスＬはROM14の下位８ビットのアドレスである。また、ｂ［31,…,24］は最上位８ビットの係数データであり、ｂ［23,…,16］は次の上位８ビットの係数データであり、ｂ［15,…,8］は次の上位８ビットの係数データであり、ｂ［7,…,0］は最下位８ビットの係数データである。

表３はガロア体GF（2^8）上の元をベクトル表現からべき表現に変換するための変換表である。表３は参考のために記載したものであり、本実施形態において直接用いるデータではない。表３から明らかなように、表１及び表２に示すROM14のアドレスＬのデータ値をべき表現に変換すれば、そのアドレスに書かれている値のα^n（ここで、ｎは０以上の整数である。）の部分に対応することがわかる。従って、ROM14において、例えばアドレスＨに０を入力し、アドレスＬにベクトル表現（８ビット）のガロア体GF（2^8）の元を入力すると、ROM14は生成多項式の係数k_{12},k_{11},k_{10},k_9とアドレスＬに入力した値のガロア体上の積を出力することがわかる。

表1

アドレスH	アドレスL	b[31, …, 24]	b[23, …, 16]	b[15, …, 8]	b[7, …, 0]
0	00H	0	0	0	0
0	01H	k_{12}	k_{11}	k_{10}	k_9
0	02H	$k_{12}\cdot\alpha$	$k_{11}\cdot\alpha$	$k_{10}\cdot\alpha$	$k_9\cdot\alpha$
0	03H	$k_{12}\cdot\alpha^{25}$	$k_{11}\cdot\alpha^{25}$	$k_{10}\cdot\alpha^{25}$	$k_9\cdot\alpha^{25}$
0	04H	$k_{12}\cdot\alpha^{2}$	$k_{11}\cdot\alpha^{2}$	$k_{10}\cdot\alpha^{2}$	$k_9\cdot\alpha^{2}$
0	05H	$k_{12}\cdot\alpha^{50}$	$k_{11}\cdot\alpha^{50}$	$k_{10}\cdot\alpha^{50}$	$k_9\cdot\alpha^{50}$
…	…	…	…	…	…
0	FBH	$k_{12}\cdot\alpha^{234}$	$k_{11}\cdot\alpha^{234}$	$k_{10}\cdot\alpha^{234}$	$k_9\cdot\alpha^{234}$
0	FCH	$k_{12}\cdot\alpha^{168}$	$k_{11}\cdot\alpha^{168}$	$k_{10}\cdot\alpha^{168}$	$k_9\cdot\alpha^{168}$
0	FDH	$k_{12}\cdot\alpha^{80}$	$k_{11}\cdot\alpha^{80}$	$k_{10}\cdot\alpha^{80}$	$k_9\cdot\alpha^{80}$
0	FEH	$k_{12}\cdot\alpha^{88}$	$k_{11}\cdot\alpha^{88}$	$k_{10}\cdot\alpha^{88}$	$k_9\cdot\alpha^{88}$
0	FFH	$k_{12}\cdot\alpha^{175}$	$k_{11}\cdot\alpha^{175}$	$k_{10}\cdot\alpha^{175}$	$k_9\cdot\alpha^{175}$
1	00H	0	0	0	0
1	01H	k_8	k_7	k_6	k_5
1	02H	$k_8\cdot\alpha$	$k_7\cdot\alpha$	$k_6\cdot\alpha$	$k_5\cdot\alpha$
1	03H	$k_8\cdot\alpha^{25}$	$k_7\cdot\alpha^{25}$	$k_6\cdot\alpha^{25}$	$k_5\cdot\alpha^{25}$
1	04H	$k_8\cdot\alpha^{2}$	$k_7\cdot\alpha^{2}$	$k_6\cdot\alpha^{2}$	$k_5\cdot\alpha^{2}$
1	05H	$k_8\cdot\alpha^{50}$	$k_7\cdot\alpha^{50}$	$k_6\cdot\alpha^{50}$	$k_5\cdot\alpha^{50}$
…	…	…	…	…	…
1	FBH	$k_8\cdot\alpha^{234}$	$k_7\cdot\alpha^{234}$	$k_6\cdot\alpha^{234}$	$k_5\cdot\alpha^{234}$
1	FCH	$k_8\cdot\alpha^{168}$	$k_7\cdot\alpha^{168}$	$k_6\cdot\alpha^{168}$	$k_5\cdot\alpha^{168}$
1	FDH	$k_8\cdot\alpha^{80}$	$k_7\cdot\alpha^{80}$	$k_6\cdot\alpha^{80}$	$k_5\cdot\alpha^{80}$
1	FEH	$k_8\cdot\alpha^{88}$	$k_7\cdot\alpha^{88}$	$k_6\cdot\alpha^{88}$	$k_5\cdot\alpha^{88}$
1	FFH	$k_8\cdot\alpha^{175}$	$k_7\cdot\alpha^{175}$	$k_6\cdot\alpha^{175}$	$k_5\cdot\alpha^{175}$

表２

アドレスH	アドレスL	b[31, …, 24]	b[23, …, 16]	b[15, …, 8]	b[7, …, 0]
2	00H	0	0	0	0
2	01H	k_4	k_3	k_2	k_1
2	02H	$k_4 \cdot \alpha$	$k_3 \cdot \alpha$	$k_2 \cdot \alpha$	$k_1 \cdot \alpha$
2	03H	$k_4 \cdot \alpha^{25}$	$k_3 \cdot \alpha^{25}$	$k_2 \cdot \alpha^{25}$	$k_1 \cdot \alpha^{25}$
2	04H	$k_4 \cdot \alpha^{2}$	$k_3 \cdot \alpha^{2}$	$k_2 \cdot \alpha^{2}$	$k_1 \cdot \alpha^{2}$
2	05H	$k_4 \cdot \alpha^{50}$	$k_3 \cdot \alpha^{50}$	$k_2 \cdot \alpha^{50}$	$k_1 \cdot \alpha^{50}$
…	…	…	…	…	…
2	FBH	$k_4 \cdot \alpha^{234}$	$k_3 \cdot \alpha^{234}$	$k_2 \cdot \alpha^{234}$	$k_1 \cdot \alpha^{234}$
2	FCH	$k_4 \cdot \alpha^{168}$	$k_3 \cdot \alpha^{168}$	$k_2 \cdot \alpha^{168}$	$k_1 \cdot \alpha^{168}$
2	FDH	$k_4 \cdot \alpha^{80}$	$k_3 \cdot \alpha^{80}$	$k_2 \cdot \alpha^{80}$	$k_1 \cdot \alpha^{80}$
2	FEH	$k_4 \cdot \alpha^{88}$	$k_3 \cdot \alpha^{88}$	$k_2 \cdot \alpha^{88}$	$k_1 \cdot \alpha^{88}$
2	FFH	$k_4 \cdot \alpha^{175}$	$k_3 \cdot \alpha^{175}$	$k_2 \cdot \alpha^{175}$	$k_1 \cdot \alpha^{175}$
3	00H	0	0	0	0
3	01H	0	0	0	0
3	02H	0	0	0	0
3	03H	0	0	0	0
3	04H	0	0	0	0
3	05H	0	0	0	0
…	…	…	…	…	…
3	FBH	0	0	0	0
3	FCH	0	0	0	0
3	FDH	0	0	0	0
3	FEH	0	0	0	0
3	FFH	0	0	0	0

表３

ベクトル表現	べき表現
00000000	×
00000001	α^{0}
00000010	α^{1}
00000011	α^{25}
00000100	α^{2}
00000101	α^{50}
00000110	α^{26}
00000111	α^{198}
…	…
11111010	α^{244}
11111011	α^{234}
11111100	α^{168}
11111101	α^{80}
11111110	α^{88}
11111111	α^{175}

以下、図２を参照して図１の誤り訂正符号化装置における誤り符号化処理について説明する。

まず、第１番目のクロックCLKの立ち下がりと同期するデコーダ26の出力パルス信号の立ち上がりにおいて、32ビットバス選択器19,23は32ビットラッチ22を選択している。このとき、８ビット排他的論理和演算器18の入力データは、ROM14の出力データの下位８ビットであるデータ00Hと、32ビットラッチ22の出力データの第16ビットから第９ビットにあたるデータ00Hであるから、８ビット排他的論理和演算器18の出力データは00Hになる。そのため、８ビット排他的論理和演算器11の入力データは、８ビット排他的論理和演算器18の出力データであるデータ00Hと、入力データi_{000}とになり、８ビットラッチ12にはデータi_{000}が格納される。従って、図２のデータd_{000}は式（15）によって表わすことができる。これと同時に、８ビットラッチ27の出力データと、ROM14の出力データの上位８ビットの排他的論理和のデータを８ビット排他的論理和演算器15が出力しており、また32ビットラッチ22の出力データの上位24ビットと、ROM14の出力データの下位24ビットとの排他的論理和の24ビットのデータを８ビット排他的論理和演算器16,17,18が出力している。従って、以上の４個の８ビット排他的論理和演算器15,16,17,18の出力データは32ビットラッチ22に書き込まれる。また、それと同時に８ビットラッチ27には、以前32ビットラッチ22に書き込まれていたデータの下位８ビットが書き込まれる。この場合、８ビット排他的論理和演算器15,16,17,18の各８ビットの入力データが全て00Hであるので、32ビットラッチ22に書き込まれるデータは00000000Hである。また、８ビットラッチ27に書き込まれるデ

ータは00Hである。以上の動作の後に、ROM14に与えられるアドレスの下位８ビットはd_{000}になり、その上位２ビットは“3"になる。ここで、ROM14には表１に示すようなデータが格納されており、ガロア体上の積の演算をテーブル参照で求めることができる。この場合、アドレスの上位２ビットは“3"であるので、ROM14はそのデータ端子からデータ00000000Hを出力する。従って、図２のROM14の出力データA_0を８ビットずつ区切って表現すると、次式で表わすことができる。

A_0＝（00H,00H,00H,00H）　　…（17）

次に、第２番目のクロックCLKの立ち下がりと同期するデコーダ28の出力パルス信号の立ち上がりにおいて、８ビットラッチ27が０にクリアされる。また、ROM14に与えられるアドレスの下位８ビットはデータd_{000}のまま変わらず、その上位２ビットは“0"になるため、表１より図２のROM14の出力データA_1は次式で表わすことができる。

$A_1=(k_{12}\cdot d_{000},k_{11}\cdot d_{000},k_{10}\cdot d_{000},k_9\cdot d_{000})$…（18）

ただし、積の記号“・”はガロア体上の積の演算を意味する。

次に、第３番目のクロックCLKの立ち下がりと同期するデコーダ24の出力信号の立ち上がりにおいて、32ビットバス選択器19,23は32ビットラッチ20を選択している。このとき、８ビットラッチ27の８ビットの出力データと、ROM14の出力データの上位８ビットの排他的論理和のデータを８ビット排他的論理和演算器15が出力しており、また32ビットラッチ20の出力データの上位24ビットと、ROM14の出力データの下位24ビットとの排他的論理和のデータを８ビット排他的論理和演算器16,17,18が出力している。従って、以上の８ビット排他的論理和演算器15,16,17,18の32ビットの出力データは、32ビットラッチ20に書き込まれる。また、それと同時に８ビットラッチ27には、以前32ビットラッチ20に書き込まれていたデータの下位８ビットが書き込まれる。この場合、８ビットラッチ27の出力データはデータ00Hであり、また32ビットラッチ20の出力データの上位24ビットもデータ000000Hであるので、32ビットラッチ20に書き込まれるデータは式（18）で表されるデータA_1そのものである。従って、図２の32ビットラッチ20の出力データである（$p_{000},p_{001},p_{002},p_{003}$）は次式で表わすことができる。

$(p_{000},p_{001},p_{002},p_{003})$

$=(k_{12}\cdot d_{000},k_{11}\cdot d_{000},k_{10}\cdot d_{000},k_9\cdot d_{000})$　…（19）

また、８ビットラッチ27に書き込まれるデータはデータ00Hである。以上の動作の後に、ROM14に与えられるアドレスの下位８ビットはデータd_{000}となり、その上位２ビットは“1"になる。従って、図２のROM14の出力データA_2は次式で表わすことができる。

$A_2=(k_8\cdot d_{000},k_7\cdot d_{000},k_6\cdot d_{000},k_5\cdot d_{000})$…（20）

同様にして、データ（$p_{004},p_{005},p_{006},p_{007}$）は次式で表わすことができる。

$(p_{004},p_{005},p_{006},p_{007})$

$=(k_8\cdot d_{000},k_7\cdot d_{000},k_6\cdot d_{000},k_5\cdot d_{000})$　…（21）

また、ROM14の出力データA_3は次式で表わすことができる。

$A_3=(k_4\cdot d_{000},k_3\cdot d_{000},k_2\cdot d_{000},k_1\cdot d_{000})$…（22）

また、データ（$p_{008},p_{009},p_{010},p_{011}$）は次式で表わすことができる。

$(p_{008},p_{009},p_{010},p_{011})$

$=(k_4\cdot d_{000},k_3\cdot d_{000},k_2\cdot d_{000},k_1\cdot d_{000})$　…（23）

ただし、第５番目のクロックCLKの立ち下がりと同期して８ビットラッチ12の出力データが変化する。具体的にはデコーダ26の出力パルス信号の立ち上がりにおいて、32ビットバス選択器19,23が32ビットラッチ22を選択しているため、第１番目のクロックCLKの時と同様に、８ビット排他的論理和演算器11の入力データは８ビット排他的論理和演算器18の出力データであるp_{011}と、入力データi_{001}になり、８ビットラッチ12には、式（15）で表されるデータd_{001}が格納される。ただし、式（15）において和の記号はガロア体上の和の演算を意味する。

次に、第７番目のクロックCLKの立ち下がりと同期するデコーダ24の出力信号の立ち上がりにおける状態を説明する。このとき、32ビットバス選択器19,23は32ビットラッチ20を選択している。上記の処理と同様に考えると、８ビットラッチ27の出力データは00Hであ

り、また32ビットラッチ20の出力データの上位24ビットは、

$(p_{000},p_{001},p_{002})$

$=(k_{12}\cdot d_{000},k_{11}\cdot d_{000},k_{10}\cdot d_{000})$ …（24）

であるので、

$(p_{012},p_{013},p_{014},p_{015})$

$=(k_{12}\cdot d_{001},k_{11}\cdot d_{001}+p_{000},$

$k_{10}\cdot d_{001}+p_{001},k_{9}\cdot d_{001}+p_{002})$

$=(k_{12}\cdot d_{001},k_{11}\cdot d_{001}+k_{12}\cdot d_{000},k_{10}\cdot d_{001}$

