INFORMATION AND CONTROL 10, 616–630 (1967)

Synchronizable Error-Correcting Codes*

R. C. Bose and J. G. Caldwell

Department of Statistics, University of North Carolina, Chapel Hill, North Carolina 27514

A new technique for correcting synchronization errors in the transmission of discrete-symbol information is developed. The technique can be applied to any t-additive-error-correcting Bose–Chaudhuri–Hocquenghem code, to provide protection against synchronization errors. The synchronization error is corrected at the first complete received word after the word containing the synchronization error, even if this following word contains up to t additive errors. An example is presented illustrating in detail the application of the technique.

I. INTRODUCTION

In order for digital information to be accurately and efficiently transmitted over a noisy channel, efficient procedures for eliminating or determining the effect of the noise must be devised. Considerable research has been performed to determine means to accomplish reliable transmission in the presence of additive noise, i.e., noise which may cause transmitted symbols to be altered, or changed into other symbols. An effective means for coping with additive errors is to employ an additive-error-correcting code. For channels in which noise affects successive symbols independently, one of the best among the known classes of additive-error-correcting codes is the class of Bose–Chaudhuri–Hocquenghem, or BCH, codes.

Whether or not additive errors are of concern in a particular situation, there may occur a much more serious kind of error. This second type of error arises due to the fact that the individual symbols of a sequence of symbols have physical meaning to the receiver only when considered together with certain other symbols of the sequence. Generally, the

* This research was supported in part by the National Science Foundation Grant no. GP-5790 and Army Research Office, Durham Grant no. DA-ARO-31-124-G670.

sequence of received symbols must be correctly grouped into "words," or "frames," in order for the receiver to properly understand the message. When noise is such that the receiver incorrectly groups the symbols into words, reception is said to be out of synchronization with transmission, and a synchronization error is said to have occurred. Note that a synchronization error may be considered to be a loss or gain of a certain number of symbols in transmission.

In contrast to the situation for additive errors, research concerned with the development of efficient techniques for synchronization-error correction has been limited. This paper presents a new technique for synchronization-error correction. The technique can be applied to any cyclic additive-error-correcting code, and enables immediate correction of synchronization errors, simultaneously with the correction of additive errors.

Just as BCH codes exist for a range of values of t, the number of additive errors allowed per code word, the new synchronization technique can be applied to provide protection against synchronization errors involving a range of symbol losses or gains. If the new technique is chosen so that up to t_l symbol losses can be corrected and up to t_r symbol gains can be corrected, then we say that the code to which the technique is applied is a t_s-synchronization-error-correcting code, where $t_s = t_l + t_r$. If the technique is applied to a t-additive-error-correcting BCH code, we call the resulting code a (t, t_s)-error-correcting code. The resulting code can simultaneously correct synchronization errors and additive errors, and we refer to such a code as a synchronizable error-correcting code.

Early work concerned with mathematical analysis of the synchronization problem was done by Barker (1953). Recent work aimed at finding synchronizable error-correcting codes has been done by Stiffer (1965), Levy (1966), and Tong (1966). Many others have studied the synchronization problem, and an extensive bibliography of articles relating to synchronization is given by Caldwell (1966).

II. BCH CODES

Because the synchronizable error-correcting codes to be developed later will be based on BCH codes, we shall give here a brief description of these codes. (Bose and Ray-Chaudhuri, 1960a, b; and Hocquenghem, 1959).

Consider a q-ary channel, i.e., a channel capable of transmitting q

distinct symbols where q is a prime or the power of a prime. Let the Galois field $GF(q)$ be extended to $GF(q^m)$, and let α be an element of the extended field such that $1, \alpha, \alpha^2, \cdots, \alpha^{n-1}$ are all different and $\alpha^n = 1$. Then n is a divisor of $q^m - 1$, and if θ is a primitive element of $GF(q^m)$ then $\alpha = \theta^u$, where $un = q^m - 1$. Consider the matrix

$$H_0 = \begin{bmatrix} 1 & \alpha^{m_0} & (\alpha^{m_0})^2 & \cdots & (\alpha^{m_0})^{n-1} \\ 1 & \alpha^{m_0+1} & (\alpha^{m_0+1})^2 & \cdots & (\alpha^{m_0+1})^{n-1} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ 1 & \alpha^{m_0+d-2} & (\alpha^{m_0+d-2})^2 & & (\alpha^{m_0+d-2})^{n-1} \end{bmatrix}.$$

