マイクロコンピュータによる2重および3重誤り訂正 BCH 符号の復号†

岡野 博一††

最近のディジタルシステムにおいて，信頼性を向上させるために誤り訂正符号が広く実用されるようになってきた．誤り訂正符号の実現にはたいていハード技術が用いられている．しかし，マイクロコンピュータ程度のもので高度な誤り訂正処理がソフト的に実現できれば，経済的で柔軟なデータ通信システムが構成され得る．

本論文において，BCH 符号を用いた2重および3重誤り訂正処理をソフト的に実現している．復号のアルゴリズムは Polkinghorn が特殊計算機を用いて行った方法に基づいており，通常用いられる Chien のアルゴリズムを用いるよりも効率が良い．さらに剰余表を用いたシンドローム算出法および効率的な誤りビット数判定法等の高速化手法を用いた結果，従来のハード技術とは異なった BCH 符号の高速処理をマイクロコンピュータ上で実現している．さらに，計算処理量等の検討を行い，本方式は高速データ通信回線に対しても実用可能であることを明らかにしている．

1. まえがき

最近，データ通信や計算機システムの信頼性を向上させるために符号理論が広く実用されるようになってきた．しかし，既存の多重誤り訂正符号においては能率の良いことと復号の容易さとは相反する問題であるため，現在の段階では特定の場合を除いてたかだか3重誤り訂正符号が実用化されているにすぎない．したがってより効率の良い符号，より復号の容易な符号の研究が続けられている[7),8)]．また，ROM を用いた復号法は効率の良い注目すべき方法である[9)]．

これらの方法は主としてハード技術によって実現されているが，ソフト技術によっても高度な誤り訂正処理を実現することができる[3),6)]．筆者はすでにマイクロコンピュータを用いた Fire 符号の効率の良い復号法を提案している[10),11)]．さらに，2重および3重誤り訂正 BCH 符号の復号をマイクロコンピュータの上でソフト的に実現したので報告する．

この方法は文献 3) の方法に基づいており，誤り位置多項式をテーブルを用いて解くので，通常用いられる Chien のアルゴリズムを用いるよりも効率が良い．しかし文献 3) には計算処理量 (time/space) などが明確に示されておらず，しかもガロア体の演算を効率良く行えるように特別に設計されたコンピュータを使用している．そこで，汎用コンピュータを用いた場合の復号方法および計算処理量などを明確にしておくことは意義のあることと考えられる．

さらに，マイクロコンピュータ程度のもので高度な誤り訂正処理が実現できれば工学的に有益である．つまり，半導体技術の進歩によって安価で高性能なマイクロコンピュータが種々の分野に進出している．そして，データ通信に用いられる端末もマイクロコンピュータを内蔵したかなり高度な処理能力を持つインテリジェント端末が用いられるようになってきている．したがって，データ通信における符号の処理をマイクロコンピュータのソフトによって実現できれば，経済的で柔軟なデータ通信システムが構成され得る．

また，制御関係，移動通信などにも符号理論を容易に応用することが可能となろう．

しかしながらマイクロコンピュータの処理能力はそれほど高速とはいえないので，出来得る限り高速処理をするように工夫する必要がある．ここで述べる方法はシンドロームの算出を剰余表を用い，誤りビット数の判定を効率化し，誤り位置多項式の解法はテーブルを用いて行うなどの高速化手法を用い，従来のハード技術とは異なった BCH 符号の高速処理をマイクロコンピュータを用いてソフト的に実現している．

本方式は 4,800 ボーおよび 48K ボーの高速データ通信回線に対して十分有効であり，ソフトによる BCH 符号のデータ通信への適用が十分期待できる．なお，使用した言語は，マイクロコンピュータ PFL-16A のアセンブリ言語である．

2. BCH 符号の復号法

2.1 BCH 符号

† Decoding of Double and Triple Error Correcting BCH Codes by Microcomputer by HIROKAZU OKANO (Department of Information Electronics, Tokuyama Technical College).