$+k_{11}\cdot d_{000},k_{9}\cdot d_{001}+k_{10}\cdot d_{000})$ …（25）

で表されるデータが32ビットラッチ20に書き込まれる。

また、32ビットラッチ20の書き込み前のデータ値の下位８ビットであるデータp_{003}が同時に８ビットラッチ27に書き込まれる。以上の動作の後に、ROM14に与えられるアドレスの下位８ビットはデータd_{001}となり、その上位２ビットは“1"になる。

従って、図２のROM14の出力データB_2は、次式で表わすことができる。

$B_2=(k_8\cdot d_{001},k_7\cdot d_{001},k_6\cdot d_{001},k_5\cdot d_{001})$ …（26）

同様にして、データ（$p_{016},p_{017},p_{018},p_{019}$）は次式で表わすことができる。

$(p_{016},p_{017},p_{018},p_{019})$

$=(k_8\cdot d_{001}+p_{003},k_7\cdot d_{001}+p_{004},$

$k_6\cdot d_{001}+p_{005},k_5\cdot d_{001}+p_{006})$

$=(k_8\cdot d_{001}+k_9\cdot d_{000},k_7\cdot d_{001}+k_8\cdot d_{000},$

$k_6\cdot d_{001}+k_7\cdot d_{000},k_5\cdot d_{001}+k_6\cdot d_{000})$ …（27）

また、ROM14の出力データB_3は次式で表わすことができる。

$B_3=(k_4\cdot d_{001},k_3\cdot d_{001},k_2\cdot d_{001},k_1\cdot d_{001})$ …（28）

さらに、データ（$p_{020},p_{021},p_{022},p_{023}$）は次式で表わすことができる。

$(p_{020},p_{021},p_{022},p_{023})$

$=(k_4\cdot d_{001}+p_{007},k_3\cdot d_{001}+p_{008},$

$k_2\cdot d_{001}+p_{009},k_1\cdot d_{001}+p_{010})$

$=(k_4\cdot d_{001}+k_5\cdot d_{000},k_3\cdot d_{001}+k_4\cdot d_{000},$

$k_2\cdot d_{001}+k_3\cdot d_{000},k_1\cdot d_{001}+k_2\cdot d_{000})$ …（29）

また、第９番目のクロックCLKの立ち下がりと同期するデコーダ26の出力信号の立ち上がりにおいて、８ビット排他的論理和演算器18の出力データはp_{023}であるため、データd_{002}は次式で表わすことができる。

$d_{002}=i_{002}+p_{023}=i_{002}+k_1\cdot d_{001}$ …（30）

以上の動作を第｛４×20＋１｝番目のクロックCLKが入力されるまで繰り返すと、32ビットラッチ20,21,22に合計12シンボルのパリティ語が生成される。今まで入力した20シンボルの入力データの情報シンボル、すなわち情報語に、ここで生成した12シンボルのパリティ語を付加すると、合計32シンボルの符号語、つまり出力シンボル列が完成する。従って、遅延回路300及び出力データ選択器310とにより、20シンボルの情報語に続いて、12シンボルのパリティ語が８ビットずつパラレルに順次、誤り訂正符号化された符号語（＝情報語＋パリテイ語）の出力データとして出力される。

以上説明したように、本実施形態では、

（ａ）情報語である８ビットの入力データに応答して、生成多項式の各係数と入力情報シンボルのガロア体上の複数の積データをそれぞれ演算して、各アドレスに対して４個の積データを１組として記憶し、４個の積データを１組として並列に（同時に）排他的論理和演算器15乃至18に対して読み出し可能に構成された積データ記憶手段である32ビットROM14と、

（ｂ）第１の記憶手段である３個の32ビットラッチ20,21,22と、（ｃ）第２の記憶手段である８ビットラッチ27と、

（ｄ）排他的論理和演算手段である８ビット排他的論理和演算器15,16,17,18と、

（ｅ）第１の選択手段である32ビットバス選択器19,20と、

（ｆ）読み出し制御手段である４進カウンタ13、ラッチ12,29及びデコーダ24乃至26,28と、
（ｇ）第２の選択手段であるシンボル遅延回路300及び出力データ選択器310と
を用いることによって、最小符号間距離ｄ＝13であるリード・ソロモン符号RS（n,n－12,d＝13）を符号化できる。
以上の第１の実施形態においては、複数の積データを記憶したROM14から４個の積データを１組として並列に（同時に）排他的論理和演算器15乃至18に対して読み出し可能に構成し、４個の積データを用いて同時に処理し、かつ３個の32ビットラッチ20乃至22を選択的に順次繰り返して使用することにより、符号化処理の演算をしているので、回路構成を、図14の従来技術の誤り訂正符号化装置の回路構成に比較して極めて簡単にすることができるとともに、効率的にかつ高速で演算することができ、実用に供することが可能な符号化速度で誤り訂正符号を符号化することができる。
第１の実施形態において、第１の記憶手段を構成する32ビットラッチ20,21,22の数を３以上の数ｍに増やし、それに伴ってROM14の内容を変更すれば、回路構成を変更することなく、同様な回路でさらに最小符号間距離ｄの長い誤り訂正符号も符号化できる。
＜第２の実施形態＞
図３は、本発明に係る第２の実施形態の誤り訂正符号化装置の構成を示すブロック図である。図３において、この第２の実施形態の誤り訂正符号化装置は、
（ａ）32ビットデータバス81を有し当該装置の動作を制御するCPU（中央演算制御装置）61と、
（ｂ）CPU61からアドレスバス82を介して出力されるアドレスデータと、メモリ制御信号とに応答して、キューメモリ72における複数ｍ個のラッチ72－1,72－2,…,72－ｍに対する制御信号を発生するラッチ制御装置62と、
（ｃ）CPU61のワークエリアとして用いられる作業用RAM63と、
（ｄ）８ビット排他的論理和演算器64,65,66,67と、
（ｅ）32ビットバス選択器70と、
（ｆ）例えばEPROM又はEEPROMにてなる書き換え可能なROM（読み出し専用メモリ）であって、CPU61によって実行される符号化処理のプログラム及びそれを実行するために必要なデータを予め格納するROM71と、
（ｇ）複数ｍ個の32ビットラッチ72－１乃至72－ｍを備えて、FIFO（First－in First－out）メモリであるシフトレジスタを構成する第１の記憶手段であるキューメモリ72と、
（ｈ）CPU61の制御により、キューメモリ72の内容を一時的にラッチして読み出す32ビットラッチ69と、
（ｉ）32ビットラッチ69の一部分であって、32ビットラッチ69における下位８ビットのデータをラッチする第２の記憶手段である８ビットラッチ68とを備える。
第２の実施形態では、第１の実施形態のバス選択器19,23に代わりに、複数ｍ個の32ビットラッチ72－１乃至72－ｍを備えた32ビットｍ段のキューメモリ72を用いている。第２の実施形態の動作例の説明を簡潔にするために、キューメモリ72を構成するラッチ段数ｍを３とし、キューメモリ72は３個の32ビットラッチ73,74,75からなる。図３においてｍ＝３のときのブロック図を図４に示す。従って、図４を参照して当該動作例について説明する。
図４において、CPU61は32ビットデータバス81に接続され、作業用RAM63も32ビットデータバス81に接続される。入力されて処理対象とされる情報語のデータは、32ビットデータバス81を介して作業用RAM63内の符号語バッファBufB［］の情報語領域に書き込まれ、当該領域から32ビットデータベス81を介して読み出されて８ビット排他的論理和演算器64,65,66,67の第１の入力端子に入力されて、以下に詳述する符号化処理が実行された後、上記情報語のデータに続いて、作業用RAM63内の符号語バッファBufB［］のパリティ語領域からパリティ語を読み出して符号化された符号語として出力する。
図４において、８ビット排他的論理和演算器64,65,66,67から出力されるデータは32ビットのキューメモリ72内で縦続接続された３個の32ビットラッチ73,74,75及び32ビットラッ

チ69を介して32ビットバス選択器70のｂ接点に入力される。32ビットラッチ75から出力される32ビットの出力データのうち上位24ビットのデータは、排他的論理和演算器65,66,67の第２の入力端子に入力され、また、８ビットラッチ68からの出力データは排他的論理和演算器65の第２の入力端子に入力される。また、32ビットラッチ73から出力される32ビットの出力データは、32ビットバス選択器70のａ接点に入力される。32ビットバス選択器70は、ラッチ制御回路62から出力される制御信号に応答して、ａ接点に入力される32ビットのデータと、ｂ接点に入力される32ビットのデータとのうちの１つの32ビットのデータを選択的に32ビットデータバス81を介してCPU61及び作業用RAM63に出力する。

図４において、縦続接続された３個の32ビットラッチ73,74,75はキューメモリ72を構成しており、32ビットラッチ73は書き込み用ラッチとして動作する一方、32ビットラッチ75は読み出し用ラッチとして動作するように選択される。当該選択は、物理的には固定であるが、キューメモリ72へのデータの書き込み動作、もしくはキューメモリ72からのデータの読み出し動作によって、各32ビットラッチに格納されているデータが次段の32ビットラッチに移動して行くので、論理的には固定ではなく、順番に32ビットラッチ内のデータを選択して行くことになる。本実施形態では、CPU61から入出力制御信号（以下、I/O制御信号という。）とともに、アドレスバス82を介して入出力アドレス（以下、I/Oアドレスという。）をラッチ制御回路62に送信することにより、ラッチ制御回路62は、以下に示す制御信号をラッチ73,74,75,69及びバス選択器70に送信して制御する。

（ａ）CPU61がアドレスデータバス82を介してI/Oアドレスとして“AdrsQueue"をラッチ制御回路62に出力するとともに、I/O制御信号として“書き込み信号”をラッチ制御回路62に出力する。これに応答して、ラッチ制御回路62は、クロックCLKに同期する書き込みクロック信号を発生して32ビットラッチ73,74,75,69に出力する。

（ｂ）CPU61がアドレスデータバス82を介してI/Oアドレスとして“AdrsQueue"をラッチ制御回路62に出力するとともに、I/O制御信号として“読み出し信号”をラッチ制御回路62に出力する。これに応答して、ラッチ制御回路62は、クロックCLKに同期する書き込みクロック信号を発生して32ビットラッチ73,74,75,69に出力する。

（ｃ）CPU61がアドレスデータバス82を介してI/Oアドレスとして“AdrsFeed"をラッチ制御回路62に出力するとともに、I/O制御信号として“書き込み信号”をラッチ制御回路62に出力する。これに応答して、ラッチ制御回路62は、クロックCLKに同期する書き込みクロック信号を発生して32ビットラッチ73,74,75に出力する。

（ｄ）CPU61がアドレスデータバス82を介してI/Oアドレスとして“AdrsClearPort"をラッチ制御回路62に出力するとともに、I/O制御信号として“書き込み信号”をラッチ制御回路62に出力する。これに応答して、ラッチ制御回路62はクリア信号を32ビットラッチ69に出力して、32ビットラッチ69に記憶されているデータを０にクリアする。

（ｅ）CPU61がアドレスデータバス82を介してI/Oアドレスとして“AdrsQueue"をラッチ制御回路62に出力するとともに、I/O制御信号として“アクセス信号”をラッチ制御回路62に出力する。これに応答して、ラッチ制御回路62は、バス選択器70をａ接点に切り換える。これにより、32ビットラッチ73から出力されるデータは、バス選択器70のａ接点及び32ビットデータバス81を介してCPU61及び作業用RAM63に入力される。

（ｆ）CPU61がアドレスデータバス82を介してI/Oアドレスとして“AdrsQueueLast"をラッチ制御回路62に出力するとともに、I/O制御信号として“アクセス信号”をラッチ制御回路62に出力する。これに応答して、ラッチ制御回路62は、バス選択器70をｂ接点に切り換える。これにより、32ビットラッチ69から出力されるデータは、バス選択器70のｂ接点及び32ビットデータバス81を介してCPU61及び作業用RAM63に入力される。

図５は、図３の誤り訂正符号化装置のCPU61によって実行される符号化処理を示すフローチャートであり、図６は、図５のサブルーチンである初期化処理（ステップS1）を示すフローチャートであり、図５は、図５のサブルーチンである符号語生成処理（ステップS2）を示すフローチャートである。また、図８は、図７のサブルーチン処理P1（ステップS7）を示すフローチャートであり、図９は、図７及び図８のサブルーチン処理P2（ステップS23,S35）を示すフローチャートであり、図10は、図８のサブルーチン処理P3（ステップS37

）を示すフローチャートであり、図11は、図７のサブルーチン処理P4（ステップS24）を示すフローチャートである。図５乃至図11に図示された符号化処理の制御フローのプログラムは、図３及び図４に示す誤り訂正符号化装置を動作させるために、CPU61に接続されたROM1に予め記憶され、CPU61によって実行される。

図５に示すように、まず、ステップS1で図６に示す初期化処理を実行した後、ステップS2で図７に示す符号語生成処理を実行して、当該符号化処理を終了する。

次いで、図６乃至図11における処理で用いる種々の記号について説明する。テーブル名の［］及びバッファ名の［］はそれぞれ配列を表し、メモリ領域の名前をnameとすると、一次元配列の要素はname［ｍ］、二次元配列の要素はname［ｍ］［ｎ］のように表し、ここで、ｍ及びｎはそれぞれ０以上の整数である。また、配列の大きさを表すときにname［Ｍ］［Ｎ］と書くとすると、有効な要素はname［０］［０］からname［Ｍ－１］［Ｎ－１］である。

IOReadW（address）は、32ビットのワードデータをaddressで表されるI/Oアドレスから読み出したデータ値を表す関数である。また、IOWriteW（address,data）はdataで表される32ビットのワードデータをaddressで表されるIOアドレスに書き込む処理である。GetByte（data,index）はdataで表される32ビットのワードデータの上位からindex番目のバイトデータを表す関数である。ただし、indexは１から数える自然数である。

スラッシュ記号“/"は整数の除算を表わす二項演算子であり、Ａ　MOD　ＢはＡをＢで除算したときの剰余を演算する二項演算子である。また、ANDはビットごとの論理積を演算する二項演算子であり、EORはビットごとの排他的論理和を演算する二項演算子である。

さらに、配列名をnameとすると、name［x,…,y］は配列nameのｘ番目の要素からｙ番目の要素までを表す。変数名又は配列名の最後の１文字であるW,Bは、それぞれ32ビットのデータ、８ビットのデータを表す。