Each element of $GF(q^m)$ can be expressed as a polynomial of degree $m - 1$ of the primitive root θ, the coefficients of the polynomial belonging to $GF(q)$. Hence any element can be identified with an m-vector over $GF(q)$, viz., the coefficient vector of the corresponding polynomial. Hence the matrix H_0 can be regarded either as a $(d - 1) \times n$ matrix over $GF(q^m)$ or as a $(d - 1) \times mn$ matrix over $GF(q)$, on identifying the elements of $GF(q^m)$ with column m-vectors over $GF(q)$. In particular the element 1 is identified with the transpose of $(1, 0, 0, \cdots, 0)$. When regarded in this second way, the rank of H_0 is $r \leqq m(d - 1)$. If H is the matrix obtained from H_0 by retaining r suitably chosen independent rows then it is known, Peterson (1961), that H is the parity-check matrix of an (n, k) BCH code C, with minimum distance d and redundancy r, $(k = n - r)$. If $d = 2t + 1$, then the code will be t-error-correcting.

Let $g_i(x)$ be the minimum function of α^i over $GF(q)$, i.e., $g_i(x)$ is the smallest-degree monic polynomial over $GF(q)$ which has α^i for a root. Then the degree of $g_i(x)$ is a divisor of m, and therefore cannot exceed m. Let

$$g(x) = \text{L.C.M. } \{g_{m_0}(x), g_{m_0+1}(x), \cdots, g_{m_0+d-2}(x)\},$$

then $g(x)$ is the smallest degree monic polynomial over $GF(q)$ which has roots $\alpha^{m_0}, \alpha^{m_0+1}, \cdots, \alpha^{m_0+d-2}$, and is the generator polynomial of the BCH code C. The vector $\mathbf{v}' = (v_0, v_1, \cdots, v_{n-1})$ is a word of C if and only if the corresponding polynomial $v(x) = v_0 + v_1x + \cdots + v_{n-1}x^{n-1}$ is a multiple of $g(x)$; i.e., $v(x) = g(x)\Phi(x)$ where $\Phi(x)$ is a polynomial of degree $k - 1$ or less over $GF(q)$. The code C is cyclic; i.e., if $\mathbf{v}' = (v_0, v_1, \cdots, v_{n-1})$ is a word of C, then so is $\mathbf{v}'(i) = (v_i, v_{i+1}, \cdots, v_{n-1}, v_0, v_1, \cdots, v_{i-1})$. The generator polynomial $g(x)$ is of degree r

and is a divisor of $x^n - 1$; i.e., we can find a polynomial $h(x)$ over $GF(q)$ such that $g(x)h(x) = x^n - 1$.

The BCH code C is said to be primitive if $u = 1$, i.e., α is a primitive element of $GF(q^m)$ and $n = q^m - 1$.

III. THE ENCODING PROCEDURE FOR A CLASS OF SYNCHRONIZABLE CODES

Let C be the BCH code described in Section II, where $d = 2t + 1$, so that C is t-error-correcting. Let β be a root of $x^n - 1 = 0$ but not a root of $g(x)$. Thus β is a root of $h(x)$. Let $f(x)$ be the minimum function of β, i.e., $f(x)$ is the smallest-degree monic polynomial over $GF(q)$ which has β for a root. The degree of $f(x)$ will not exceed m, and will be m_1 where m_1 is a divisor of m. In this case β will belong to a subfield of order q^{m_1} of the field $GF(q^m)$. The polynomial $f(x)$ will be a divisor of $h(x)$. Let n_1 be the order of β, i.e., n_1 is the smallest positive (nonzero) integer such that $\beta^{n_1} = 1$. Then n_1 is a divisor of n.

Let C^* be the subcode of C generated by $g(x)f(x)$; i.e., the polynomial corresponding to a word of C^* is divisible by $g(x)f(x)$. Then C^* is an (n, k^*) code where $k^* = n - r - m_1$. Any word $\mathbf{v}' = (v_0, v_1, \cdots, v_{n-1})$ of C^* satisfies

$$\mathbf{v}'[H', H_1'] = \mathbf{0},$$

where H is the parity-check matrix of C, and

$$H_1 = [1, \beta, \beta^2, \cdots, \beta^{n-1}].$$

As before, H_1 may be regarded as either a row vector over $GF(q^m)$ or an $m \times n$ matrix over $GF(q)$.