†† 徳山工業高等専門学校情報電子工学科

BCH 符号はランダム誤りを訂正する代表的な符号の一つである．詳細については文献 1)～6) を参照されたい．

ここではまず，3重以下のランダム誤りを訂正する符号（単に3重誤り訂正 BCH 符号と呼ぶ）について具体例によって説明する．2重以下のランダム誤りを訂正する符号（単に2重誤り訂正 BCH 符号と呼ぶ）についても原理は同様なので簡単な説明に留める．

本論文で用いた3重誤り訂正 BCH 符号の生成多項式は GF(2^4) の原始元を α とすると，α, α^3, α^5 を根とする多項式の積で表わされ次式となる．

$$G(x)=m_1(x)\cdot m_3(x)\cdot m_5(x)$$
$$=x^{10}+x^8+x^5+x^4+x^2+x+1 \qquad (1)$$

ただし，$\begin{cases} m_1(x)=x^4+x+1 \\ m_2(x)=x^4+x^3+x^2+x+1 \\ m_3(x)=x^2+x+1 \end{cases}$

この符号の符号長は 15，情報符号長は 5，チェックビット長は 10，最小距離は7である．

符号化については，一般の巡回符号と同様であり，シフト演算法あるいは剰余表を用いたテーブル・ルックアップ法によって行われる[10),11)]．前者はシフトレジスタの演算をそのまま模擬したものであり，計算速度が遅い．後者の方法によれば，計算速度を5～10倍程度に高速にすることができる．剰余表は入力8ビットとその入力に8ビットの all ϕ's を付加したものを多項式で割り算したときの剰余との対応表である．剰余表は256語のテーブルである．詳細は省略する．

なお，ガロア体の演算は，和はベクトル表現，乗除は指数表現を用いて行うのが便利なので相互の変換をテーブルを用いて行う．テーブルはメインプログラムのデータ定義域に格納しておき，各サブルーチンから参照することができるようにしている．

TBL 1 は，α^i（ベクトル）$\rightarrow i$（指数），

TBL 2 は，$i \rightarrow \alpha^i$

の変換を行うためのものである[1)~6)]．TBL 1，TBL 2 は省略する．

3重誤り訂正 BCH 符号をソフトによって復号する場合，次の手順で行う．

（1） 受信系列からシンドローム S_1, S_3, S_5 を算出

（2） 誤りビット数の判定

（3） 誤り位置多項式の解法

（4） 誤り訂正の実行

以下に，各ステップについて述べる．

2.2 シンドロームの算出[1)~6)]

シンドロームは受信系列を $m_1(x), m_3(x), m_5(x)$ で割ったときの剰余をそれぞれ $R_1(x), R_3(x), R_5(x)$ とすると次式で表わされる．

$$\left.\begin{aligned} s_1&=R_1(\alpha) \\ s_3&=R_3(\alpha^3) \\ s_5&=R_5(\alpha^5) \end{aligned}\right\} \qquad (2)$$

したがって，符号化と同様に剰余を算出すれば良い．そして，$R_3(x)\rightarrow R_3(\alpha^3)$, $R_5(x)\rightarrow R_5(\alpha^5)$ の変換を行う．この変換はガロア体の指数↔ベクトル変換表を使用してガロア体の演算をしても良いが，高速に行うために，剰余のパターンより合成する方法を用いている．この処理は，サブルーチン SBCH で行う．詳細は省略する．

2.3 誤り位置多項式の解法[3),6)]

順序は前後するが，誤りビット数が決定した後の誤り位置多項式の解法について述べる．

2.3.1 1ビット誤りの訂正

1ビット誤りの場合の誤り位置多項式は次式で与えられる．

$$x+s_1=0 \qquad (3)$$

この場合，根は s_1 である．サブルーチン CR 1 によって s_1 の指数を SAVE 1 に格納する．

2.3.2 2ビット誤りの訂正

2ビット誤りの場合の誤り位置多項式は次式で与えられる．

$$x^2+s_1x+\frac{s_1^3+s_3}{s_1}=0 \qquad (4)$$

この場合，$s_1\neq 0$, $s_1^3+s_3\neq 0$ としてよいので，$x=s_1Y$ とおいて次式を得る．

$$Y^2+Y+\frac{s_1^3+s_3}{s_1^3}=0 \qquad (5)$$

そして，(4)式の根を $x_1=\alpha^r$, $x_2=\alpha^s$ とすると，$\frac{s_1^3+s_3}{s_1^3}$ と $(s-r)$ は1対1に対応しているので，

表 1 $(s-r)$ の表
Table 1 Table of $(s-r)$.