ガロア体演算テーブルtGaloisW［nQueueLatch］［256］は、入力データと生成多項式の各係数とのガロア体上の積のデータであって、予め演算されてROM71に格納された後、CPU61の初期化時や、誤り訂正符号の最小符号化距離ｄの変更に伴う初期化処理において、ROM71から作業用RAM63内の領域に転送されて書き込まれる。第１の実施形態と同様の12シンボルのパリティ語を付加する場合では、表１及び表２に示したROM14の内容と同一のデータが当該ガロア体演算テーブルtGaloisW［nQueueLatch］［256］に予め記憶される（ステップS11）。ただし、tGaloisW［ｍ］［ｎ］は、第１の実施形態におけるROM14のアドレスＨをｍとし、アドレスＬをｎとしたデータ内容を表す。

符号語バッファBufB［］は、作業用RAM63内の領域に設定されたバッファメモリであり、情報語領域とパリティ語領域とに分割され、当該符号語バッファBufB［］の情報語領域において、この符号化処理を実行する前に入力された符号化すべき情報語が先頭から予め格納される一方、符号語バッファBufB［］のパリティ語領域において、上記入力された情報語に基づいて当該符号化処理を実行したときに得られるパリティ語が格納された後、情報語とパリティ語が符号語として外部装置に出力される。

I/OアドレスのAdrsQueueLastは、32ビットラッチ73の内容を読み出すためのアドレスである。ここで、CPU61がI/O制御信号の“読み出し信号”とともに、I/Oアドレスとして“AdrsQueueLast"をラッチ制御回路62に出力した場合であっても、上述のように、32ビットラッチ73,74,75,69に対して書き込みクロック信号を発生しないので、キューメモリ72内の記憶データの内容はそのままである。

表４は、図５乃至図11のフローチャートに従って本実施形態の誤り訂正符号化装置が動作する場合において、本動作例での図４の各32ビットラッチ73,74,75,69の記憶データを示す。表４における二重下線はデータの書き込みを表し、下線はデータの読み込みを表す。また、p_{000}から始まるパリティシンボルの記号は、第１の実施形態のものと同一であり、図13に示す従来技術の計算方法で演算される値である。

表４

ｽﾃｯﾌﾟ	cnt	cntQ	ラッチ73	ラッチ74	ラッチ75	ラッチ69
S1001	-	-	00000000	00000000	00000000	00000000
S1002	0	-	00000000	00000000	00000000	00000000
S1003	0	-	00000000	00000000	00000000	00000000
S1004	0	3	(p000, …, p003)	00000000	00000000	00000000
S1005	0	2	(p004, …, p007)	(p000, …, p003)	00000000	00000000
S1006	0	1	(p008, …, p011)	(p004, …, p007)	(p000, …, p003)	00000000
S1007	1	0	(p008, …, p011)	(p004, …, p007)	(p000, …, p003)	00000000
S1008	1	0	(p008, …, p011)	(p004, …, p007)	(p000, …, p003)	00000000
S1009	1	3	(p012, …, p015)	(p008, …, p011)	(p004, …, p007)	(p000, …, p003)
S1010	1	2	(p106, …, p019)	(p012, …, p015)	(p008, …, p011)	(p004, …, p007)
S1011	1	1	(p020, …, p023)	(p016, …, p019)	(p012, …, p015)	(p008, …, p011)
S1012	2	0	(p020, …, p023)	(p016, …, p019)	(p012, …, p015)	(p008, …, p011)
S1013	2	0	(p020, …, p023)	(p016, …, p019)	(p012, …, p015)	00000000
S1014	2	3	(p024, …, p027)	(p020, …, p023)	(p016, …, p019)	(p012, …, p015)
S1015	2	2	(p028, …, p031)	(p024, …, p027)	(p020, …, p023)	(p016, …, p019)
S1016	2	1	(p032, …, p035)	(p028, …, p031)	(p024, …, p027)	(p020, …, p023)
S1017	3	0	(p032, …, p035)	(p028, …, p031)	(p024, …, p027)	(p020, …, p023)
S1018	3	0	(p032, …, p035)	(p028, …, p031)	(p024, …, p027)	00000000
…	…	…	…	…	…	…
S1102	20	0	(p236, …, p239)	(p232, …, p235)	(p228, …, p231)	(p224, …, p227)
S1103	20	0	xxxxxxxx	(p236, …, p239)	(p232, …, p235)	(p228, …, p231)
S1104	20	0	xxxxxxxx	xxxxxxxx	(p236, …, p239)	(p232, …, p235)
S1105	20	0	xxxxxxxx	xxxxxxxx	xxxxxxxx	(p236, …, p239)

以下に、第１の実施形態と同様に、本実施形態を用いて８ビットを１シンボルとしたリード・ソロモン符号RS（32,20,d＝13）を符号化する処理例（以下、第１の動作例という。）を、図６乃至図11及び表４を参照して説明する。本動作例の場合、情報シンボル数nInfoは20であり、パリティシンボル数nParityは12である。

まず、図６の初期化処理について以下説明する。

図６のステップS11において、ガロア体演算テーブルtGalois［nQueueLatch］［256］の初期化を行う。本動作例の場合、キューメモリ72を構成する32ビットラッチの数nQueueLatc

h（＝ｍ）は３である。本動作例のように最小符号間距離ｄが13の場合、ガロア体演算テーブルtGalois［nQueueLatch］［256］の内容は、表１と表２に示す第１の実施形態におけるROM14のデータ内容と同一になるようにROM71から作業用RAM63に転送されて初期化される。また、32ビットラッチ73,74,75,69は０にクリアされる。この処理は表４のステップS1001に対応する。

次いで、ステップS12において、

nFeed＝nQueueLatch－（nParity＋３）/4　…（31）

で表すことができるキューメモリ72のフィード回数nFeedを演算して、フィード回数パラメータnFeedに代入される。本動作例の場合、キューメモリ72を構成する32ビットラッチの数nQueueLatchは３であり、付加するパリティシンボル数nParityは12であるから、キューメモリ72のフィード回数nFeedは０に初期化される。次いで、ステップS13において、

nQ＝（nParity＋３）/4　…（32）

で表すことができるキューメモリ72の書き込み数nQを演算して書き込み数パラメータnQに代入される。本動作例の場合、付加するパリティシンボル数nParityは12であるから、キューメモリ72の書き込み数nQは３になる。

次いで、ステップS14において、

index＝（（nParity＋３）MOD4）＋１　…（33）

で表すことができるキュー溢れシンボル取り出し位置indexを演算して、キュー溢れシンボル取り出し位置パラメータindexに代入する。ここで、キュー溢れシンボル取り出し位置indexは、キューメモリ72を８ビット×nParity段のキューとして動作させるために、擬似的にキューメモリ72のnParity段目から８ビットシンボルが溢れたように取り出す位置を示す。本動作例の場合、付加するパリティシンボル数nParityは12であるから、キュー溢れシンボル取り出し位置indexは４になる。以上で初期化処理を終了する。

次に、図７から図11を参照して実際の符号語生成処理を説明する。

まず、図７のステップS21において、作業用RAM63内の符号語バッファBufB［］から情報語を読み出し、当該符号語生成処理によって生成したパリティ語をキューメモリ72上に作成する。この処理をサブルーチン処理P1とする。

次に、サブルーチン処理P1の内容を図８を参照して具体的に説明する。まず、図８のステップS31において、情報シンボルカウンタcntを０に初期化し、ステップS32において、符号語バッファから情報シンボルカウンタcnt番目の８ビットの情報シンボルBufB［cnt］を読み出して入力データinDataBとする。この場合、情報シンボルカウンタcntは０であるから、入力データinDataBは、符号語バッファの先頭のシンボルである。以後、ここで取り出した先頭の情報シンボルをi_{000}とする。次いで、ステップS33において、入力ポートである32ビットラッチ73から32ビットのデータを読み出し、その上位からindex（本動作例の場合は４）バイト目、つまり下位８ビットを得る。この場合、32ビットラッチ73の内容は、クリアされて00000000Hであるので、その下位８ビットは00Hである。これは表４のステップS1002に対応する。さらに、ステップS34において、出力ポートである32ビットラッチ69の内容を０にクリアする。これは表４のステップS1003に対応する。

次いで、ステップS35において、キューメモリ72の内容を、キューメモリ72のフィード回数nFeed回だけ進める。当該処理においては、32ビットラッチ75のデータを32ビットラッチ69に書き込み、32ビットラッチ74のデータを32ビットラッチ75に書き込み、32ビットラッチ73のデータを32ビットラッチ74に書き込む。ここで、フィード回数nFeed回だけ進めるとは、上記ラッチからラッチへの書き込み処理をnFeed回だけ繰り返すことを意味する。ここで、入力ポートである32ビットラッチ73のデータはそのまま保存される。この処理をサブルーチン処理P2とする。

次に、サブルーチン処理P2の内容を図９を参照して具体的に説明する。まず、図９のステップS41において、キューメモリ72のフィードカウンタの計数値cntFeedをキュー72のフィード回数nFeedに初期化する。本動作例の場合、フィード回数nFeedは０であるので、計数値cntFeedは０になる。次いで、ステップS42において、キューメモリ72のフィードカウンタの計数値cntFeedは０であるから、サブルーチン処理P2を終了して図８の元のサブルー

チン処理P1に戻る。つまり、本動作例の場合、サブルーチン処理P2では何も実行しない。
図８のステップS36において、入力データinDataBとステップS33で求めた32ビットラッチ73の下位８ビットのデータpDataBとの排他的論理和を求め、演算結果のデータdataBの値をd_{000}とする。この場合、入力データi_{000}とデータ00Hとの排他的論理和なので、第１の実施形態と同様に、演算結果のデータd_{000}は式（15）で表される。次いで、ステップS37において、ステップS36で求めた演算結果のデータdataBと生成多項式の各係数との積を作業用RAM63内のガロア体演算テーブルから読み出してキューメモリ72に書き込む。この処理をサブルーチン処理P3とする。
次に、サブルーチン処理P3の内容を図10を参照して具体的に説明する。まず、図10のステップS51において、キュー書き込みカウンタの計数値cntWQを０に初期化する。次いで、ステップS52において、ステップS36で求めたデータd_{000}と生成多項式（式（14））の係数とのガロア体上の積を、４シンボル（32ビット）だけ作業用RAM63内のガロア体演算テーブルから読み出す。ここで、キュー書き込みカウンタの計数値cntWQが０の時には、生成多項式（式（14））の係数としてk_{12},k_{11},k_{10},k_9が選ばれる。次いで、ステップS53において、ステップS52で読み出したガロア体上の積のデータをキューメモリ72に書き込む。このとき、32ビットラッチ75は、データ00000000Hを出力しており、８ビットラッチ68はデータ00Hを出力しているので、排他的論理和演算器64,65,66,67の入力データはデータ（00H,00H,00H,00H）と、データ（k_{12}・d_{000},k_{11}・d_{000},k_{10}・d_{000},k_9・d_{000}）である。従って、排他的論理和演算器64,65,66,67の出力データを（p_{000},p_{001},p_{002},p_{003}）すると、第１の実施形態と同様に、このデータ値は式（19）で表すことができる。これは表４のステップS1004に対応する。
次いで、ステップS54において、キュー書き込みカウンタの計数値cntWQを１だけインクリメントする。このとき、キュー書き込みカウンタの計数値cntWQは１になる。そして、ステップS55において、キュー書き込みカウンタの計数値cntWQがキュー書き込み数nQより小さい場合は、ステップS52に戻る一方、そうでない場合はサブルーチン処理P3を終了してリターンする。この場合、キュー書き込みカウンタの計数値cntWQは１であり、キュー書き込み数nQは３であるから、ステップS52に戻る。
次いで、ステップS52において、ステップS36で求めたデータd_{000}と生成多項式（式（14））の係数とのガロア体上の積を、４シンボル（32ビット）求める。その際に、ガロア体演算テーブルを用いるが、キュー書き込みカウンタの計数値cntWQが１の時には、生成多項式（式（14））の係数としてk_8,k_7,k_6,k_5が選ばれる。さらに、ステップS53において、ステップS52で求めた積のデータをキューメモリ72に書き込む。このとき、32ビットラッチ75はデータ00000000Hを出力しており、８ビットラッチ68はデータ00Hを出力しているので、排他的論理和演算器64,65,66,67の入力データは、（00H,00H,00H,00H）と、データ（k_8・d_{000},k_7・d_{000},k_6・d_{000},k_5・d_{000}）である。従って、排他的論理和演算器64,65,66,67の出力を（p_{004},p_{005},p_{006},p_{007}）とすると、第１の実施形態と同様に、このデータ値は式（21）で表すことができる。これは表４のステップS1005に対応する。
次いで、ステップS54において、キュー書き込みカウンタの計数値cntWQを１だけインクリメントする。このとき、キュー書き込みカウンタの計数値cntWQは２になる。そして、ステップS55において、キュー書き込みカウンタの計数値cntWQがキュー書き込み数nQより小さい場合は、ステップS52に戻る一方、そうでない場合はサブルーチン処理P3を終了してリターンする。この場合、キュー書き込みカウンタの計数値cntWQは２であり、キュー書き込み数nQは３であるから、ステップS52に戻るが、その後、上記の処理と同様にステップS52、ステップS53、ステップS54を経て、再びステップS55に進む。この時、キュー書き込みカウンタの計数値cntWQは３であるから、当該サブルーチン処理P3を終了して、図８のサブルーチン処理P1に戻る。
図８のステップS38において、情報シンボルカウンタの計数値cntを１だけインクリメントする。このとき、情報シンボルカウンタの計数値cntは１になる。そして、ステップS39において、情報シンボルカウンタの計数値cntが情報シンボル数nInfoより小さい場合はステップS32に戻る一方、そうでない場合はサブルーチン処理P1を終了してリターンする。こ