Let $\mathbf{c}' = (c_0, c_1, \cdots, c_{n-1})$ be a fixed nonnull word of C, which does not belong to the subcode C^*. Let $t_s = t_l + t_r < n_1$ (the order of β). It is our objective to construct a code which can correct a shift of order up to t_l to the left or a shift of order up to t_r to the right. Since $t_s > 0$, the requirement $t_s < n_1$ implies $n_1 > 1$. Thus $\beta \neq 1$. Corresponding to $\mathbf{v}'$ and $\mathbf{c}'$, we now take augmented words

$$\mathbf{v}_a' = (v_{n-t_r}, v_{n-t_r+1}, \cdots, v_{n-1} \vdots v_0, v_1, \cdots, v_{n-1} \vdots v_0, v_1, \cdots, v_{t_l-1}),$$

$$\mathbf{c}_a' = (c_{n-t_r}, c_{n-t_r+1}, \cdots, c_{n-1} \vdots c_0, c_1, \cdots, c_{n-1} \vdots c_0, c_1, \cdots, c_{t_l-1}).$$

Thus we buffer $\mathbf{v}'$ by cyclically adding t_r symbols to the left and t_l symbols to the right of $\mathbf{v}'$. A similar procedure is adopted for buffering $\mathbf{c}'$. We now consider a new code C_a whose words are $\mathbf{v}_a' + \mathbf{c}_a'$. The words of C_a are in $(1, 1)$ correspondence with the words of C^*. Hence the number of message sequences is the same as for C^*, viz., q^{k^*}. When we want to send a message corresponding to the word $\mathbf{v}'$ of C^* we shall actually transmit $\mathbf{v}_a' + \mathbf{c}_a'$. The length of the new code is $n_a = n + t_r + t_l$, and the number of information places is $k_a = k^* = n - r - m_1$. Hence the redundancy is $r_a = r + t_s + m_1$, where $t_s = t_r + t_l$ is the sum of the orders of the maximum shifts to the right and left which are to be corrected.

IV. THE DECODING PROCEDURE

Suppose $\mathbf{v}_a' + \mathbf{t}_a'$ is transmitted, and the additive-error vector is

$$\mathbf{e}_a' = (f_{n-t_r}, \cdots, f_{n-1}, \vdots\, e_0, e_1, \cdots, e_{n-1}, \vdots\, f_0, f_1, \cdots, f_{t_l-1}).$$

Thus the received vector will be

$$\mathbf{y}_a' = \mathbf{v}_a' + \mathbf{c}_a' + \mathbf{e}_a',$$

if there is no shift error. If there is a shift of L places to the left, $L \leqq t_l$, then L of the initial symbols of $\mathbf{y}_a'$ will go over to the previous word, and the received word will contain in the end L symbols from the beginning of the succeeding word. Similarly if there is a shift of R places to the right, $R \leqq t_r$, then R of the end symbols of $\mathbf{y}_a'$ will be shifted to the subsequent word, and in the beginning of the received word we will have R symbols from the end of the previous word. The decoding proceeds step by step as follows:

Step I. We form the truncated received word $\mathbf{y}'$ by dropping the first t_r and the last t_l symbols of the received word $\mathbf{y}_a'$. The truncated received word is of length n. Note that the symbols dropped are just those which in an extreme case under the permissible synchronization errors could have come from a previous or a subsequent word. We now consider three cases separately.

Case (i). If there are no synchronization errors the truncated received word will be

$$\mathbf{y}' = (v_0, v_1, \cdots, v_{n-1}) + (c_0, c_1, \cdots, c_{n-1}) + (e_0, e_1, \cdots, e_{n-1})$$
$$= \mathbf{v}' + \mathbf{c}' + \mathbf{e}'.$$

Case (ii). If there is a left shift of $L \leqq t_l$ places, then the truncated received word will be

$$\begin{aligned}
\mathbf{y}' &= (v_L, v_{L+1}, \cdots, v_{n-1}, v_0, \cdots, v_{L-1}) \\
&\quad + (c_L, c_{L-1}, \cdots, c_{n-1}, c_0, c_1, \cdots, c_{L-1}) \\
&\quad + (e_L, e_{L+1}, \cdots, e_{n-1}, f_0, f_1, \cdots, f_{L-1}) \\
&= \mathbf{v}'(L) + \mathbf{c}'(L) + \mathbf{e}_L'.
\end{aligned}$$

where we use the notation $\mathbf{v}'(i)$ to denote $(v_i, v_{i-1}, \cdots, v_{n-1}, v_0, v_1, \cdots, v_{i-1})$.

