

2重および3重誤り訂正 Reed-Solomon 符号の復号†

岡野博一^{††} 守川和夫^{††}

デジタルシステムの信頼性向上のために、誤り訂正符号が広く用いられている。BCH 符号、Reed-Solomon 符号の復号法に関しては、主としてハード面からの研究がなされており、ROM を用いた復号器は実用的であると思われる。一方、ソフト的に実現できれば、システムの変更等にも柔軟に対応できるので高速処理を要しない場合は工学的に実用価値が高いと思われる。そこで、本論文では、2重および3重誤り訂正 Reed-Solomon 符号の復号を、まず、マイクロコンピュータを用いてソフト的に実現するとともに、ついで、ROM を用いた効率的な復号器を提案している。ここで用いたソフト手法は、シンドロームの算出には2値の場合に有効なシフト演算法を拡張した方法を用いた。すなわち、1シンボルに対する剰余表を索表して剰余を求め、その剰余を足し込んで演算を行うことにより高速演算を可能とした。また、誤り位置多項式の解法は定数に対して根のテーブルを索く Polkinghorn の方法を用いた。ROM を用いた復号器も同様な復号アルゴリズムを用いている。復号性能の検討を行い、マイクロコンピュータでソフト的に実現した本方式が高速データ通信回線に対しても実用可能であることを示す。また、ROM を用いた復号器を通常の TTL 回路で構成したとして、復号遅延時間、ゲート数等を評価したところ LSI 化が十分可能であることが明らかとなった。したがって、デジタルシステムへの実用化が期待できる。

1. ま え が き

デジタルシステムの信頼性向上のために、誤り訂正符号が広く用いられている。とくに BCH 符号¹⁾はランダム誤りに対して高い誤り訂正能力をもつ優れた符号である。また、Reed-Solomon (以下 RS と略す) 符号^{1), 3)}は BCH 符号の一種であり、主としてバースト誤り訂正に対して有効である。BCH 符号、RS 符号の復号法に関しては、ハード面からの研究がなされており、ROM を用いた復号器は実用的であると思われる⁴⁾。

一方、筆者らはソフト的に誤り訂正符号を実現することを試み、すでにマイクロコンピュータを用いた Fire 符号、BCH 符号の復号法を提案している⁵⁾⁻⁷⁾。BCH 符号、RS 符号の復号器は複雑になるので、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ろう。とくにマイクロコンピュータを用いてソフト的に実現すればシステムの変更にも柔軟に対応できるので実用的である。しかしながらマイクロコンピュータの処理能力はあまり高くないので、できる限りの高速処理を実現する必要がある。

本論文においては、2重および3重誤り訂正 RS 符

号をマイクロコンピュータで実現するとともに、ROM を用いた実用的な3重誤り訂正 RS 符号の復号器を提案している。ここで用いたソフト手法としては、まずシンドロームを算出する際の剰余演算を高速に行うために、Fire および BCH 符号の際に用いたシフト演算手法を拡張して原理的に1シンボルに対する剰余表を用いて足し込み演算を行う方法を用いた。そして、誤り位置多項式の解法は Polkinghorn の方法²⁾を用い復号処理を効率的にしている。

2. RS 符号の原理

RS 符号は BCH 符号より先に発見されているが、BCH 符号の一種とみなされている。この符号は、特にバースト誤り訂正に有用である。 t 重誤り訂正の RS 符号は、次の多項式で生成される⁸⁾。

$$g(x) = (x - \alpha)(x - \alpha^2) \cdots (x - \alpha^{2^t}) \quad (1)$$

ここで、 α は $GF(2^m)$ の原始元である。以下に RS 符号のパラメータを示す。

シンボル: m ビット/シンボル

ブロック長: $n = (2^m - 1)$ シンボル

$$= m \times (2^m - 1) \text{ ビット}$$

パリティシンボル数: $n - k = 2t$ シンボル

$$= m \times 2t \text{ ビット}$$

最小距離: $d_{\min} = 2t + 1$

さて、本論文で作成したシミュレーションプログラムである3重以下の誤りを訂正する RS 符号のパラメータは次のようになる。なお、2重誤り訂正の場合

† Decoding of Double and Triple Error Correcting Reed-Solomon Codes by HIROKAZU OKANO and KAZUO MORIKAWA (Department of Information and Electronics, Tokuyama Technical College).