の場合、情報シンボルカウンタの計数値cntは１であり、かつ情報シンボル数nInfoは20であるから、ステップS32に戻る。
ステップS32において、符号語バッファから情報シンボルカウンタの計数値cnt番目の情報シンボルBufB［cnt］を読み出す。この場合、情報シンボルカウンタの計数値cntは１であるから、符号語バッファの先頭から２番目のシンボルである。ここで読み出した２番目の情報シンボルをi_{001}とする。次いで、ステップS33において、32ビットラッチ73から32ビットのデータを読み出し、その上位からindex（本動作例の場合は４）バイト目、つまり下位８ビットのデータを得る。この場合、32ビットラッチ73内のデータはデータ（p_{008},p_{009},p_{010},p_{011}）であるので、その下位８ビットのデータはp_{011}である。これは表４のステップS1007に対応する。さらに、ステップS34において、32ビットラッチ69の内容をクリアする。これは表４のステップS1008に対応する。
次いで、ステップS35において、キューメモリ72の内容を、キューメモリ72のフィード回数nFeed回だけ進めるが、上記の処理と同様に、本動作例の場合、当該サブルーチン処理P2において何も実行しない。さらに、ステップS36において、入力データinDataBとステップS33で求めた32ビットラッチ73の下位８ビットのデータとの排他的論理和を演算し、演算結果のデータ値をd_{001}とすると、この場合、入力データi_{001}とデータp_{011}との排他的論理和なので、第１の実施形態と同様に、演算結果のデータd_{001}は式（16）で表される。さらに、ステップS37において、ステップS36で求めたデータ値と生成多項式の各係数との積を、キューメモリ72に書き込む処理であるサブルーチン処理P3を実行する。
次に、サブルーチン処理P3の内容を図10を参照して具体的に説明する。まず、図10のステップS51において、キュー書き込みカウンタの計数値cntWQを０に初期化し、ステップS52において、ステップS36で求めたデータd_{001}と生成多項式（式（14））の係数とのガロア体上の積を、４シンボル（32ビット）求める。その際に、ガロア体演算テーブルを用いるが、キュー書き込みカウンタの計数値cntWQが０の時には、生成多項式（式（14））の係数としてk_{12},k_{11},k_{10},k_9が選ばれる。そして、ステップS53において、ステップS52で求めた結果のデータをキューメモリ72に書き込む。このとき、32ビットラッチ75はデータ（p_{000},p_{001},p_{002},p_{003}）を出力しており、８ビットラッチ68はデータ00Hを出力しているので、排他的論理和演算器64,65,66,67の入力データはデータ（00H,p_{000},p_{001},p_{002}）とデータ（k_{12}・d_{001},k_{11}・d_{001},k_{10}・d_{001},k_9・d_{001}）である。従って、排他的論理和演算器64,65,66,67の出力データをデータ（p_{012},p_{013},p_{014},p_{015}）とすると、第１の実施形態と同様に、このデータ値は式（25）で表すことができる。これは表４のステップS1009に対応する。
次いで、ステップS54において、キュー書き込みカウンタの計数値cntWQを１だけインクリメントする。このとき、キュー書き込みカウンタの計数値cntWQは１になる。さらに、ステップS55において、キュー書き込みカウンタの計数値cntWQがキュー書き込み数nQより小さい場合はステップS52に戻る一方、そうでない場合は当該サブルーチン処理P3を終了してリターンする。この場合、キュー書き込みカウンタの計数値cntWQは１であり、キュー書き込み数nQは３であるから、ステップS52に戻る。
ステップS52において、ステップS36で求めたデータd_{001}と生成多項式（式（14））の係数とのガロア体上の積を、４シンボル（32ビット）求める。その際に、ガロア体演算テーブルを用いるが、キュー書き込みカウンタの計数値cntWQが１の時には、生成多項式（式（14））の係数としてk_8,k_7,k_6,k_5が選ばれる。そして、ステップS53において、ステップS52で求めた結果データをキューメモリ72に書き込む。このとき、32ビットラッチ75はデータ（p_{004},p_{005},p_{006},p_{007}）を出力しており、８ビットラッチ68はデータp_{003}を出力しているので、排他的論理和演算器64,65,6,67の入力データはデータ（p_{003},p_{004},p_{005},p_{006}）とデータ（k_8・d_{001},k_7・d_{001},k_6・d_{001},k_5・d_{001}）である。従って、排他的論理和演算器64,65,66,67の出力データをデータ（p_{016},p_{017},p_{018},p_{019}）とすると、第１の実施形態と同様に、このデータ値は式（27）で表すことができる。これは表４のステップS1010に対応する。
次いで、ステップS54において、キュー書き込みカウンタの計数値cntWQを１だけインクリ

メントする。このとき、キュー書き込みカウンタの計数値cntWQは２になる。次いで、ステップS55において、キュー書き込みカウンタの計数値cntWQがキュー書き込み数nQより小さい場合はステップS52に戻る一方、そうでない場合はサブルーチン処理P3を終了してリターンする。この場合、キュー書き込みカウンタの計数値cntWQは２であり、キュー書き込み数nQは３であるから、ステップS52に戻る。その後、上記の処理と同様にステップS52、ステップS53、ステップS54を経て、再びステップS55に進む。この時、キュー書き込みカウンタの計数値cntWQは３であるから、当該サブルーチン処理P3を終了して、図８のサブルーチン処理P1にリターンする。

図８のステップS38において、情報シンボルカウンタの計数値cntを１だけインクリメントする。このとき、情報シンボルカウンタの計数値cntは２になる。次いで、ステップS39において、情報シンボルカウンタの計数値cntが情報シンボル数nInfoより小さい場合はステップS32に戻り、そうでない場合はサブルーチン処理P1を終了してリターンする。この場合、情報シンボルカウンタの計数値cntは２であり、情報シンボル数nInfoは20であるから、ステップS32に戻る。

ステップS32において、符号語バッファから情報シンボルカウンタの計数値cnt番目の情報シンボルBufB［cnt］を読み出す。この場合、情報シンボルカウンタの計数値cntは２であるから、符号語バッファの先頭から３番目のシンボルである。ここで取り出した３番目の情報シンボルをデータi_{002}とする。そして、ステップS33において、32ビットラッチ73から32ビットのデータを読み出し、その上位からindex（本動作例の場合は４）バイト目、つまり下位８ビットを得る。この場合、32ビットラッチ73の内容はデータ（p_{020},p_{021},p_{022},p_{023}）であるので、その下位８ビットのデータはp_{023}である。これは表４のステップS1012に対応する。次いで、ステップS34において、出力ポートの32ビットラッチ69の内容をクリアする。これは表４のステップS1013に対応する。そして、ステップS35において、キューメモリ72の内容を、キューメモリ72のフィード回数nFeed回だけ進めるが、上記の処理と同様に、本動作例の場合、当該サブルーチン処理P2において何も実行しない。

次いで、ステップS36において、入力データとステップS33で求めた32ビットラッチ73の下位８ビットのデータとの排他的論理和を求める。このデータ値をd_{002}とすると、この場合、入力データi_{002}とデータp_{023}との排他的論理和なので、第１の実施形態と同様に、データd_{002}は式（30）で表される。以下、上記の処理と同様に、ステップS37、S38、S39と進み、さらに、情報シンボルカウンタの計数値cntが情報シンボル数nInfo（本動作例の場合は20）に一致するまで、ステップS32からステップS39までの処理を繰り返し、サブルーチン処理P1を終了して図７のメインルーチンに戻る。

そして、図７のステップS22において、キューメモリ72から溢れた32ビットのデータをダミーデータとして読み出す。これは表４のステップS1102に対応する。次いで、ステップS23において、キューメモリ72の内容を、キューメモリ72のフィード回数nFeed回だけ進めるが、上記の処理と同様に、本動作例の場合、当該サブルーチン処理P2において何も実行しない。次いで、ステップS24において、キューメモリ72からパリティシンボルを４シンボルずつ合計nParitysシンボル読み出し、符号語バッファBufB［］のパリティ語領域に格納する。この処理をサブルーチン処理P4とする。

次に、サブルーチン処理P4の内容を図11を参照して具体的に説明する。まず、ステップS61において、キュー読み出しカウンタの計数値cntRQを０に初期化する。次いで、ステップS62において、キューメモリ72からパリティシンボルを４シンボルずつ読み出し、ステップS63において、符号語バッファのパリティ語領域に、ステップS62で取り出した４シンボル（32ビット）分のパリティシンボルを格納する。この場合、情報シンボル数nInfoは20であり、キュー読み出しカウンタの計数値cntRQは０であるから、符号語バッファに格納する位置BufB［nInfo＋cntRQ・4,…,nInfo＋cntRQ・４＋３］はBufB［20,…,23］となる。本実施形態の場合、４シンボル（＝32ビット）分のデータ（a,b,c,d）を符号語バッファBufB［i,…,i＋３］に格納する場合、BufB［ｉ］にはデータａが、BufB［ｉ＋１］にはデータｂが、BufB［ｉ＋２］にはデータｃが、BufB［ｉ＋３］にはデータｄが格納されるものとする。

さらに、ステップS64において、キュー読み出しカウンタの計数値cntRQを１だけインクリメントする。この場合、計数値cntRQは１になる。次いで、ステップS65において、キュー読み出しカウンタの計数値cntRQがキュー書き込み回数nQより小さい場合、ステップS62に戻る一方、そうでない場合、当該サブルーチン処理P4を終了してリターンする。この場合、計数値cntRQは１であり、キュー書き込み回数nQは３であるから、ステップS62に戻る。次に、ステップS62において、キューメモリ72からパリティシンボルを４シンボルずつ読み出し、ステップS63において、符号語バッファのパリティ語領域に、ステップS62で読み出した４シンボル（＝32ビット）分のパリティシンボルを格納する。この場合、情報シンボル数nInfoは20であり、キュー読み出しカウンタの計数値cntRQは１であるから、符号語バッファに格納する位置BufB［nInfo＋cntRQ・4,…,nInfo＋cntRQ・４＋３］はBufB［24,…,27］となる。そして、ステップS64において、キュー読み出しカウンタの計数値cntRQを１だけインクリメントする。この場合、計数値cntRQは２になる。さらに、ステップS65において、キュー読み出しカウンタの計数値cntRQがキュー書き込み数nQより小さい場合、ステップS62に戻る一方、そうでない場合、当該サブルーチン処理P4を終了してリターンする。この場合、計数値cntRQは２であり、書き込み回数nQは３であるから、ステップS62に戻る。以降、ステップS62、S63、S64の処理を同様に実行し、その後ステップS65に進む。このとき、計数値cntRQは３であり、書き込み回数nQは３であるから、当該サブルーチン処理P4を終了して元のメインルーチンに戻る。上記サブルーチン処理P4は、表４のステップS1103、S1104、及びS1105の処理に対応する。

以上で、本動作例であるRS（32,20,d＝13）の符号語生成処理を終了する。ただし、符号語は符号語バッファBufB［0,…,nInfo＋nParity－１］に格納されている。この場合、情報シンボル数nInfoは20、パリティシンボル数nParityは12であるから、符号語は符号語バッファBufB［0,…,31］に格納されている。

次に、本実施形態の誤り訂正符号化装置を用いて、上記第１の動作例と異なる最小符号間距離ｄを有し、８ビットを１シンボルとしたリード・ソロモン符号RS（15,10,d＝６）を符号化する処理について、図５乃至図11及び表５を用いて説明する。これを第２の動作例とする。

ただし、原始多項式ｍ（Ｘ）と原始元αは、第１の実施形態と同様にそれぞれ式（12）、式（13）で表される。ただし、生成多項式Ｇ（Ｘ）は次式で表わすことができる。

$$G(X)=\prod_{i=0}^{8}(X-\alpha^{i})=k_1\cdot X^8+k_2\cdot X^7+\cdots+k_8\cdot X+k_9 \quad \cdots(34)$$

従って、第２の動作例では、係数k_n（１≦ｎ≦５）の値が第１の実施形態や本実施形態の第１の動作例とは異なる。

表５は、図５乃至図11のフローチャートに従って、誤り訂正符号化装置が動作する場合において、本動作例での図４の各32ビットラッチの状態である。表５の２重下線はデータの書き込みを表し、下線はデータの読み込みを表す。また、データp_{000}から始まるパリティシンボルの記号やデータd_{000}から始まる記号は、第１の実施形態や本実施形態の第１の動作例における記号と異なり、図14に示すような計算方法で求められる値である。本動作例の場合、情報シンボル数nInfoは10であり、パリティシンボル数nParityは５である。

表5

ステップ	cnt	cntWQ	ラッチ73	ラッチ74	ラッチ75	ラッチ69
S2001	-	-	00000000	00000000	00000000	00000000
S2002	0	-	00000000	00000000	00000000	00000000
S2003	0	-	00000000	00000000	00000000	00000000
S2004	0	-	00000000	00000000	00000000	00000000
S2005	0	2	(p000, …, p003)	00000000	00000000	00000000
S2006	0	1	(p004, 00, 00, 00)	(p000, …, p003)	00000000	00000000
S2007	1	0	(p004, 00, 00, 00)	(p000, …, p003)	00000000	00000000
S2008	1	0	(p004, 00, 00, 00)	(p000, …, p003)	00000000	00000000
S2009	1	0	00000000	(p004, 00, 00, 00)	(p000, …, p003)	00000000
S2010	1	2	(p005, …, p008)	00000000	(p004, 00, 00, 00)	(p000, …, p003)
S2011	1	1	(p009, 00, 00, 00)	(p005, …, p008)	00000000	(p004, 00, 00, 00)
S2012	2	0	(p009, 00, 00, 00)	(p005, …, p008)	00000000	(p004, 00, 00, 00)
S2013	2	0	(p009, 00, 00, 00)	(p005, …, p008)	00000000	00000000
S2014	2	0	00000000	(p009, 00, 00, 00)	(p005, …, p008)	00000000
S2015	2	2	(p010, …, p013)	00000000	(p009, 00, 00, 00)	(p005, …, p008)
S2016	2	1	(p014, 00, 00, 00)	(p010, …, p013)	00000000	(p009, 00, 00, 00)
S2017	3	0	(p014, 00, 00, 00)	(p010, …, p013)	00000000	(p009, 00, 00, 00)
S2018	3	0	(p014, 00, 00, 00)	(p010, …, p013)	00000000	00000000
S2019	3	0	00000000	(p014, 00, 00, 00)	(p101, …, p013)	00000000
…	…	…	…	…	…	…
S2052	10	0	(p049, 00, 00, 00)	(p045, …, p048)	00000000	(p044, 00, 00, 00)
S2053	10	0	xxxxxxxx	(p049, 00, 00, 00)	(p045, …, p048)	00000000
S2054	10	0	xxxxxxxx	xxxxxxxx	(p049, 00, 00, 00)	(p045, …, p048)
S2055	10	0	xxxxxxxx	xxxxxxxx	00000000	(p049, 00, 00, 00)

以下に、本実施形態の第２の動作例における初期化処理について図６を参照して説明する。

まず、図６のステップS11において、ガロア体演算テーブルtGalois［nQueueLatch］［256］の初期化を行う。本動作例の場合、キューメモリ72を構成する32ビットラッチの数nQueueLatchは３である。本動作例のように最小符号間距離が６の場合、ガロア体演算テーブルの内容は、表６及び表７に示す内容となるように初期化される。また、32ビットラッチ73,74,75,69を０にクリアする。これは表５のステップS2001に対応する。