$\frac{S_1^3+S^3}{S_1^3}$	$s-r$
α^0	1
α	2
α^4	3
α^2	4
α^8	5
α^{10}	6
α^1	7

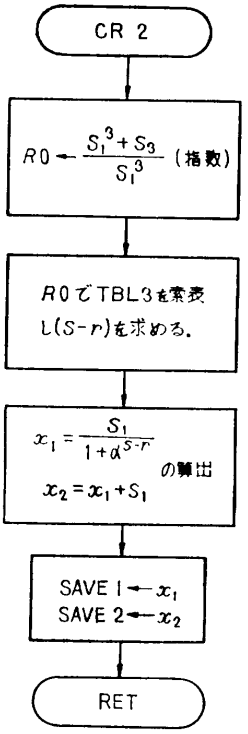


図 1 2次方程式の解法アルゴリズム
Fig. 1 Algorithm for solving a quadratic equation.

両者をあらかじめ算出して TBL 3 に格納しておく．この $(s-r)$ の表を**表 1** に示す．(4)式の根は次式となる．

$$\begin{cases} x_1=\dfrac{s_1}{1+\alpha^{(s-r)}} \\ x_2=x_1+s_1 \end{cases} \qquad (6)$$

この方法を用い，サブルーチン CR 2 によって(4)式の根，x_1, x_2 を求めその指数を SAVE 1, SAVE 2 に格納する．(4)式の2次方程式の解法アルゴリズムを**図 1** に示す．また，そのプログラムを**図 2** に示す．

なお，(5)式の定数は $1+s_3/s_1^3$ と変形されるので，s_3/s_1^3 の値に対して(5)式の2つの解を直接対応させたテーブルを用いれば若干効率が良くなる．これは文献 9) の方法であるが，本質的には同一と考えられる．

2.3.3 3ビット誤りの訂正

3ビット誤りの場合の誤り位置多項式は次式で与えられる．

$$x^3+s_1x^2+\frac{s_1^2s_3+s_5}{s_1^3+s_3}x+s_1^3+s_3+\frac{s_1(s_1^2s_3+s_5)}{s_1^3+s_3}=0 \qquad (7)$$

```
STNO. LABEL    OP.     OPERANDS/COMMENT
    1          BGN     CR2
    2          L       X1,(TBL1)
    3          L       R1,(SAVE1)
    4          A       X1,R1
    5          L       R1,0(X1)
    6          L       R2,(S13)
    7          L       X1,(TBL1)
    8          A       X1,R2
    9          L       R2,0(X1)
   10          S       R1,R2
   11          MV      R0,R1
   12          BAL     FUKA
   13          L       X1,(TBL3)
   14          A       X1,R0
   15          L       R2,0(X1)
   16          L       X1,(TBL2)
   17          A       X1,R2
   18          L       R0,0(X1)
   19          MVI     R2,X'01'
   20          EOR     R0,R2
   21          L       X1,(TBL1)
   22          A       X1,R0
   23          L       R0,0(X1)
   24          L       R1,(S1)
   25          L       X1,(TBL1)
   26          A       X1,R1
   27          L       R1,0(X1)
   28          S       R1,R0
   29          MV      R0,R1
   30          BAL     FUKA
   31          MV      R1,R0
   32          ST      R1,(SAVE1)
   33          L       X1,(TBL2)
   34          A       X1,R1
   35          L       R2,0(X1)
   36          L       R0,(S1)
   37          EOR     R2,R0
   38          L       X1,(TBL1)
   39          A       X1,R2
   40          L       R2,0(X1)
   41          ST      R2,(SAVE2)
   42          RET
   43  *
   44  *
   45  FUKA    SKIP    R0,M
   46          B       J1
   47          L       R3,N15
   48          A       R0,R3
   49  J1      RET
   50  N15     DC      F'15'
   51  S1      EXTRN   S1
   52  S13     EXTRN   S13
   53  TBL1    EXTRN   TBL1
   54  TBL2    EXTRN   TBL2
   55  TBL3    EXTRN   TBL3
   56  SAVE1   EXTRN   SAVE1
   57  SAVE2   EXTRN   SAVE2
   58          END
```

図 2 2次方程式の解法プログラム
Fig. 2 Program for solving a quadratic equation.