†† 徳山工業高等専門学校情報電子工学科

も原理は同様なので簡単な説明に留める。また、生成多項式の根は $\alpha, \alpha^2, \dots, \alpha^{2^i}$ であるが, $\alpha^0, \alpha, \alpha^2, \dots, \alpha^{2^i-1}$ にしてもよい。ここでは後者を用いた。

$$g(x) = (x + \alpha^0)(x + \alpha)(x + \alpha^2)(x + \alpha^3) \times (x + \alpha^4)(x + \alpha^5) \quad (2)$$

ただし, α は $GF(2^6)$ の原始元である。

シンボル: $m=4$ ビット/シンボル

ブロック長: $n=15$ シンボル=60 ビット

パリティシンボル数: $n-k=6$ シンボル=24 ビット

最小距離: $d_{\min}=7$

さて, 送信符号を $f = [f_0, f_1, \dots, f_{n-1}]$ とし, 対応する多項式を $f(x) = f_0 + f_1x + \dots + f_{n-1}x^{n-1}$ とすると, $f(x)$ は $g(x)$ で割り切れる。いま, 受信符号を $r = [r_0, r_1, \dots, r_{n-1}]$ とし, 誤りパターンを $e = [e_0, e_1, \dots, e_{n-1}]$ とする。そして r, e を f と同様の多項式表示すると, $r(x) = f(x) + e(x)$ となる。このとき, シンドローム S_i は次式で表される。

$$S_i = e(\alpha^i) = r(\alpha^i), (i=0 \sim 5) \quad (3)$$

次に, 誤り位置多項式とその Polkinghorn による解法および誤り大きさを表す式を示す²⁾。

2.1 単一誤り

単一誤りのとき, 誤り位置多項式は

$$x + \sigma_{11} = 0 \quad (4)$$

となる。ただし $\sigma_{11} = S_1/S_0$ 。この場合, 誤り位置数 X_1 は(4)式の根となり

$$X_1 = S_1/S_0 \quad (5)$$

となる。また誤りの大きさ Y_1 は次式となる。

$$Y_1 = S_0 \quad (6)$$

2.2 2重誤り

2重誤りのとき, 誤り位置多項式は次式となる。

$$x^2 + \sigma_{21}x + \sigma_{22} = 0 \quad (7)$$

ただし, $\sigma_{21} = (S_0S_3 + S_1S_2)/A$, $\sigma_{22} = (S_1S_3 + S_2^2)/A$ 。ここで $A = S_1^2 + S_0S_2$ である。この場合 $\sigma_{21} \neq 0$ となるので $x = \sigma_{21}y$ とおくと,

$$y^2 + y + \sigma_{22}/\sigma_{21}^2 = 0 \quad (8)$$

となる。この根を定数 $\sigma_{22}/\sigma_{21}^2$ に対応したテーブルに格納しておく。根を y_1, y_2 とすると, 誤り位置数 X_1, X_2 は(7)式の根となり次式となる。

$$X_1 = \sigma_{21}y_1, X_2 = \sigma_{21}y_2 = \sigma_{21} + X_1 \quad (9)$$

また, 誤りの大きさ Y_i は

$$Y_i = \frac{S_0\sigma_{22}/X_i + S_1}{\sigma_{21}} \quad (i=1, 2) \quad (10)$$

となる。または, 次式でも表される。

$$Y_1 = \frac{S_0X_2 + S_1}{\sigma_{21}}, Y_2 = \frac{S_0X_1 + S_1}{\sigma_{21}} \quad (11)$$