表6

アドレスH	アドレスL	b[31, …, 24]	b[23, …, 16]	b[15, …, 8]	b[7, …, 0]
0	00H	0	0	0	0
0	01H	k_5	k_4	k_3	k_2
0	02H	$k_5 \cdot \alpha$	$k_4 \cdot \alpha$	$k_3 \cdot \alpha$	$k_2 \cdot \alpha$
0	03H	$k_5 \cdot \alpha^{25}$	$k_4 \cdot \alpha^{25}$	$k_3 \cdot \alpha^{25}$	$k_2 \cdot \alpha^{25}$
0	04H	$k_5 \cdot \alpha^{2}$	$k_4 \cdot \alpha^{2}$	$k_3 \cdot \alpha^{2}$	$k_2 \cdot \alpha^{2}$
0	05H	$k_5 \cdot \alpha^{50}$	$k_4 \cdot \alpha^{50}$	$k_3 \cdot \alpha^{50}$	$k_2 \cdot \alpha^{50}$
…	…	…	…	…	…
0	FBH	$k_5 \cdot \alpha^{234}$	$k_4 \cdot \alpha^{234}$	$k_3 \cdot \alpha^{234}$	$k_2 \cdot \alpha^{234}$
0	FCH	$k_5 \cdot \alpha^{168}$	$k_4 \cdot \alpha^{168}$	$k_3 \cdot \alpha^{168}$	$k_2 \cdot \alpha^{168}$
0	FDH	$k_5 \cdot \alpha^{80}$	$k_4 \cdot \alpha^{80}$	$k_3 \cdot \alpha^{80}$	$k_2 \cdot \alpha^{80}$
0	FEH	$k_5 \cdot \alpha^{88}$	$k_4 \cdot \alpha^{88}$	$k_3 \cdot \alpha^{88}$	$k_2 \cdot \alpha^{88}$
0	FFH	$k_5 \cdot \alpha^{175}$	$k_4 \cdot \alpha^{175}$	$k_3 \cdot \alpha^{175}$	$k_2 \cdot \alpha^{175}$
1	00H	0	0	0	0
1	01H	k_1	0	0	0
1	02H	$k_1 \cdot \alpha$	0	0	0
1	03H	$k_1 \cdot \alpha^{25}$	0	0	0
1	04H	$k_1 \cdot \alpha^{2}$	0	0	0
1	05H	$k_1 \cdot \alpha^{50}$	0	0	0
…	…	…	…	…	…
1	FBH	$k_1 \cdot \alpha^{234}$	0	0	0
1	FCH	$k_1 \cdot \alpha^{168}$	0	0	0
1	FDH	$k_1 \cdot \alpha^{80}$	0	0	0
1	FEH	$k_1 \cdot \alpha^{88}$	0	0	0
1	FFH	$k_1 \cdot \alpha^{175}$	0	0	0

表７

アドレスH	アドレスL	b[31,…,24]	b[23,…,16]	b[15,…,8]	b[7,…,0]
2	00H	0	0	0	0
2	01H	0	0	0	0
2	02H	0	0	0	0
2	03H	0	0	0	0
2	04H	0	0	0	0
2	05H	0	0	0	0
…	…	…	…	…	…
2	FBH	0	0	0	0
2	FCH	0	0	0	0
2	FDH	0	0	0	0
2	FEH	0	0	0	0
2	FFH	0	0	0	0

次に、ステップS12において、式（31）で表わされるキューメモリ72のフィード回数nFeedを求める。本動作例の場合、キューメモリ72を構成する32ビットラッチの数nQueueLatchは３であり、付加するパリティシンボル数nParityは５であるから、キューメモリ72のフィード回数nFeedは１になる。次いで、テップS13において、式（32）で表わされるキューメモリ72の書き込み数nQを求める。本動作例の場合、付加するパリティシンボル数nParityは５であるから、キューメモリ72の書き込み数nQは２になる。さらに、ステップS14において、式（33）で表わすことができるキュー溢れシンボル取り出し位置indexを求める。本動作例の場合、付加するパリティシンボル数nParityは５であるから、キュー溢れシンボル取り出し位置indexは１になる。以上で初期化処理を終了する。

次に、図７乃至図11を用いて実際の符号語生成処理を説明する。まず、図７のステップS21において、作業用RAM63内の符号語バッファBufB［］から情報語を読み出し、パリティ語をキューメモリ72上に作成するサブルーチン処理P1を実行する。ここで、以下、当該サブルーチン処理P1の内容を図８を参照して具体的に説明する。

まず、図８のステップS31において、情報シンボルカウンタの計数値cntを０に初期化する。次いで、ステップS32において、符号語バッファから情報シンボルカウンタの計数値cnt番目の情報シンボルBufB［cnt］を読み出す。この場合、情報シンボルカウンタの計数値cntは０であるから、符号語バッファの先頭のシンボルである。以後、ここで読み出した先頭の情報シンボルをデータi_{000}とする。さらに、ステップS33において、32ビットラッチ73から32ビットのデータを読み出し、その上位からindex（本動作例の場合は１）バイト目、つまり上位８ビットを得る。この場合、32ビットラッチ73の内容は、クリアされて00000000Hであるので、その上位８ビットのデータはデータ00Hである。これは表５のステップS2002に対応する。次いで、ステップS34において、32ビットラッチ69の内容を０にクリアする。これは表５のステップS2003に対応する。さらに、ステップS35において、キューメモリ72の内容を、キューメモリ72のフィード回数nFeed回だけ進めるサブルーチン処理P2を実行する。

次に、このサブルーチン処理P2を図９を参照して具体的に説明する。まず、ステップS41において、キューメモリ72のフィードカウンタの計数値cntFeedをキューメモリ72のフィード回数nFeedに初期化する。本動作例の場合、フィード回数nFeedは１であるので、計数値cntFeedは１になる。次いで、ステップS42において、キューメモリ72のフィードカウンタの計数値cntFeedが０より大きければステップS43に進む一方、そうでないからサブルーチン処理P2を終了してリターンする。この場合、計数値cntFeedは１であるから、ステップS43に進む。ステップS43において、キューメモリ72の内容を進める。ただし、この時32ビットラッチ69の内容はそのままに保たれる。これは表５のステップS2004に対応する。
次いで、ステップS44でキューメモリ72のフィードカウンタの計数値cntFeedを１だけデクリメントする。この場合、計数値cntFeedは０になる。そして、ステップS42において、キューメモリ72のフィードカウンタの計数値cntFeedが０より大きければステップS43に進む一方、そうでないならサブルーチン処理P2を終了してリターンする。この場合、計数値cntFeedは０であるから、サブルーチン処理P2を終了して、図８のサブルーチン処理P1に戻る。
図８のステップS36において、入力データとステップS33で求めた32ビットラッチ73の上位８ビットのデータとの排他的論理和を求める。このデータ値をd_{000}とすると、この場合、入力データi_{000}とデータ00Hとの排他的論理和なので、データd_{000}は次式で表わすことができる。

$$d_{000}=i_{000} \quad \cdots(35)$$

次いで、ステップS37において、ステップS36で求めたデータ値と生成多項式の各係数との積をキューメモリ72に書き込むサブルーチン処理P3を実行する。以下、このサブルーチン処理P3を図10を参照して具体的に説明する。
まず、図10のステップS51において、キュー書き込みカウンタの計数値cntWQを０に初期化し、ステップS52において、ステップS56で求めたデータd_{000}と生成多項式（式（34））の係数とのガロア体上の積を、４シンボル（32ビット）だけ求める。その際に、ガロア体演算テーブルを用いるが、キュー書き込みカウンタの計数値cntWQが０の時には、生成多項式（式（34））の係数としてk_5,k_4,k_3,k_2が選ばれる。さらに、ステップS53において、ステップS52で求めた結果のデータをキューメモリ72に書き込む。このとき、32ビットラッチ75はデータ00000000Hを出力しており、８ビットラッチ68はデータ00Hを出力しているので、排他的論理和演算器64,65,66,67の入力データは、データ（00H,00H,00H,00H）と、データ（k_5・d_{000},k_4・d_{000},k_3・d_{000},k_2・d_{000}）である。従って、排他的論理和演算器64,65,66,67の出力データをデータ（$p_{000},p_{001},p_{002},p_{003}$）とすると、このデータは次式で表わすことができる。

$$(p_{000},p_{001},p_{002},p_{003})=(k_5\cdot d_{000},k_4\cdot d_{000},k_3\cdot d_{000},k_2\cdot d_{000}) \quad \cdots(36)$$

これは表５のステップS2005に対応する。
次に、ステップS54において、キュー書き込みカウンタの計数値cntWQを１だけインクリメントする。この場合、キュー書き込みカウンタの計数値cntWQは１になる。次いで、ステップS55において、キュー書き込みカウンタの計数値cntWQがキュー書き込み数nQより小さい場合はステップS52に戻り、そうでない場合はサブルーチン処理P3を終了してリターンする。この場合、キュー書き込みカウンタの計数値cntWQは１であり、キュー書き込み数nQは２であるから、ステップS52に戻る。
ステップS52において、ステップS36で求めたデータd_{000}と生成多項式（式（34））の係数とのガロア体上の積を、４シンボル（32ビット）だけ求める。その際に、ガロア体演算テーブルを用いるが、キュー書き込みカウンタの計数値cntWQが１の時には、生成多項式（式（14））の係数としてk_1,0,0,0が選ばれる。次いで、ステップS53において、S52で求めた結果のデータをキューメモリ72に書き込む。このとき、32ビットラッチ75はデータ00000000Hを出力しており、８ビットラッチ68はデータ00Hを出力しているので、排他的論理和演算器64,65,66,67の入力データは、データ（00H,00H,00H,00H）と、データ（k_1・d_{000},00H,00H,00H）である。従って、排他的論理和演算器64,65,66,67からの出力データを（p_{00}

$_4$,00H,00H,00H）とすると、データp_{004}は次式で表わすことができる。

$$p_{004}=k_1\cdot d_{000} \quad \cdots(37)$$

これは表５のステップS2006に対応する。

次に、ステップS54において、キュー書き込みカウンタの計数値cntWQを１だけインクリメントする。つまり、キュー書き込みカウンタの計数値cntWQは２になる。次いで、ステップS55において、キュー書き込みカウンタの計数値cntWQがキュー書き込み数nQより小さい場合はステップS52に戻り、そうでない場合はサブルーチン処理P3を終了してリターンする。この場合、キュー書き込みカウンタの計数値cntWQは２であり、キュー書き込み数nQは２であるから、サブルーチン処理P3を終了して、図８のサブルーチン処理P1にリターンする。

図８のステップS38において、情報シンボルカウンタの計数値cntを１だけインクリメントする。この場合、情報シンボルカウンタの計数値cntは１になる。次いで、ステップS39において、情報シンボルカウンタの計数値cntが情報シンボル数nInfoより小さい場合はステップS32に戻る一方、そうでない場合はサブルーチン処理P1を終了する。この場合、情報シンボルカウンタの計数値cntは１であり、情報シンボル数nInfoは10であるから、ステップS32に戻る。

ステップS32において、符号語バッファから情報シンボルカウンタの計数値cnt番目の情報シンボルつまりBufB［cnt］を読み出す。この場合、情報シンボルカウンタの計数値cntは１であるから、符号語バッファの先頭から２番目のシンボルである。ここで取り出した２番目の情報シンボルをデータi_{001}とする。次いで、ステップS33において、32ビットラッチ73から32ビットのデータを読み出し、その上位からindex（本動作例の場合は１）バイト目、つまり上位８ビットを得る。この場合、32ビットラッチ73の内容は、データ（p_{004},00H,00H,00H）であるので、その上位８ビットのデータはp_{004}である。これは表５のステップS2007に対応する。次いで、ステップS34において、32ビットラッチ69の内容を０にクリアする。これは表５のステップS2008に対応する。

さらに、ステップS35において、キューメモリ72の内容を、キューメモリ72のフィード回数nFeed回だけ進めるサブルーチン処理P2を実行する。当該サブルーチン処理P2では、上記の処理と同様にしてキューメモリ72を１回だけ進める。これは表５のステップS2009に対応する。そして、図８のステップS36において、入力データとステップS33で求めた32ビットラッチ73の上位８ビットのデータとの排他的論理和を求める。このデータ値をd_{001}とすると、この場合、入力データi_{001}とデータp_{004}との排他的論理和なので、データd_{001}は次式で表わすことができる。

$$d_{001}=i_{001}+p_{004} \quad \cdots(38)$$

次に、ステップS37において、ステップS36で求めたデータ値と生成多項式の各係数との積をキューメモリ72に書き込むサブルーチン処理P3を実行する。以下、そのサブルーチン処理P3を図10を参照して具体的に説明する。

まず、図10のステップS51において、キュー書き込みカウンタの計数値cntWQを０に初期化し、ステップS52において、ステップS36で求めたデータd_{001}と生成多項式（式（34））の係数とのガロア体上の積を、４シンボル（32ビット）だけ求める。その際に、ガロア体演算テーブルを用いるが、キュー書き込みカウンタの計数値cntWQが０の時には、生成多項式（式（34））の係数としてk_5,k_4,k_3,k_2が選ばれる。さらに、ステップS53において、ステップS52で求めた結果データをキューメモリ72に書き込む。このとき、32ビットラッチ75は、データ（p_{000},p_{001},p_{002},p_{003}）を出力しており、８ビットラッチ68はデータ00Hを出力しているので、排他的論理和演算器64,65,66,67の入力データは、データ（00H,p_{000},p_{001},p_{002}）と、データ（$k_5\cdot d_{001}$,$k_4\cdot d_{001}$,$k_3\cdot d_{001}$,$k_2\cdot d_{001}$）である。従って、排他的論理和演算器64,65,66,67の出力データを（p_{005},p_{006},p_{007},p_{008}）とすると、この出力データは、次式で表わすことができる。

$$(p_{005}, p_{006}, p_{007}, p_{008})$$
$$= (k_5 \cdot d_{001}, k_4 \cdot d_{001} + p_{000},$$
$$k_3 \cdot d_{001} + p_{001}, k_2 \cdot d_{001} + p_{002}) \quad \cdots (39)$$