この場合，$s_1^3+s_3\neq 0$ としてよいから，$x=Y+s_1$ とおき次式を得る．

$$Y^3+\frac{s_1^5+s_5}{s_1^3+s_3}Y+s_1^3+s_3=0 \qquad (8)$$

したがって，$s_1^5+s_5=0$ のとき

$$Y=(s_1^3+s_3)^{1/3} \qquad (9)$$

さて，$s_1^5+s_5\neq 0$ のときは，さらに

$$Z=\left[\frac{s_1^5+s_5}{s_1^3+s_3}\right]^{-1/2}\cdot Y \qquad (10)$$

とおき，次式を得る．

表 2 Z_1, Z_2（3次方程式の根）の表

Table 2 Table of Z_1 and Z_2 (roots of cubic equations).

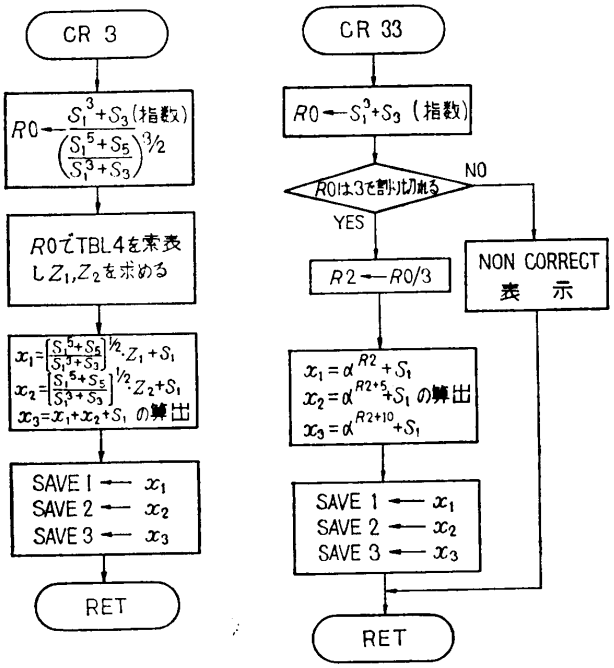
$\frac{S_1^3+S_3}{\left(\frac{S_1^5+S_5}{S_1^3+S_3}\right)^{3/2}}$	Z_1, Z_2
α^5	α^{10}, α^{11}
α^{10}	α^5, α^7

$$Z^3+Z+\frac{S_1^3+S_3}{\left[\frac{S_1^5+S_5}{S_1^3+S_3}\right]^{3/2}}=0 \qquad (11)$$

(11)式の解 Z_1, Z_2 は $\frac{S_1^3+S_3}{\left[\frac{S_1^5+S_5}{S_1^3+S_3}\right]^{3/2}}$ に対してあらかじめ算出し TBL 4 に格納しておく．この Z_1, Z_2（3次方程式の根）の表を表 2 に示す．すると，(7)式の解は次式で与えられる．

$$x_i=\left[\frac{S_1^5+S_5}{S_1^3+S_3}\right]^{1/2}\cdot Z_i+S_1 \qquad (12)$$

以上の方法によって，$S_1^5+S_5\neq 0$ の場合はサブルーチン CR 3, 3乗根の場合はサブルーチン CR 33 を用いて(7)式の根 x_1, x_2, x_3 を求め，その指数を SAVE 1, SAVE 2, SAVE 3 に格納する．(7)式の3次方程式の解法アルゴリズムを図 3 に示す．またそのプログラム (CR 3) を図 4 に示す．



(注) 符号長が3で整除できないときは，異なる3つの3乗根は存在しない．

図 3 3次方程式の解法アルゴリズム

Fig. 3 Algorithm for solving a cubic equation.