2.3 3重誤り

3重誤りのとき, 誤り位置多項式は次式となる。

$$x^3 + \sigma_{31}x^2 + \sigma_{32}x + \sigma_{33} = 0 \quad (12)$$

ただし,

$$\sigma_{31} = (S_1S_3^2 + S_1^2S_5 + S_2^2S_3 + S_0S_2S_5 + S_0S_3S_4 + S_1S_2S_4)/B,$$

$$\sigma_{32} = (S_0S_4^2 + S_2S_3^2 + S_2^2S_4 + S_0S_3S_5 + S_1S_2S_5 + S_1S_3S_4)/B,$$

$$\sigma_{33} = (S_3^3 + S_1S_4^2 + S_2^2S_5 + S_1S_3S_5)/B.$$

ここで, $B = S_2^3 + S_0S_3^2 + S_1^2S_4 + S_0S_2S_4$ である。

(12)式において $x = \sigma_{31} + y$ とおき, さらに $\eta = \sigma_{31}^2 + \sigma_{32}$, $\delta = \sigma_{31}\sigma_{32} + \sigma_{33}$ とすると次式を得る。

$$y^3 + \eta y + \delta = 0 \quad (13)$$

$\eta = 0$ のときは $y = \delta^{1/3}$ となる。つぎに $\eta \neq 0$ のとき $z = \eta^{-1/2}y$ とおき

$$z^3 + z + \delta/\eta^{3/2} = 0 \quad (14)$$

を得る。定数 $\delta/\eta^{3/2}$ に対する(14)式の根のテーブルを索表し, 根を $z_i (i=1, 2, 3)$ とすると, (13)式の根は $y_i = \eta^{1/2}z_i (i=1, 2, 3)$ となる。したがって, 誤り位置数 X_i は次式となる。

$$X_i = \eta^{1/2}z_i + \sigma_{31}, (i=1, 2, 3) \quad (15)$$

また, 誤りの大きさ Y_i は次式となる。

$$Y_i = \frac{S_0\sigma_{33}/X_i + S_1(\sigma_{31} + X_i) + S_2}{X_i^2 + \sigma_{32}}, (i=1, 2, 3) \quad (16)$$

さて, 復号手順を要約すると次のようになる。

(ステップ1) シンドローム S_i の算出

(ステップ2) S_i から誤り位置多項式の係数の算出および誤りシンボル数の判定

(ステップ3) 誤り位置多項式の解法

(ステップ4) 誤りの大きさの算出

(ステップ5) 誤り訂正の実行

3. ソフトプログラムによる復号

3.1 TEC-RS 符号のシミュレーションプログラム

ソフト手法による3重誤り訂正(以下, TEC と略す)RS 符号の復号法について述べる。パラメータは前項で述べた。符号長は15シンボル(60ビット)である。符号化は省略したので, 情報9シンボル(36ビット)を all 0 とし, したがってチェック6シンボル(24ビット)も all 0 を送信符号と考える。誤りを含んだ受信符号はエントリスイッチ(ESW)より入力し,

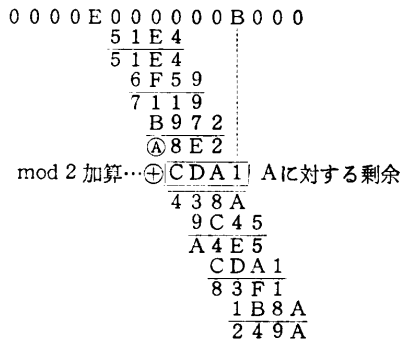


図 1 剰余演算例

Fig. 1 Calculating example of remainder.