Case (iii). Similarly, if there is a right shift of $R \leqq t_r$ places, then the truncated received word will be

$$\begin{aligned}
\mathbf{y}' &= (v_{n-R}, \cdots, v_{n-1}, v_0, v_1, \cdots, v_{n-R-1}) \\
&\quad + (c_{n-R}, \cdots, c_{n-1}, c_0, c_1, \cdots, c_{n-R-1}) \\
&\quad + (f_{n-R}, \cdots, f_{n-1}, e_0, e_1, \cdots, e_{n-R-1}) \\
&= \mathbf{v}'(n-R) + \mathbf{c}'(n-R) + \mathbf{e}_{n-R}'.
\end{aligned}$$

Step II. We form the additive-error syndrome $\mathbf{y}'H'$. Note that from the cyclic nature of BCH codes, $\mathbf{v}'$, $\mathbf{v}'(L)$ or $\mathbf{v}'(n - R)$ in cases (i), (ii), and (iii), respectively, will belong to the code C. Similarly $\mathbf{c}'$, $\mathbf{c}'(L)$, or $\mathbf{c}'(n - R)$ will belong to the C. Hence

$$\begin{aligned}
\mathbf{y}'H' &= \mathbf{e}'H', \text{ in case (ii)}, \\
\mathbf{y}'H' &= \mathbf{e}_L'H', \text{ in case (ii)}, \\
\mathbf{y}'H' &= \mathbf{e}_{n-R}'H', \text{ in case (iii)}.
\end{aligned}$$

By assumption, the number of additive errors is less than or equal to t, so that $wt(\mathbf{e}_a') \leqq t$. Consequently $wt(\mathbf{e}')$, $wt(\mathbf{e}_L')$ or $wt(\mathbf{e}_{n-R}')$ will be less than or equal to t. Then as for BCH codes there will be a (1, 1) correspondence between the error vector and the syndrome. Hence the error vector can be determined by using any error-correction procedure for t-error correcting BCH codes.

Step III. The received truncated word $\mathbf{y}'$ is now corrected for additive errors by subtracting the determined error vector $\mathbf{e}'$, $\mathbf{e}_L'$, or $\mathbf{e}_{n-R}'$. We thus obtain

$$\begin{aligned}
\mathbf{z}' &= \mathbf{v}' + \mathbf{c}' \text{ in case (i)}, \\
\mathbf{z}' &= \mathbf{v}'(L) + \mathbf{c}'(L) \text{ in case (ii)}, \\
\mathbf{z}' &= \mathbf{v}'(n-R) + \mathbf{c}'(n-R) \text{ in case (iii)}.
\end{aligned}$$

Step IV. We now form the shift-error syndrome $\mathbf{z}'H_1'$. Since from our method of formation, the subcode C^* is also a cyclic code, $\mathbf{v}'(L)$ and $\mathbf{v}'(n - R)$ belong to C^* in cases (ii) and (iii), respectively.

Hence we obtain

$$\mathbf{z}'H_1' = \mathbf{c}'H_1' \text{ in case (i)},$$
$$\mathbf{z}'H_1' = \mathbf{c}'(L)H_1' \text{ in case (ii)},$$
$$\mathbf{z}'H_1' = \mathbf{c}'(n - R)H_1' \text{ in case (iii)}.$$

Since the order of β is n_1, a divisor of n, we have $\beta^{n_1} = 1$, $\beta^n = 1$. Also $1, \beta, \beta^2, \cdots, \beta^{n_1-1}$ are all different. Now

$$\mathbf{c}'H_1' = c_0 + c_1\beta + c_2\beta^2 + \cdots + c_{n-1}\beta^{n-1} = \zeta, \text{ say},$$

where ζ is a known element of $GF(q^m)$, since $\mathbf{c}'$ and β are known. Again,

$$\mathbf{c}'(L)H_1' = c_L + c_{L+1}\beta + c_{L+2}\beta^2 + \cdots + c_{L-1}\beta^{n-1} = \beta^{-L}\zeta = \beta^{n_1-L}\zeta,$$

$$\mathbf{c}'(n - R)H_1' = c_{n-R} + c_{n-R+1}\beta + c_{n-R+2}\beta^2 + \cdots + c_{n-R-1}\beta^{n-1} = \beta^R\zeta.$$