2.4 誤りビット数の判定

誤り位置方程式の定数項に着目すれば容易に誤りビット数を判定できる．

すなわち，(7)式の定数項が非零であれば3ビット誤りであり，零であれば2ビット以下の誤りである．ここで(7)式において定数項が零の条件を用いて S_5 を消去すると(4)式と一致する．したがって，さらに(4)式の定数項が非零であれば2ビット誤りであり，零であれば1ビット以下の誤りとなる．つぎに $S_1=S_3=S_5=0$ のときは誤りがない．以上の誤りビット数の判定の効率の良いアルゴリズムは次項の図 5 の中に含まれている．

2.5 誤り訂正の実行

誤り位置方程式の根の指数は SAVE 1, SAVE 2, SAVE 3に格納される．いま 15 ビットの受信符号は1語内の0ビット～14ビットの位置に格納されており，このビット位置に α^{14}～α^0 が対応している．したがって，いま SAVE 1 の値が i とすると $(14-i)$ ビットを訂正すれば良い．ただし，使用したマイクロコンピュータの S BiT 命令（レジスタに1をセットする）のビットセット位置は変数ではないので実行中に機械語になっている S BiT 命令のビット指定位置を $(14-i)$ の値に書き換えている．SAVE 2, SAVE 3 に対しても同様である．これはメインプログラムの内部サブルーチン SUB で行っている．

以上述べた動作をまとめた，3重誤り訂正 BCH 符号の復号アルゴリズムを図 5 に示す．プログラムは紙面の都合で省略する．また，本プログラムで復号したときの出力例を図 6 に示す．いま便宜上，誤りのない送信符号（符号語）は allϕ's（最初からの 14 ビット）としている．Oは誤りなし，Cは訂正，Nは訂正不能を示す．4 ビット誤りの場合は訂正不能であり，3 ビット以下の誤りに対しては訂正を行っている．