RCV~RCV+3 番地の 4 ワードに格納する。

なお、GF(2^m) 上のガロア体の元の演算は、加算はベクトル表現、乗除算は指数表現で行うのが効率が良いので、ベクトル表現⇔指数表現変換テーブルを用いて行う。

まず、シンドロームは(3)式を用いて算出されるが、剰余の定理を用いると効率が良い。いま(3)式から $S_i = r(\alpha^i)$ ($i=0\sim 3$) を同時に算出する場合を考える。 $g_1(x) = \prod_{i=0}^3 (x + \alpha^i)$ として、 $r(x) \div g_1(x)$ の剰余を $R_1(x)$ とすると、 $S_i = R_1(\alpha^i)$ ($i=0\sim 3$) である。剰余演算は BCH 符号の場合は 2 値なのでシフト演算法⁵⁾⁻⁷⁾ を用いるが、RS 符号は多値なのでシフト演算を多値に拡張する必要がある。具体的には 1 シンボルに対する剰余、すなわち $\alpha^j x^s \div g_1(x)$ の剰余 $R_{j1}(x)$ ($j=0\sim 14$) は

$$R_{j1}(x) = \alpha^{12+j} x^{s-1} + \alpha^{4+j} x^{s-2} + \alpha^j x^{s-3} + \alpha^{6+j} x^{s-4} \quad (17)$$

となるので、各シンボルの対応するアドレスに(17)式の各係数を格納したテーブル(以下、剰余表と称す)を作成しておき、この剰余表を用いて受信符号の先頭

の 1 シンボルの剰余を求め、この剰余を次の信号に足し込んで、この動作を繰り返して剰余 $R_1(x)$ を算出する。 S_4, S_5 も同様にして算出する。

一例として、受信符号を $r(x) = \alpha^{11} x^{10} + \alpha^7 x^3$ としたときの $g_1(x)$ に対する剰余演算の過程を図 1 に示す。図において、元はすべてベクトル表現とし、16 進数で表した。たとえば、E、すなわち $[\alpha^{11}]$ の剰余は 51E4、 $[\alpha^8, \alpha^0, \alpha^{11}, \alpha^2]$ であり、足し込み演算を繰り返すと、剰余は $R_1(x) = \alpha x^3 + \alpha^2 x^2 + \alpha^{14} x + \alpha^9$ と求まる。

なお、いま 1 シンボルに対する剰余表を用いたが、複数シンボルに対する剰余表を用いると、より高速に演算できる。ただし、表の容量は大きくなる。

つぎに、シンドロームから誤り位置多項式の係数を求め、その定数項に着目して誤りシンボル数の判定を行う。まず、サブルーチン SGM3 によって、(12)式を用い各係数 $\sigma_{31} \sim \sigma_{33}$ の分母子を求める。ここで分母 B は SG、分子は SG+1~SG+3 に格納される。そこで σ_{33} の分母子が非零であれば、3 シンボル誤りと判定し、サブルーチン CR3D で訂正を行う。また、 σ_{33} の分母子がともに零のとき 2 シンボル以下の誤りであると判定できる。さらに、サブルーチン SGM2 によって(7)式を用い、各係数 σ_{21}, σ_{22} の分母子を求める。A は SG、分子は SG+1, SG+2 に格納される。同様に、 σ_{22} の分母子が非零であれば 2 シンボル誤り、分母子が零のときは 1 シンボル誤りと判定し、それぞれサブルーチン CR2D, CR1D により誤り訂正を実行する。また、 $S_0 \sim S_5 = 0$ のときは誤りなしと判定する。

そして、誤り位置多項式を解き、誤り位置数 X_i を求める。このとき、(8)、(13)式の 2 次、3 次方程式の解法には Polkinghorn の方法²⁾を用い、それぞれ定数 $\sigma_{22}/\sigma_{21}^2, \delta/\eta^{3/2}$ に対する根を格納したテーブルを

表 1 TEC-RS 符号の復号性能
Table 1 Decoding performance of TEC-RS code.

m	4		8				
	(n, k) (シンボル, ビット)	(15, 9) (60, 36)	(14, 8) (56, 32)	(255, 249) (2040, 1992)	(134, 128) (1072, 1024)	(38, 32) (304, 256)	(14, 8) (112, 64)
シンドロームの算出 (ms)	3.10	3.09	34.30	18.64	6.11	2.98	
誤り数の判定 2.00, 誤り位置数 0.68, 誤りの大きさ 0.74, 訂正 0.21 (ms)							
復号時間 (ms)	6.7	6.7	37.9	22.3	9.8	6.6	
復号速度 (baud)	8.9k	8.3k	53.8k	48.1k	31.2k	16.9k	