これは表５のステップS2010に対応する。
次に、ステップS54において、キュー書き込みカウンタの計数値cntWQを１だけインクリメントする。つまり、キュー書き込みカウンタの計数値cntWQは１になる。次いで、ステップS55において、キュー書き込みカウンタの計数値cntWQがキュー書き込み数nQより小さい場合はステップS52に戻る一方、そうでない場合はサブルーチン処理P3を終了する。この場合、キュー書き込みカウンタの計数値cntWQは１であり、かつキュー書き込み数nQは２であるから、ステップS52に戻る。
ステップS52において、ステップS36で求めたデータd_{001}と生成多項式（式（34））の係数とのガロア体上の積を、４シンボル（32ビット）だけ求める。その際に、ガロア体演算テーブルを用いるが、キュー書き込みカウンタの計数値cntWQが１の時には、生成多項式（式（14））の係数としてk_1,0,0,0が選ばれる。次いで、ステップS53において、S52で求めた結果データをキューメモリ72に書き込む。このとき、32ビットラッチ75は、データ（p_{004},00H,00H,00H）を出力しており、８ビットラッチ68はp_{003}を出力しているので、排他的論理和演算器64,65,66,67の入力データは、データ（p_{003},p_{004},00H,00H）と、データ（$k_1 \cdot d_{001}$,00H,00H,00H）である。従って、排他的論理和演算器64,65,66,67の出力データをデータ（p_{009},p_{004},00H,00H）とすると、データp_{009}は次式で表わすことができる。

$$p_{009} = k_1 \cdot d_{001} + p_{003} \quad \cdots (40)$$

これは表５のステップS2011に対応する。
次に、図10のステップS54において、キュー書き込みカウンタの計数値cntWQを１だけインクリメントする。つまり、キュー書き込みカウンタの計数値cntWQは２になる。次いで、ステップS55において、キュー書き込みカウンタの計数値cntWQがキュー書き込みnQより小さい場合はステップS52に戻る一方、そうでない場合はサブルーチン処理P3を終了してリターンする。この場合、キュー書き込みカウンタの計数値cntWQは２であり、キュー書き込み数nQは２であるから、サブルーチン処理P3を終了して、図８のサブルーチン処理P1に戻る。
図８のステップS38において、情報シンボルカウンタの計数値cntを１だけインクリメントする。つまり、情報シンボルカウンタの計数値cntは２になる。次いで、ステップS39において、情報シンボルカウンタの計数値cntが情報シンボル数nInfoより小さい場合はステップS32に戻る一方、そうでない場合はサブルーチン処理P1を終了してリターンする。この場合、情報シンボルカウンタの計数値cntは２であり、情報シンボル数nInfoは10であるから、ステップS32に戻る。
ステップS32において、作業用RAM63内の符号語バッファから情報シンボルカウンタの計数値cnt番目の情報シンボルBufB［cnt］を読み出す。この場合、情報シンボルカウンタの計数値cntは２であるから、符号語バッファの先頭から３番目のシンボルである。ここで取り出した３番目の情報シンボルをデータi_{002}とする。次いで、ステップS33において、32ビットラッチ73から32ビットのデータを取り出し、その上位からindex（本動作例の場合は１）バイト目、つまり上位８ビットのデータを得る。この場合、32ビットラッチ73の内容は（p_{009},p_{004},00H,00H）であるので、その上位８ビットのデータはp_{009}である。これは表５のステップS2012に対応する。さらに、ステップS34において、32ビットラッチ69の内容を０にクリアする。これは表５のステップS2013に対応する。
次いで、ステップS35において、キューメモリ72の内容を、キューメモリ72のフィード回数nFeed回だけ進めるサブルーチン処理P2を実行する。当該サブルーチン処理P2では、上記の処理と同様にしてキューメモリ72を１回だけ進める。これは表５のステップS2014に対応する。

次いで、図８のステップS36において、入力データとステップS33で求めた32ビットラッチ73の上位８ビットのデータとの排他的論理和を求める。このデータ値をd_{002}とすると、この場合、入力データi_{002}とデータp_{009}との排他的論理和なので、データd_{002}は次式で表わすことができる。

$$d_{002}=i_{002}+p_{009} \quad \cdots(41)$$

以下、上記の処理と同様に、ステップS37、S38、S39と進み、さらに、情報シンボルカウンタの計数値cntが情報シンボル数nInfo（本動作例において、nInfo＝10である。）に一致するまでステップS32からステップS39までの処理を繰り返し、サブルーチン処理P1を終了して、図７のメインルーチンに戻る。

次いで、図７のステップS22において、キューメモリ72から溢れた32ビットのデータをダミーデータとして読み出す。これは表５のステップS2052に対応する。次いで、ステップS23において、キューメモリ72の内容を、キューメモリ72のフィード回数nFeed回だけ進めるサブルーチン処理P2を実行する。当該サブルーチン処理P2では、上記の処理と同様にしてキューメモリ72を１回だけ進める。これは表５のステップS2053に対応する。

次に、図７のステップS24において、キューメモリ72からパリティシンボルを４シンボルずつ合計nParityシンボル読み出し、作業用RAM63内の符号語バッファBufB［］のパリティ語領域に格納するサブルーチン処理P4を実行する。次に、当該サブルーチン処理P4の内容を図11を参照して具体的に説明する。

まず、図11のステップS61において、キュー読み出しカウンタの計数値cntRQを０に初期化し、ステップS62において、キューメモリ72からパリティシンボルを４シンボルずつ読み出す。さらに、ステップS63において、作業用RAM63内の符号語バッファのパリティ語領域に、ステップS62で読み出した４シンボル（32ビット）分のパリティシンボルを格納する。この場合、情報シンボル数nInfoは10であり、キュー読み出しカウンタの計数値cntRQは０であるから、符号語バッファに格納する位置BufB［nInfo＋cntRQ・4,…,nInfo＋cntRQ・４＋３］はBufB［10,…,13］となる。そして、ステップS64において、キュー読み出しカウンタの計数値cntRQを１だけインクリメントする。この場合、計数値cntRQは１になる。次いで、ステップS65において、キュー読み出しカウンタの計数値cntRQがキュー書き込み回数nQより小さい場合、ステップS62に戻る一方、そうでない場合、サブルーチン処理P4を終了してリターンする。この場合、計数値cntRQは１であり、キュー書き込み回数nQは２であるから、ステップS62に戻る。

ステップS62において、キューメモリ72からパリティシンボルを４シンボルずつ読み出し、次いで、ステップS63において、作業用RAM63内の符号語バッファのパリティ語領域に、ステップS62で読み出した４シンボル（32ビット）分のパリティシンボルを格納する。この場合、情報シンボル数nInfoは10であり、キュー読み出しカウンタの計数値cntRQは１であるから、符号語バッファに格納する位置BufB［nInfo＋cntRQ・4,…,nInfo＋cntRQ・４＋３］はBufB［14,…,17］となる。さらに、ステップS64において、キュー読み出しカウンタの計数値cntRQを１だけインクリメントする。この場合、計数値cntRQは２になる。次いで、ステップS65において、キュー読み出しカウンタの計数値cntRQがキュー書き込み回数nQより小さい場合、ステップS62に戻る一方、そうでない場合、サブルーチン処理P4を終了してリターンする。この場合、計数値cntRQは２であり、キュー書き込み回数nQは２であるから、サブルーチン処理P4を終了してメインルーチンにリターンする。このサブルーチン処理P4は、表５のステップS2054及びS2055に対応する。

以上で、本動作例であるRS（15,10,d＝６）の符号語生成処理を終了する。ただし、符号語は符号語バッファBufB［0,…,nInfo＋nParity－１］に格納されている。この場合、情報シンボル数nInfoは10であり、パリティシンボル数nParityは５であるから、符号語は符号語バッファBufB［0,…,14］に格納されている。

当該第２の本実施形態においては、第１の実施形態に示した効果に加えて以下のような効果が得られる。入力データと生成多項式の各係数とのガロア体上の積データを予め演算してROM71に格納され、初期化処理時に作業用RAM63に転送され、作業用RAM63は係数記憶手段を構成する。図３及び図４の回路装置では、係数記憶手段である作業用RAM63から同時

に４シンボルの積データを読み出すことができるように構成される。また、ｍ段の32ビットラッチ72－１乃至72－ｍからなるキューメモリ72を用いて、第１の記憶手段と選択手段を実現する一方、８ビットラッチ68を用いて、第２の記憶手段を実現する。さらに、８ビット排他的論理和演算器64,65,66,67を用いて、排他的論理和演算手段を実現し、また、ROM71に格納されたプログラムを実行するCPU61とラッチ制御装置62とを用いて、読み出し制御手段と第１と第２の選択手段を実現する。

従って、誤り訂正符号化装置の回路構成を従来技術に比較して簡単化することができるとともに、リード・ソロモン符号RS（n,n－P,d＝Ｐ＋１）を、１以上４×ｍ以下である任意のＰについて符号化できる。例えば、ｍ＝16とすると、最小符号間距離ｄを２から65まで自由に設定できる。ここで、ガロア体上の積データを含むガロア体演算テーブルは、初期化処理において、ROM71から作業用RAM63上に転送されて設定されるので、上記初期化処理において、転送する積データを変更しかつ初期化のパラメータ（本実施形態においては、32ビットラッチの数nQueueLatchと、パリティシンボル数nParityである。）を変更することにより、装置の回路構成を変えずに、原始多項式の変更や符号語の最小符号間距離ｄの変更が可能である。また、符号語バッファを作業用RAM63上に有しているので、ROM71に格納されるプログラムにおいて、情報シンボルの取り込み処理の内容とパリティシンボル格納処理の内容を変更することによって、種々の積符号などの誤り訂正符号を符号化することができる。また、このようなプログラムの変更を行うと、LDCと積符号の切り換えも、装置の回路構成を変えずに行えるという特有の効果を有する。

＜第３の実施形態＞

図15は、本発明に係る第３の実施形態の誤り訂正復号化装置の構成を示すブロック図である。図15に示すように、この第３の実施形態の誤り訂正復号化装置は、

（ａ）図３に示す誤り訂正符号化装置と同様の構成を有する剰余演算器208と、

（ｂ）誤り数値及び誤り位置演算器209と、

（ｃ）受信語記憶装置読み出しコントローラ210と、

（ｄ）排他的論理和演算器211と、

（ｅ）受信語記憶装置書き込みコントローラ212と、

（ｆ）複数Ｎ個のラッチが縦続接続されたシフトレジスタからなる受信語記憶装置213とを備える。

以下、図15を参照してリード・ソロモン符号RS（N,N－2t,d＝2t＋１）の誤り訂正処理について説明する。

まず、シンドロームを演算する処理を述べる。リード・ソロモン符号RS（N,N－2t,d＝2t＋１）の生成多項式は次式で表わすことができる。

$$G(X)=(X-\alpha^{0})(X-\alpha^{1})\cdots(X-\alpha^{2t-1}) \quad \cdots(42)$$

ここで、受信すべき符号語Ａを符号シンボルa_iの列として次式で表し、

$$A=(a_0,a_1,a_2,\cdots,a_{n-1}) \quad \cdots(43)$$

受信時の誤りパターンＥを誤りシンボルe_iの列として次式で表し、

$$E=(e_0,e_1,e_2,\cdots,e_{n-1}) \quad \cdots(44)$$

受信語Ｙを受信シンボルy_iの列として次式で表すとき、

$$Y=(y_0,y_1,y_2,\cdots,y_{n-1}) \quad \cdots(45)$$

符号多項式Ａ（Ｘ）を

$$A(X)=a_0+a_1\cdot X+a_2\cdot X^2+\cdots+a_{n-1}\cdot X^{n-1} \quad \cdots(46)$$

とし、誤り多項式Ｅ（Ｘ）を

$$E(X)=e_0+e_1\cdot X+e_2\cdot X^2+\cdots+e_{n-1}\cdot X^{n-1} \quad \cdots(47)$$

とし、受信多項式Ｙ（Ｘ）を

$$Y(X)=y_0+y_1\cdot X+y_2\cdot X^2+\cdots+y_{n-1}\cdot X^{n-1} \quad \cdots(48)$$

と定義する。受信語Ｙは受信すべき符号語Ａに対して誤りパターンＥが加わったものであるから、受信多項式Ｙ（Ｘ）は、

$$Y(X)=A(X)+E(X) \quad \cdots(49)$$

と表わすことができる。ここで、剰余多項式ｒ（Ｘ）を
ｒ（Ｘ）＝［Ｙ（Ｘ）］ＭＯＤ　Ｇ（Ｘ）　　…（50）
として定義する。ここで、Ａ　ＭＯＤ　ＢはＡをＢで除算したときの剰余を演算する演算子である。符号語Ａ（Ｘ）は生成多項式Ｇ（Ｘ）で割り切れるので、式（49）及び式（50）から次式を得る。
ｒ（Ｘ）＝［Ｅ（Ｘ）］ＭＯＤ　Ｇ（Ｘ）　　…（51）
また、式（51）を展開した次式
$r(X)=r_0+r_1\cdot X+r_2\cdot X^2+\cdots+r_{2t-1}\cdot X^{2t-1}$…（52）
を用いて、剰余多項式ｒ（Ｘ）の各係数$r_0,r_1,r_2,\cdots,r_{2t-1}$を並べたベクトル表現である剰余ベクトルを次式で表わす。
$r=(r_0,r_1,r_2,\cdots,r_{2t-1})$　　…（53）
剰余多項式ｒ（Ｘ）は、受信多項式Ｙ（Ｘ）を生成多項式Ｇ（Ｘ）で割った余りであるから、情報多項式Ｉ（Ｘ）を生成多項式Ｇ（Ｘ）で割った余りであるパリティ多項式Ｒ（Ｘ）を求める方法と同じ方法で求められることがわかる。つまり、誤り訂正符号化装置に受信語Ｙを入力すると、入力された受信語Ｙを生成多項式Ｇ（Ｘ）で割ったときの剰余としてパリティ語の代わりに剰余ベクトルｒが得られる。
以上説明したように、剰余演算器208として、第２の実施形態の誤り訂正符号化装置を用いることができる。剰余ベクトルｒを得ると、例えば、次式を用いて、

$$S(X)=\sum_{i=0}^{2t-1}\left[r_i\cdot\left(G(X)-G(\alpha^i)\right)/(X-\alpha^i)\right]$$