3. 符号性能の検討

3.1 3重誤り訂正 BCH 符号の復号性能

以上で述べたマイクロコンピュータを用いた3重誤り訂正 BCH 符号の復号性能を表 3 に示す．シンドロームの算出は剰余表を用いた方が高速に演算できる[10),11)]．一例として符号長 127，情報符号長 106 の符号を考えると，(符号長)/(復号時間)≒60K (ボー) なので，4,800 ボーのみならず，48K ボーの高速データ

```
STNO. LABEL   OP.    OPERANDS/COMMENT      STNO. LABEL   OP.    OPERANDS/COMMENT
    1         BGN    CR3                     56          L      R2,(SAVE2)
    2         L      R1,(SAVE1)              57          BAL    KAIX
    3         L      X1,(TBL1)               58          ST     R1,(SAVE2)
    4         A      X1,R1                   59          L      R1,SAVE4
    5         L      R1,0(X1)                60          EOR    R1,R2
    6         ST     R1,(SAVE1)              61          L      R2,(S1)
    7         L      R2,(SAVE2)              62          EOR    R1,R2
    8         L      X1,(TBL1)               63          L      X1,(TBL1)
    9         A      X1,R2                   64          A      X1,R1
   10         L      R2,0(X1)                65          L      R1,0(X1)
   11         S      R2,R1                   66          ST     R1,(SAVE3)
   12         MV     R0,R2                   67          RET
   13         BAL    FUKA3                   68    *
   14         MV     R1,R0                   69    KAIX  A      R1,R2
   15         L      R3,N2                   70          BAL    OVF
   16   P1    CLEAR  R2                      71          L      X1,(TBL2)
   17   P2    SKIP   R0,NZ                   72          A      X1,R1
   18         B      P4                      73          L      R1,0(X1)
   19         S      R0,R3                   74          L      R2,(S1)
   20         SKIP   R0,PZ                   75          EOR    R1,R2
   21         B      P3                      76          MV     R2,R1
   22         AI     R2,1                    77          L      X1,(TBL1)
   23         B      P2                      78          A      X1,R1
   24   P3    AI     R1,15                   79          L      R1,0(X1)
   25         MV     R0,R1                   80          RET
   26         B      P1                      81    *
   27   P4    ST     R2,(SAVE2)              82    FUKA3 SKIP   R0,M
   28         MV     R1,R2                   83          B      J13
   29         A      R1,R2                   84          L      R3,N15
   30         A      R1,R2                   85          A      R0,R3
   31         BAL    OVF                     86    J13   RET
   32         MV     R2,R1                   87    *
   33         L      R1,(SAVE1)              88    OVF   L      R4,N15
   34         S      R1,R2                   89    J5    MV     R3,R4
   35         MV     R0,R1                   90          S      R3,R1,MZ
   36         BAL    FUKA3                   91          B      J6
   37         MV     R1,R0                   92          S      R1,R4
   38         L      X1,(TBL4)               93          B      J5
   39         A      X1,R1                   94    J6    RET
   40         L      R1,0(X1)                95    *
   41         ST     R1,(SAVE3)              96    SAVE1 EXTRN  SAVE1
   42         L      R2,(SAVE2)              97    SAVE2 EXTRN  SAVE2
   43         L      R0,MSK1                 98    SAVE3 EXTRN  SAVE3
   44         AND    R1,R0                   99    S1    EXTRN  S1
   45         BSWP   R1,R1                  100    TBL1  EXTRN  TBL1
   46         BAL    KAIX                   101    TBL2  EXTRN  TBL2
   47         ST     R1,(SAVE1)             102    TBL4  EXTRN  TBL4
   48         ST     R2,SAVE4               103    N2    DC     F'2'
   49         L      R1,(SAVE3)             104    N15   DC     F'15'
   50         SR     R1,RE                  105    MSK1  DC     X'0F00'
   51         SR     R1,RE                  106    MSK2  DC     X'000F'
   52         SR     R1,RE                  107    SAVE4 DS     XL2
   53         SR     R1,RE                  108          END
   54         L      R0,MSK2
   55         AND    R1,R0
```

図 4 3次方程式の解法プログラム

Fig. 4 Program for solving a cubic equation.

表 3 3重誤り訂正 BCH 符号の復号性能

Table 3 Decoding performance of triple-error-correcting BCH codes.

生成多項式(注1)		(4,1,0) (4,3,2,1,0) (2,1,0)	(5,2,0) (5,4,3,2,0) (5,4,2,1,0)	(6,1,0) (6,4,2,1,0) (6,5,2,1,0)	(7,3,0) (7,3,2,1,0) (7,4,3,2,0)
チェックビット長		10	15	18	21
符号長		15	31	63	127
シンドロームの算出	シフト演算	1.3	2.9	6.2	13.0
	剰余表(注2)	0.5	0.5	0.6	1.0
誤りビット数判定 (約 0.4)＋誤り位置方程式の解法 (約 0.48)＋誤り訂正 (約 0.22)＝約 1.1					
復号時間の合計(注3)	シフト演算	2.4	4.0	7.3	14.1
	剰余表	1.6	1.6	1.7	2.1

(単位は ms)

(注1) ( )内は多項式の指数を表わす．(4,1,0) → x^4+x+1

(注2) $m_1(x)$, $m_3(x)$, $m_5(x)$ に対して各々8ビットパターンの剰余表を用いる．(256語のテーブルを3つ)