STNO.	LABEL	OP.	OPERANDS/COMMENT	STNO.	LABEL	OP.	OPERANDS/COMMENT
1	* TRIPLE-ERROR-CORRECTING RS CODE			56	A		R1,RO,NZ
2	BGN		RS3C,,X'200'	57	B		RS3
3	RCV	EQU	X'0'	58	* NON CORRECT ? (SG0=0 OR SG2=0)		
4	EV	EQU	X'5'	59	SKIP		RO,NZ
5	VE	EQU	X'6'	60	B		NOCR
6	SV	EQU	X'7'	61	L		RO,(SG+2)
7	SE	EQU	X'D'	62	SKIP		RO,NZ
8	SG	EQU	X'13'	63	B		NOCR
9	* STACK POINTER			64	* 2 SYMBOLS ERROR CORRECTING		
10	L		SP,SPAD	65	BAL		(CR2D)
11	* RECEIVED VECTOR FROM ESR			66	B		PR
12	CLEAR		X0	67	* NON CORRECT ? (S0=0 OR S1=0)		
13	RS1	H		68	RS3	L	RO,(SV)
14	RD		RO,X'04'	69	SKIP		RO,NZ
15	ST		RO,(RCV)(X0)	70	B		NOCR
16	SI		X0,3,NZ	71	L		RO,(SV+1)
17	B		*+3	72	SKIP		RO,NZ
18	AI		X0,4	73	B		NOCR
19	B		RS1	74	* 1 SYMBOL ERROR CORRECTING		
20	CLEAR		X0	75	BAL		(CR1D)
21	BAL		(PR4W)	76	B		PR
22	* REMAINDER,SYNDROME			77	* 'NC'		
23	BAL		(RMSY)	78	NOCR	CLEAR	X0
24	* NO ERROR ? (S0,..,S5=0)			79	MVI		X0,5
25	L		RO,(SV)	80	B		PR
26	L		R1,(SV+1)	81	* 'NE'		
27	A		RO,R1	82	NOCR	CLEAR	X0
28	L		R1,(SV+2)	83	MVI		X0,1
29	A		RO,R1	84	PR	BAL	(PR4W)
30	L		R1,(SV+3)	85	H		
31	A		RO,R1	86	* STACK		
32	L		R1,(SV+4)	87	DS		XL20
33	A		RO,R1	88	SPAD	DC	A(*-1)
34	L		R1,(SV+5)	89	*		
35	A		RO,R1,NZ	90	RMSY	EXTRN	RMSY
36	B		NOCR	91	SGM3	EXTRN	SGM3
37	BAL		(SGM3)	92	SGM2	EXTRN	SGM2
38	* DOUBLE ERRORS ? (SG0=SG3=0)			93	CR3D	EXTRN	CR3D
39	L		RO,(SG)	94	CR2D	EXTRN	CR2D
40	L		R1,(SG+3)	95	CR1D	EXTRN	CR1D
41	A		R1,RO,NZ	96	PR4W	EXTRN	PR4W
42	B		RS2	97	END		
43	* NON CORRECT ? (SG0=0 OR SG3=0)						
44	SKIP		RO,NZ				
45	B		NOCR				
46	L		RO,(SG+3)				
47	SKIP		RO,NZ				
48	B		NOCR				
49	* 3 SYMBOLS ERROR CORRECTING						
50	BAL		(CR3D)				
51	B		PR				
52	RS2	BAL	(SGM2)				
53	* SINGLE ERROR ? (SG0=SG2=0)						
54	L		RO,(SG)				
55	L		R1,(SG+2)				

図 2 TEC-RS 符号の復号プログラム (メイン部)

Fig. 2 Main program for decoding TEC-RS code.