…（54）

一般化シンドロームＳ（Ｘ）を演算することができる（例えば、従来技術文献「荒木純道（Kiyomichi Araki）ほか，“リード・ソロモン符号の復号のための一般化シンドローム多項式",1991年電子情報通信学会春季全国大会,Ａ－278,pp.1－280,1991年」参照。）。従って、第２の実施形態の誤り訂正符号化装置である剰余演算器208に受信語Ｙを入力し、式（54）の演算を行うことによって、出力として一般化シンドロームＳ（Ｘ）が得られる。
次いで、誤り数値及び誤り位置を求める処理について述べる。ここで、誤りがＬ個の位置$(j_1,j_2,\cdots,j_L)$に生じたと仮定して説明する。ここで、Ｌ≦ｔであり、$0\leqq j_1<j_2<\cdots<j_L\leqq n-1$とする。
式（54）より、WelchとBerlekampの方法（例えば、米国特許第4,633,470号参照。）や公知のユークリッド互除法等を用いて誤り数値と誤り位置の組(e_{j_n},j_n),n＝1,2,…,Lを求めることができる。誤り数値及び誤り位置演算器209には、剰余演算器208から剰余ベクトルｒが入力され、誤り数値及び誤り位置演算器209は、上記の公知の方法によって、誤り数値と誤り位置の組(e_{j_n},j_n)を求め、その誤り数値と誤り位置の組(e_{j_n},j_n)をｎ＝１からｎ＝Ｌ（Ｌ≦ｔ）まで誤り個数だけ順次出力する。ここで、誤り数値のデータは排他的論理和演算器211の第２の入力端子に出力される一方、誤り位置のデータはアドレスとして受信語記憶装置読み出しコントローラ210及び受信語記憶装置書き込みコントローラ212に出力される。
さらに、誤り訂正処理について述べる。誤り数値及び誤り位置演算器209は誤り数値と誤り位置の組(e_{j_n},j_n)を出力するが、その誤り位置j_nに位置する受信シンボルy_{j_n}に誤り数値e_{j_n}を加えることにより、誤りを訂正する。そのために、以下の処理を実行する。まず、受信語記憶装置213のデータ読み出し動作を制御する受信語記憶装置読み出しコントローラ210にアドレスとして誤り位置j_nを入力して、誤り位置j_nに位置する受信シンボルy_{j_n}を受信語記憶装置213から読み出す。その読み出した受信シンボルy_{j_n}と、それに対応する誤り数値e_{j_n}とを、排他的論理和演算器211に入力することにより、受信シンボルy_{j_n}と

誤り数値e_{j_n}の和$y_{j_n}+e_{j_n}$をその出力データとして得る。この出力データ$y_{j_n}+e_{j_n}$が訂正後の符号シンボルa_{j_n}となるので、この訂正後の符号シンボルa_{j_n}を、元の誤り位置j_nをアドレスとし、受信語記憶装置213のデータ書き込み動作を制御する受信語記憶装置書き込みコントローラ212を用いて、受信語記憶装置213に書き込む。以上の処理を、誤り数値及び誤り位置演算器209が、ｎ＝１からｎ＝Ｌ（Ｌ≦ｔ）まで順次誤り数値と誤り位置の組（e_{j_n},j_n）を出力することによって繰り返すことにより、すべての受信語に対して誤り訂正を実行して誤り復号化処理を完了することができる。このとき、受信語記憶装置213において、誤り訂正された符号語Ａを得ることができる。そして、当該符号語Ａは、受信語記憶装置213のアドレスを順次インクリメントしながら、受信語記憶装置読み出しコントローラ210に入力することにより、受信語記憶装置読み出しコントローラ210により、読み出されて誤り訂正された出力データとして出力される。

以上説明したように、第３の実施形態によれば、第２の実施形態の誤り訂正符号化装置を用いて誤り訂正復号化装置を実現することができる。従って、第１及び第２の実施形態と同様に、従来技術の装置に比較して回路構成が簡単であって、回路構成を変更することなく、実用に供することができる復号化速度を有して、種々の最小符号間距離を有する誤り訂正符号を復号化することができる誤り訂正復号化装置を実現することができる。

＜変形例＞

以上の実施形態において、誤り訂正符号化装置を以下のように構成してもよい。すなわち、変形例の誤り訂正符号化装置は、2^Nの元の数を有するガロア体GF（2^N）上の元を有するリード・ソロモン符号を用いて、１シンボル当たり自然数Ｎビットの入力データに対する誤り訂正符号を符号化する誤り訂正符号化装置において、

各入力データと上記リード・ソロモン符号の生成多項式の各係数とのガロア体上の複数の積データをそれぞれ予め演算して、上記複数の積データを、各アドレスに対して複数ｂ個の積データを１組として予め記憶する積データ記憶手段と、

それぞれＮ×ｂビットの記憶容量を有する自然数ｍ個の記憶装置からなる第１の記憶手段と、

入力データに応答して、上記積データ記憶手段に記憶された複数の積データを、複数ｂ個の積データを１組として並列に読み出すように上記積データ記憶手段を制御する読み出し制御手段と、

それぞれＮ×ｂビットの第１と第２の入力端子を有し、上記読み出し制御手段によって上記積データ記憶手段から並列に読み出される複数ｂ個の積データが第１の入力端子に入力され、上記第１の入力端子に入力されるデータと、上記第２の入力端子に入力されるデータとの排他的論理和を演算して演算結果のデータを出力する排他的論理和演算手段と、

上記ｍ個の記憶装置に記憶されたデータを１つの記憶装置毎に選択的に順次読み出して出力し、上記選択的に読み出して出力されるＮ×ｂビットのデータのうち上位Ｎ×（ｂ－１）ビットのデータを上記排他的論理和演算手段の第２の入力端子の下位Ｎ×（ｂ－１）ビットに出力するとともに、上記排他的論理和演算手段から出力される演算結果のデータを、上記ｍ個の記憶装置のうちの１つに選択的に順次切り換えて書き込むように、上記第１の記憶手段を制御する第１の選択手段と、

Ｎビットの記憶容量を有し、上記第１の選択手段によって上記ｍ個の記憶装置のうちの１つから選択的に出力されるＮ×ｂビットのデータのうち下位Ｎビットのデータを一時的に記憶して上記排他的論理和演算手段の第２の入力端子の上位Ｎビットに出力する第２の記憶手段と、

上記入力データを上記積データ記憶手段に順次入力することにより、上記第１の記憶手段のｍ個の記憶装置においてパリティデータを生成し、上記入力データに続いて、上記ｍ個の記憶装置を１つの記憶装置毎に選択的に順次切り換えることにより、上記ｍ個の記憶装置において生成される各パリティデータを順次出力する第２の選択手段とを備える。なお、図１及び図４の実施形態においては、Ｎ＝８、ｂ＝４、及びｍ＝３である。

以上の実施形態において、積データを記憶するためにROM14を用いているが、本発明はこれに限らず、ROM14に代えて、論理回路素子の組み合わせによる論理回路により構成され

、入力データに対して積データを演算する演算手段を用いてもよい。

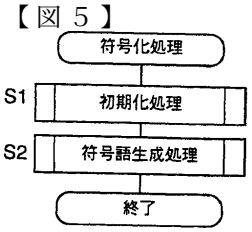
産業上の利用可能性

本発明によれば、2^Nの元の数を有するガロア体GF（2^N）上の元を有するリード・ソロモン符号を用いて、１シンボル当たり自然数Ｎビットの入力データに対する誤り訂正符号を符号化する誤り訂正符号化装置及び方法において、積データ記憶手段であるROM（14）は、各入力データと上記リード・ソロモン符号の生成多項式の各係数とのガロア体上の複数の積データをそれぞれ予め演算して、上記複数の積データを、各アドレスに対して複数ｂ個の積データを１組として予め記憶する。読み出し制御手段（12,13,24－26,28）は、入力データに応答して、ROM（14）に記憶された複数の積データを、上記複数ｂ個の積データを１組として並列に読み出した後、排他的論理和演算器（15－18）及びバス選択器（19）を介して自然数ｍ個の記憶装置（20－22）に選択的に順次書き込む。入力データをROM（14）に順次入力することにより、ｍ個の記憶装置においてパリティデータを生成して出力する。

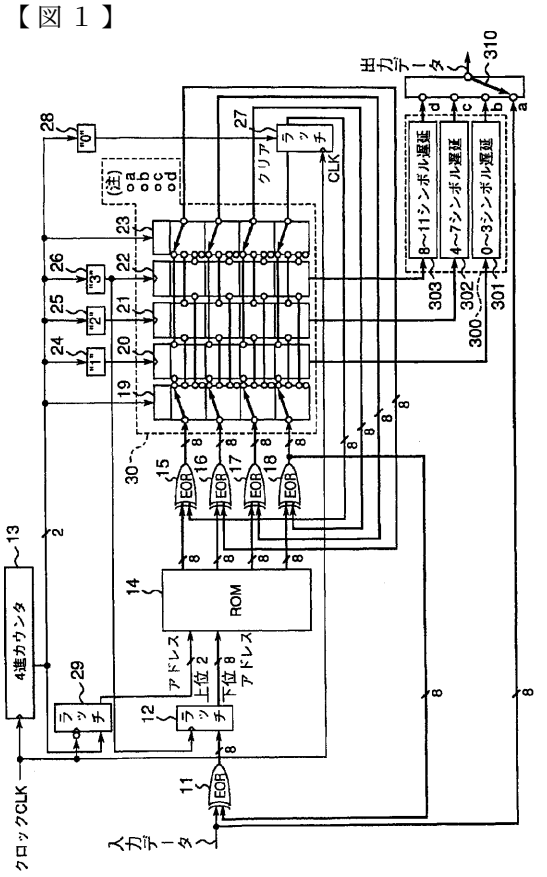
従って、複数の積データを記憶したROM14から４個の積データを１組として並列に（同時に）排他的論理和演算器（15－18）に対して読み出し可能に構成し、４個の積データを用いて同時に処理し、かつ３個の32ビットラッチ（20－22）を選択的に順次繰り返して使用することにより、符号化処理の演算をしているので、回路構成を、図14の従来技術の誤り訂正符号化装置の回路構成に比較して極めて簡単にすることができるとともに、効率的にかつ高速で演算することができ、実用に供することが可能な符号化速度で誤り訂正符号を符号化することができる。

また、本発明に係る上記誤り訂正符号化装置を用いて誤り訂正復号化装置を構成することにより、回路構成を極めて簡単にすることができるとともに、効率的にかつ高速で演算することができ、実用に供することが可能な復号化速度で誤り訂正符号を復号化することができる。

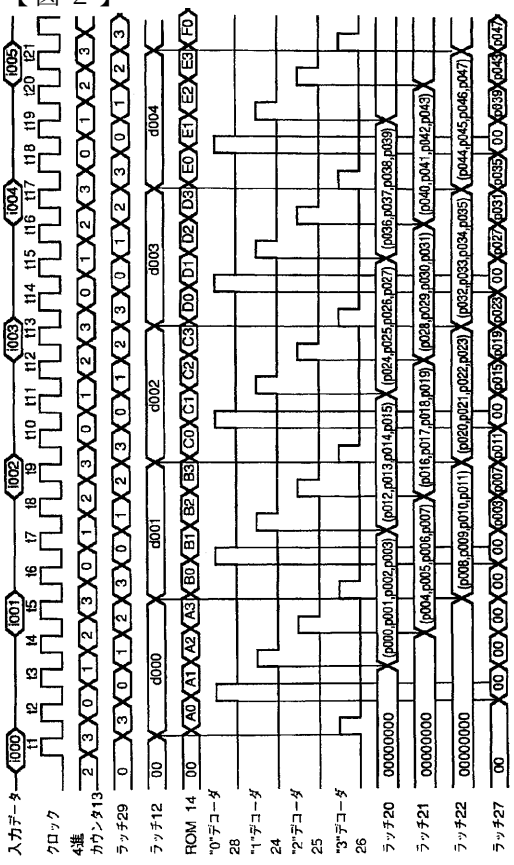
【図５】



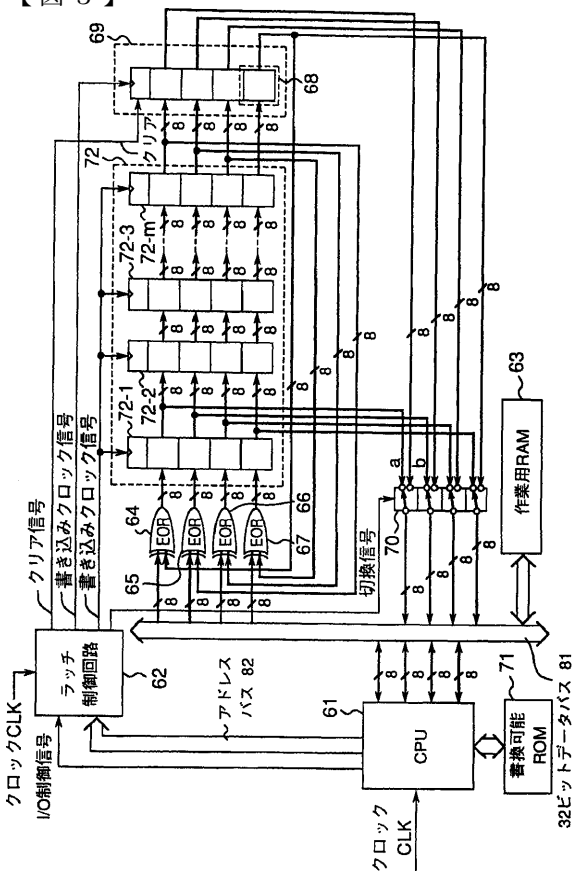
【図１】



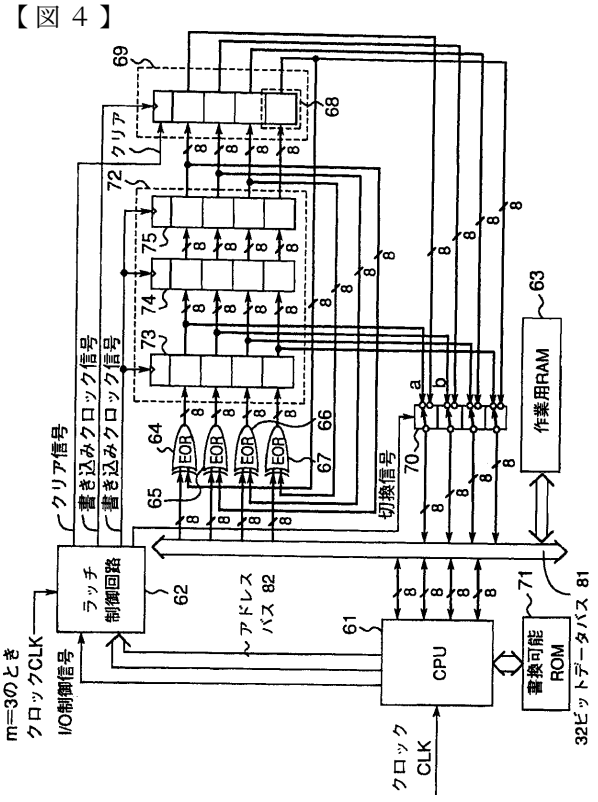
【図２】


入力データ
クロック
4進カウンタ13
ラッチ29
ラッチ12
ROM 14
"0"デコーダ 28
"1"デコーダ 24
"2"デコーダ 25
"3"デコーダ 26
ラッチ20
ラッチ21
ラッチ22
ラッチ27