(注3) シンドロームの計算にシフト演算，および剰余表を用い，ほかの処理は共通である．

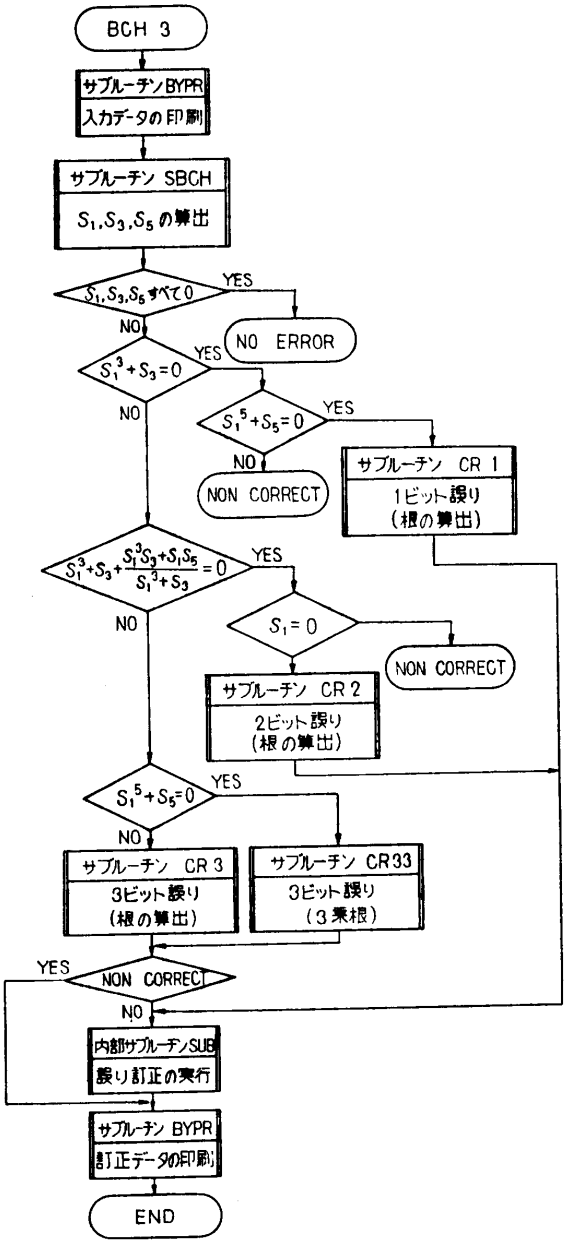


図 5 3重誤り訂正 BCH 符号の復号アルゴリズム
Fig. 5 Algorithm for decoding triple-error-correcting BCH codes.

```
00000000 00000000 00000000 00000000 O
10000000 00000000 00000000 00000000 C
10000100 00000000 00000000 00000000 C
10000100 10000000 00000000 00000000 C
00000011 10000000 00000000 00000000 C
00000011 11000000 00000011 11000000 N
入  力            訂正出力
```

図 6 出　力　例
Fig. 6 A sample output.

通信回線に対しても本方式が有効であり，実用化が期待できよう．

また，本方式を ROM 構成した復号器によれば，主記憶などにも有効となると思われる．

3.2 2重誤り訂正 BCH 符号の復号性能

3重誤り訂正 BCH 符号で述べた方法を用いれば，2重誤り訂正 BCH 符号についても容易に復号プログラムが構成できる．ここではプログラムは省略し，2重誤り訂正 BCH 符号の復号アルゴリズムを**図 7** に示す．

また，その復号性能を**表 4** に示す．

4. あ と が き

BCH 符号の復号は誤りビット数が増加するにしたがって非常に複雑になるが，2重および3重誤り訂正 BCH 符号については，復号アルゴリズム，演算方法などを最適化して高速処理を実現することによりマイ

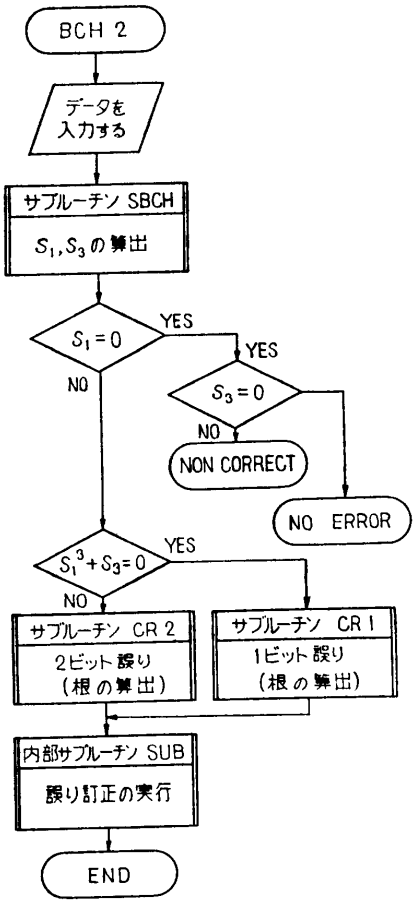


図 7 2重誤り訂正 BCH 符号の復号アルゴリズム
Fig. 7 Algorithm for decoding double-error-correcting BCH codes.