用いて方程式を解く。Polkinghorn の解法およびそのプログラムについては文献 7) に詳述しているのでここでは省略する。ついで (6), (11), (16) 式を用いて誤りの大きさ Y_i を求め、誤り位置数 X_i の値から誤り訂正を行うワードのシンボル位置を決め、そのワード内の対応するシンボルを誤りの大きさ Y_i で mod 2 加算を行って誤りを訂正する。

以上述べた動作の TEC-RS 符号の復号プログラム

(メインプログラム部) を図 2 に示す。

3.2 評 価

以上に述べた TEC-RS 符号の復号アルゴリズムを用いて、マイクロコンピュータでシミュレートした復号性能を表 1 に示す。また、同様な復号アルゴリズムを用いれば 2 重誤り訂正 RS (以下、DEC-RS と称す) 符号についても容易にシミュレーションプログラムが作成できる。表 2 にその復号性能を示す。なお、

表 2 DEC-RS 符号の復号性能
Table 2 Decoding performance of DEC-RS code.

m	4		8			
(n, k) (シンボル, ビット)	(15, 11) (60, 44)	(12, 8) (48, 32)	(255, 251) (2040, 2008)	(132, 128) (1056, 1024)	(36, 32) (288, 256)	(12, 8) (96, 64)
シンδροームの算出 (ms)	1.98	1.76	19.45	10.49	3.56	1.82
誤り数の判定 0.48, 誤り位置数 0.23, 誤りの大きさ 0.19, 訂正 0.14 (ms)						
復号時間 (ms)	3.0	2.8	20.5	11.5	4.6	2.9
復号速度 (baud)	19.9k	17.1k	99.6k	91.6k	62.6k	33.6k

情報ビットが少ない場合は、短縮化符号を用いた場合である。

ブロック長に依存するのはシンδροームの算出だけであり、シンボル単位の剰余表を索表する高速剰余演算によってシンδροームの算出の高速処理を可能とした。なお、2シンボルに対応した剰余表を用いれば、より高速な剰余演算が行える。

一例として、シンボル数 $m=4$ 、ブロック長 $n=15$ 、情報シンボル数 $k=9$ の TEC-RS 符号を考えると、復号速度 = (符号長)/(復号時間) は約 9k baud、また、 $m=8, n=255, k=249$ の符号の場合、復号速度は約 54k baud となる。4.8k baud のみならず、それ以上の高速データ通信回線に対しても本方式が十分適用できることが明らかとなった。

なお、使用したマイクロコンピュータ PFL-16A は、2MHz のクロックで動作し、大半の命令実行時間は、3~8 μ s である。最近の 8086, 68000 など CPU にした高速実行タイプのマイクロコンピュータを用いれば、さらに 4~5 倍程度の高速化が可能であり、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るであろう。

4. ROM を用いた TEC-RS 復号器

4.1 復号器の構成

上記と同様の復号アルゴリズムを用いてハード的に構成した TEC-RS 符号の復号器について述べる。テーブルは ROM (Read Only Memory) を用いている。基本演算回路の略記号を図 3 に示す。+ は元の乗算回路、- は元の除算回路、 $\times 2, \times 3$ はそれぞれ元の 2 乗、3 乗回路、 $\div 2$ は元の平方根回路である。また、AD は指数表現の元の加算を求める回路である。なお、回路の具体的構成などの詳細は文献 4) を参照さ

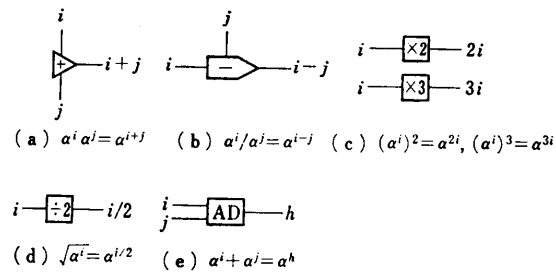


図 3 基本演算回路

Fig. 3 Circuits for some fundamental operations.

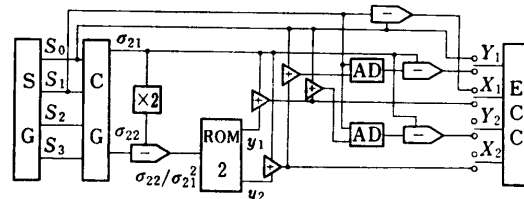


図 4 DEC-RS 符号の復号器

Fig. 4 Decoder of DEC-RS code.