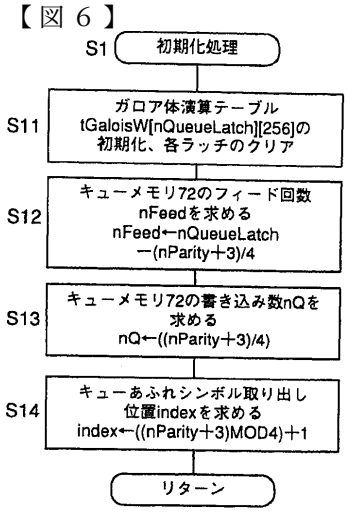
【図３】


クリア信号
書き込みクロック信号
書き込みクロック信号
ラッチ制御回路
62
アドレスバス 82
CPU
61
書換可能ROM
71
作業用RAM
63
切換信号
70
クロックCLK
I/O制御信号
クロックCLK
32ビットデータバス 81
クリア
72
72-1
72-2
72-3
72-m
68
69
64
65
66
67
EOR

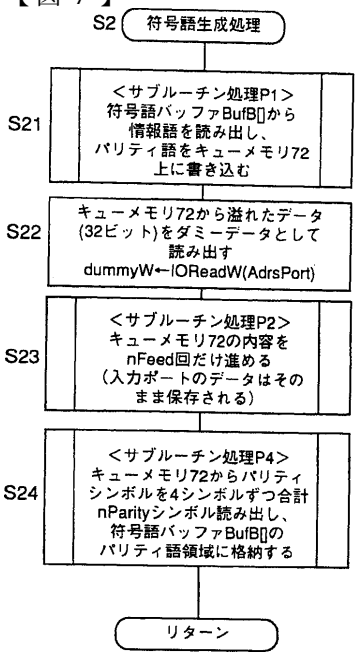
【図４】


m=3のとき
クロックCLK
I/O制御信号
クリア信号
書き込みクロック信号
書き込みクロック信号
ラッチ制御回路
62
アドレスバス 82
CPU
61
書換可能ROM
71
作業用RAM
63
切換信号
70
クロックCLK
32ビットデータバス 81
クリア
72
73
74
75
68
69
64
65
66
67
EOR

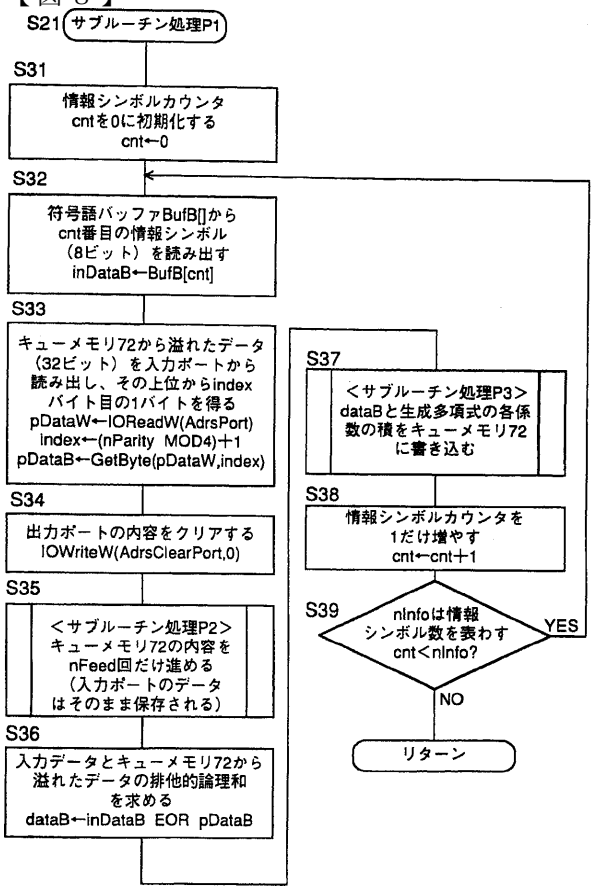
【図６】


S1
初期化処理
S11
ガロア体演算テーブル tGaloisW[nQueueLatch][256]の初期化、各ラッチのクリア
S12
キューメモリ72のフィード回数 nFeedを求める
nFeed←nQueueLatch −(nParity+3)/4
S13
キューメモリ72の書き込み数nQを求める
nQ←((nParity+3)/4)
S14
キューあふれシンボル取り出し位置indexを求める
index←((nParity+3)MOD4)+1
リターン

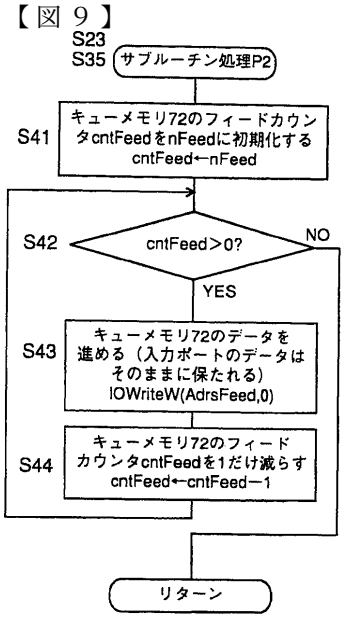
【図７】


S2
符号語生成処理
S21
<サブルーチン処理P1>
符号語バッファBufB[]から
情報語を読み出し、
パリティ語をキューメモリ72
上に書き込む
S22
キューメモリ72から溢れたデータ
(32ビット)をダミーデータとして
読み出す
dummyW←IOReadW(AdrsPort)
S23
<サブルーチン処理P2>
キューメモリ72の内容を
nFeed回だけ進める
(入力ポートのデータはそのまま保存される)
S24
<サブルーチン処理P4>
キューメモリ72からパリティ
シンボルを4シンボルずつ合計
nParityシンボル読み出し、
符号語バッファBufB[]の
パリティ語領域に格納する
リターン

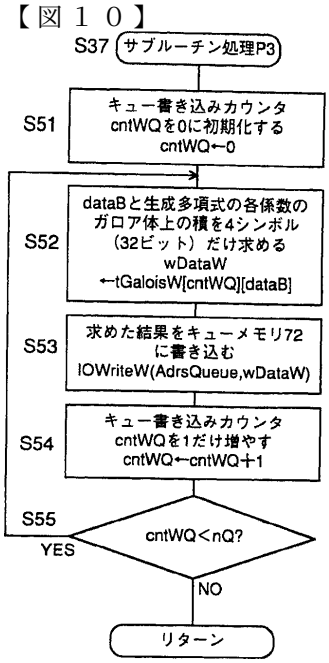
【図８】


S21
サブルーチン処理P1
S31
情報シンボルカウンタ
cntを0に初期化する
cnt←0
S32
符号語バッファBufB[]から
cnt番目の情報シンボル
(8ビット)を読み出す
inDataB←BufB[cnt]
S33
キューメモリ72から溢れたデータ
(32ビット)を入力ポートから
読み出し、その上位からindex
バイト目の1バイトを得る
pDataW←IOReadW(AdrsPort)
index←(nParity MOD4)+1
pDataB←GetByte(pDataW,index)
S34
出力ポートの内容をクリアする
IOWriteW(AdrsClearPort,0)
S35
<サブルーチン処理P2>
キューメモリ72の内容を
nFeed回だけ進める
(入力ポートのデータ
はそのまま保存される)
S36
入力データとキューメモリ72から
溢れたデータの排他的論理和
を求める
dataB←inDataB EOR pDataB
S37
<サブルーチン処理P3>
dataBと生成多項式の各係
数の積をキューメモリ72
に書き込む
S38
情報シンボルカウンタを
1だけ増やす
cnt←cnt+1
S39
nInfoは情報
シンボル数を表わす
cnt<nInfo?
YES
NO
リターン

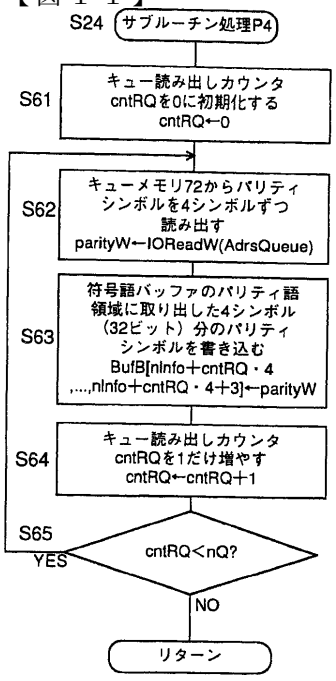
【図９】


S23
S35
サブルーチン処理P2
S41
キューメモリ72のフィードカウン
タcntFeedをnFeedに初期化する
cntFeed←nFeed
S42
cntFeed>0?
NO
YES
S43
キューメモリ72のデータを
進める(入力ポートのデータは
そのままに保たれる)
IOWriteW(AdrsFeed,0)
S44
キューメモリ72のフィード
カウンタcntFeedを1だけ減らす
cntFeed←cntFeed−1
リターン

【図１０】

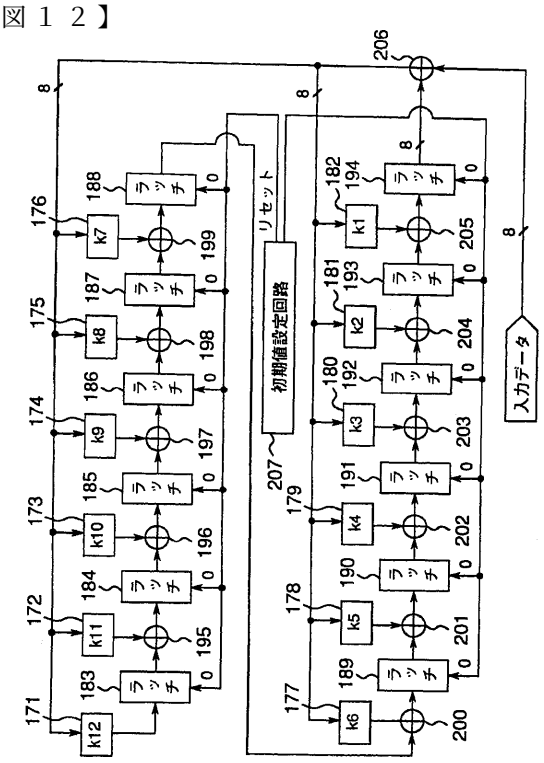

S37
サブルーチン処理P3
S51
キュー書き込みカウンタ
cntWQを0に初期化する
cntWQ←0
S52
dataBと生成多項式の各係数の
ガロア体上の積を4シンボル
(32ビット)だけ求める
wDataW
←tGaloisW[cntWQ][dataB]
S53
求めた結果をキューメモリ72
に書き込む
IOWriteW(AdrsQueue,wDataW)
S54
キュー書き込みカウンタ
cntWQを1だけ増やす
cntWQ←cntWQ+1
S55
cntWQ<nQ?
YES
NO
リターン

【図１１】



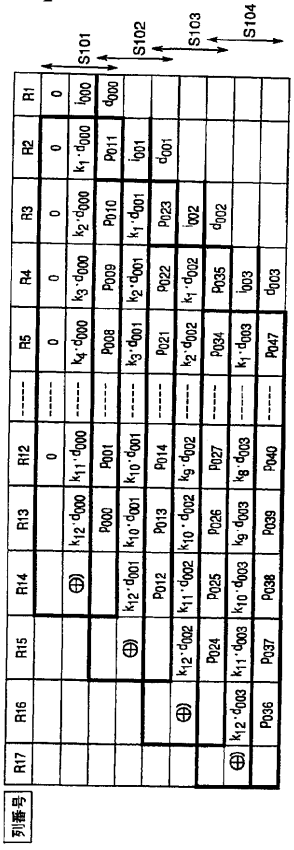
【図１２】

従来技術

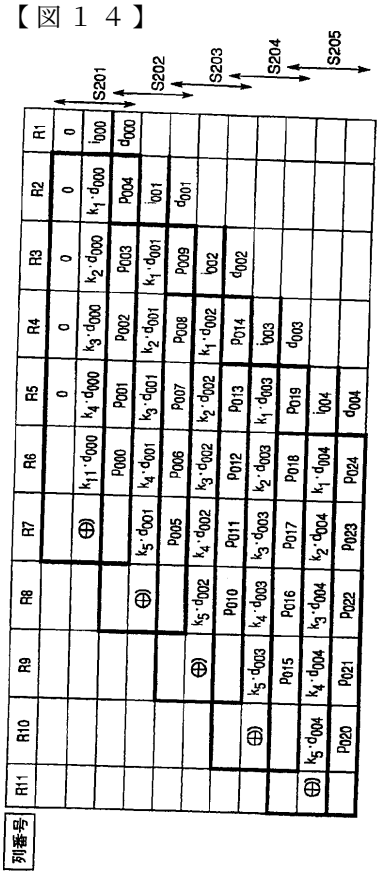


【図１３】

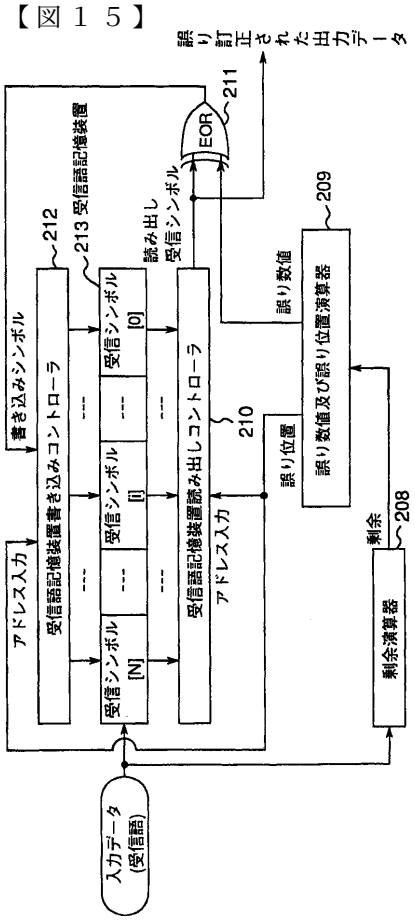
従来技術



【図１４】



【図１５】


誤り訂正された出力データ
211
EOR
読み出し受信シンボル
書き込みシンボル
212
213 受信語記憶装置
受信語記憶装置書き込みコントローラ
受信シンボル [0]
受信シンボル [i]
受信シンボル [N]
受信語記憶装置読み出しコントローラ
210
アドレス入力
誤り数値
誤り位置
209
誤り数値及び誤り位置演算器
剰余
208
剰余演算器
入力データ (受信語)

フロントページの続き

(56)参考文献 特開平０４－３６５１３９（ＪＰ，Ａ）
特開昭６１－２８９７３１（ＪＰ，Ａ）
特開平０１－２９３０１５（ＪＰ，Ａ）
特表平０４－５０５５４５（ＪＰ，Ａ）

(58)調査した分野(Int.Cl.[7]，ＤＢ名)
H03M 13/00 - 13/53