表 4 2重誤り訂正 BCH 符号の復号性能
Table 4 Decoding performance of double-error-correcting BCH codes.

生成多項式		(4,1,0)(4,3,2,1,0)	(5,2,0)(5,4,3,2,0)	(6,1,0)(6,4,2,1,0)	(7,3,0)(7,3,2,1,0)
チェックビット長		8	10	12	14
符号長		15	31	63	127
シンドロームの算出	シフト演算	0.9	1.9	4.0	8.7
	剰余表	0.3	0.3	0.4	0.6
誤りビット数判定（約 0.1）＋誤り位置方程式の解法（約 0.15）＋誤り訂正（約 0.15）＝約 0.4					
復号時間の合計	シフト演算	1.3	2.3	4.4	9.1
	剰余表	0.7	0.7	0.8	1.0

（単位 ms）

クロコンピュータによって比較的簡単に復号できることを示した．また復号性能の検討を行い，本方式は 4,800 ボーのみならず 48K ボーの高速データ通信回線に対しても十分適用できることを明らかにした．

マイクロコンピュータ程度のもので高度な誤り訂正処理が実現され得るので，通信，制御関係のみならず，最近研究の進んでいる誤り訂正符号を用いたデータ処理など広範囲にわたって本方式が適用される可能性がある．

今後はさらに4重誤り訂正 BCH 符号のマイクロコンピュータによる効率の良い復号プログラムの開発を行いたい．また，3重，4重誤り訂正 BCH 符号について，ROM およびマイクロプログラミング技術などを用いた効率の良い復号器の検討を行いたい．

最後に，日頃ご指導頂く広島大学工学部川野董教授，有益なご助言を頂いた同市川忠男教授に深甚なる謝意を表すとともに，プログラムの作成に協力頂いた本学学生桂哲也君（現中国電力），山本聡君（現ジャステック）に感謝の意を表します．

参考文献

1) Peterson, W. W. and Weldon, E. J.: Error-Correcting Codes, 2nd Edition, pp. 269-309, MIT press, Mass. (1972).
2) Berlekamp, E. R.: Algebraic Coding Theory, pp. 176-199, McGrow-Hill Book Co., New York (1968).
3) Frank Polkinghorn: Decoding of Double and Triple Error Correcting Bose-Chaudhuri Codes, IEEE Trans. IT-12, pp. 480-481 (Oct. 1966).
4) 宮川，岩垂，今井：符号理論，pp. 247-273，昭晃堂 (1973).
5) 嵩，都倉，岩垂，稲垣：符号理論，pp. 129-143，コロナ社 (1975).
6) 猪瀬，山本：データ通信，pp. 107-127，産報 (1971).
7) 杉村，笠原，滑川：復号の容易な能率の良い多重誤り訂正符号の一構成法，信学論，J 61-A, 11, pp. 1091-1098 (1978).
8) 今井，藤谷：復号の簡単な誤り訂正符号の一構成法，信学論，J 62-A, 5, pp. 271-277 (1979).
9) 山岸，今井：ROM を用いた復号器の一構成法，信学技報，AL 79-50, pp. 9-16 (1979-10).
10) 岡野博一：マイクロコンピュータによる Fire 符号のシミュレーションプログラム——データ通信への応用——，情報処理学会論文誌，Vol. 20, No. 6, pp. 468-473 (1979).
11) 岡野博一：マイクロコンピュータによる Fire 符号の高速復号法，情報処理学会論文誌，Vol. 21, No. 5 pp. 375-382 (1980).

（昭和 55 年 5 月 26 日受付）
（昭和 55 年 7 月 17 日採録）