れたい。

まず、図 4 に DEC-RS 符号の復号器を示す。この回路は 2 重以下の誤りを訂正する復号器であり、文献 4) に示されているので説明は省略する。

つぎに、TEC-RS 符号の復号器を図 5 に示す。この回路は $S_0 \sim S_6$ を用いて誤りを訂正する回路である。なお、 $S_1 \sim S_6$ を用いた同様の回路を筆者らはすでに発表している⁸⁾。図 5 の回路は 3 シンボル誤りを訂正する回路なので、2 シンボル以下の誤りを訂正するための図 4 の回路と切り換えて動作させる必要がある。すなわち、(12) 式の σ_{33} の分母子が非零のとき図 5 の回路を動作させ、3 シンボル誤りを訂正する。SG はシンδροーム生成回路であり、ROM を用いて指数表現で出力する。この回路における GF(2^m) 上の演算は

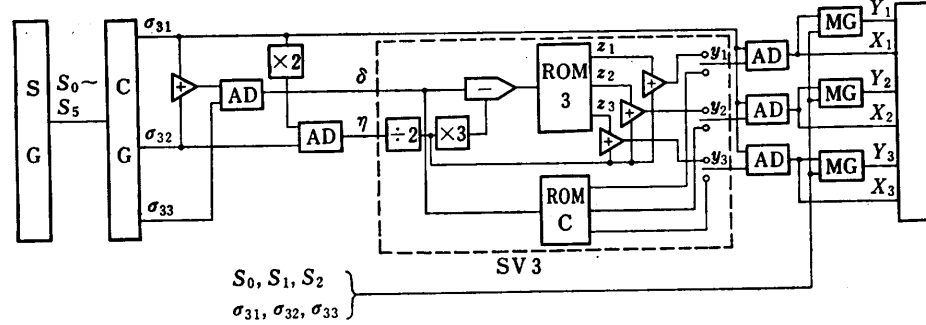


図5 TEC-RS 符号の復号器
Fig. 5 Decoder of TEC-RS code.

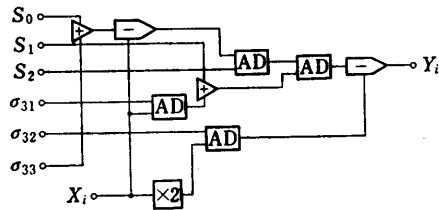


図6 誤りの大きさを算出する回路
Fig. 6 Circuit for calculating magnitude of error.

すべて指数表現で統一して行う。CG はシンドローム $S_0 \sim S_5$ から誤り位置多項式の係数 $\sigma_{31} \sim \sigma_{33}$ を求める回路である。また、誤り数を判定する制御回路もこのなかに含まれる。SV3 は3次方程式 $y^3 + \eta y + \delta = 0$ を Polkinghorn の解法アルゴリズムを用いて解く回路である^{4), 8)}。MG は誤りの大きさ $Y_i (i=1, 2, 3)$ を求める回路であり、その構成を図6に示す。そして、誤り位置数 X_i とその大きさ Y_i をもとに ECC で訂正を実行する。

4.2 評価

図5に示した TEC-RS 符号の復号器の評価を表3に示す。比較のため図4の DEC-RS 符号の評価も付記する。ただし、評価は2章で述べた(ステップ2)~(ステップ4)について行ったものである。

遅延時間は最も長い径路についての値である。ゲート数は TTL 回路で構成した場合を NAND, NOR ゲートで換算した値、チップ数は TTL IC の素子数であり、ROM 数を含んだ値である。各構成要素の評価は文献4)と同じ値を用いた。

なお、シンドローム生成回路と誤り訂正を実行する回路の評価は省略した。したがって、実際の復号時間は表3の値にこの両者の遅延時間約 100ns を加えた値となる⁴⁾。また、TEC-RS 符号の復号器は図5の部分だけなので、DEC-RS 符号の復号器の評価を加える必要がある。ただし、遅延時間は加えなくてよい。

以上の評価は通常の TTL を用いた場合であり、さ

表3 復号器の評価
Table 3 Estimation of decoders.

符号	遅延時間 (ns)	ゲート数	ROM 数	チップ数
DEC-RS 符号	790	3,619	13	273
TEC-RS 符号	1,610	16,297	45	1,202

らに高速な ECL 回路などを用いればさらに高速な復号器が実現できる。さらに、ゲート数の評価から1チップの LSI によって構成可能であると考えられる。

また、シンドローム $S_0 \sim S_5$ を用いた場合と $S_1 \sim S_6$ とした場合について復号性能を検討した結果、復号器の構成において S_0 を算出する際、受信符号の各シンボルを mod 2 加算するだけでよく、また誤りの大きさを求める回路が若干簡単になるが、ハードウェア量、遅延時間とも大差ないことが判明したので両者の比較は省略した。

5. むすび

本論文において、2重および3重誤り訂正 RS 符号の復号法について、ソフトおよびハードの両面から実現し、それらの復号性能の検討を行った。

ソフト手法による復号アルゴリズムにおいて、剰余を求める際、剰余表を索表することにより高速に復号することができることを示した。この結果、たとえば $m=4, (n, k)=(15 \text{ シンボル}, 9 \text{ シンボル})$ の TEC-RS 符号はおよそ 9k baud まではマイクロコンピュータ程度のものでソフト的に実現できることがわかり、本方式のデータ通信、制御回線等のデジタル伝送に対する実用化が期待できる。

また、ROM を用いた復号器は、通常の TTL 回路で構成すると、 $m=8$ の TEC-RS 符号で遅延時間 1.6 μs 、ハードウェア量としてゲート数 16,000 程度で実現可能である。この復号器の復号遅延時間は符号長に無関係であり、非常に効率がよく、構成も簡単であ

る。したがって、DAD (Digital Audio Disk), およびデジタルテレビジョン等のデジタルシステムへの実用化が期待できよう。

謝辞 最後に、日頃ご指導いただく広島大学工学部市川忠男教授, また有益なご助言をいただいた山口大学工学部高浪五男教授に深甚なる謝意を表します。また, プログラムの作成に協力いただいた学生, 石丸輝太氏 (現三菱レーヨン) に感謝します。

参 考 文 献

- 1) Peterson, W. W. and Weldon, E. J.: *Error Correcting Codes*, 2nd Ed., pp. 269-309, MIT Press, Mass. (1972).
- 2) Polkinghorn, F.: Decoding of Double and Triple Error Correcting Bose-Chaudhuri Codes, *IEEE Trans. Inf. Theory*, Vol. IT-12, pp. 480-481 (1966).
- 3) 宮川, 岩垂, 今井: 符号理論, pp. 247-273, 昭晃堂, 東京 (1973).
- 4) 山岸, 今井: ROM を用いた BCH 符号の一構成法, 信学論 (D), Vol. J63-D, No. 12, pp. 1034-1041 (1980).
- 5) 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, Vol. 20, No. 6, pp. 468-473 (1979).
- 6) 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, Vol. 21, No. 5, pp. 375-382 (1980).
- 7) 岡野博一: 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, 情報処理, Vol. 21, No. 6, pp. 500-506 (1980).
- 8) 岡野博一: ROM を用いた BCH 符号復号器の改良, 信学技報, AL82-56 (1982).
- 9) Bhargava, V. K. et al. (塚本賢一他訳): 最近デジタル衛星通信, pp. 400-402, 日本技術経済センター, 東京 (1983).
- 10) 岡野, 石丸: マイクロコンピュータによるリード・ソロモン符号の復号, 情報処理学会昭和57年度後期全国大会論文集, pp. 45-46 (1982).

(昭和 59 年 1 月 23 日受付)

(昭和 59 年 4 月 17 日採録)