Step V. Divide the shift-error syndrome by the known element ζ, obtaining 1, β^{n_1-L}, β^R in cases (i), (ii), and (iii), respectively. Now $1 \leqq L \leqq t_l$, $1 \leqq R \leqq t_r$, and by supposition $t_s = t_r + t_l < n_1$. Hence $n_1 - L > t_r$. Thus if the answer in step V is 1, we conclude that there is no shift error. If the answer in step V is β^u where $u > t_r$ we conclude that a left shift of order $L = n_1 - u$ has occurred. Again, if $1 \leqq u \leqq t_r$, we conclude that a right shift of order $R = u$ has occurred.

We can now correct $\mathbf{z}'$ by applying the reverse shift, and obtain $\mathbf{v}' + \mathbf{c}'$. Finally, by subtracting $\mathbf{c}'$ we obtain $\mathbf{v}'$, the word of C^* which corresponds to the message sent.

It should be remembered that in applying this procedure, synchronization errors can be corrected at the word following that where information symbols have been lost or gained, and not in the damaged word itself.

V. EXAMPLE

Application of the Technique to a BCH Binary Code of Length $n = 15$

To illustrate the new synchronization technique, we shall consider the BCH code of length $n = 2^m - 1 = 2^4 - 1 = 15$ which originally was

TABLE 1

NONZERO ELEMENTS OF $GF(2^4)$, EXPRESSED AS POWERS OF THE ROOT $\alpha = (0, 1, 0, 0)$ OF THE MINIMUM FUNCTION $g_1(x) = x^4 + x + 1$[a]

α^0	$= 1$	$= (1, 0, 0, 0)$
α	$= x$	$= (0, 1, 0, 0)$
α^2	$= x^2$	$= (0, 0, 1, 0)$
α^3	$= x^3$	$= (0, 0, 0, 1)$
α^4	$= 1 + x$	$= (1, 1, 0, 0)$
α^5	$= x + x^2$	$= (0, 1, 1, 0)$
α^6	$= x^2 + x^3$	$= (0, 0, 1, 1)$
α^7	$= 1 + x + x^3$	$= (1, 1, 0, 1)$
α^8	$= 1 + x^2$	$= (1, 0, 1, 0)$
α^9	$= x + x^3$	$= (0, 1, 0, 1)$
α^{10}	$= 1 + x + x^2$	$= (1, 1, 1, 0)$
α^{11}	$= x + x^2 + x^3$	$= (0, 1, 1, 1)$
α^{12}	$= 1 + x + x^2 + x^3$	$= (1, 1, 1, 1)$
α^{13}	$= 1 + x^2 + x^3$	$= (1, 0, 1, 1)$
α^{14}	$= 1 + x^3$	$= (1, 0, 0, 1)$

[a] The polynomial expression for each power of α is obtained by using the relation $\alpha = x$, $x^4 = x + 1$.

presented by Bose and Ray-Chaudhuri (1960a). Over the coefficient field $GF(2)$ we have the factorization

$$\begin{aligned} x^{15} - 1 &= g_1(x)g_3(x)g_5(x)g_7(x)(x + 1) \\ &= (x^4 + x + 1)(x^4 + x^3 + x^2 + x + 1)(x^2 + x + 1) \\ &\quad \cdot (x^4 + x^3 + 1)(x + 1), \end{aligned}$$

where $g_i(x)$ denotes the minimum function of α^i. We choose

$$\begin{aligned} g(x) &= g_1(x)g_3(x) \\ &= (x^4 + x + 1)(x^4 + x^3 + x^2 + x + 1) \\ &= 1 + x^4 + x^6 + x^7 + x^8 \end{aligned}$$

for the generator polynomial of the code C. The code C is an $(n, k) = (15, 7)$ code. The Galois field $GF(2^4)$ is based on the primitive polynomial $g_1(x) = x^4 + x + 1$. All the nonzero elements of $GF(2^4)$ can thus be written as powers of the root $\alpha = (0, 1, 0, 0)$ of $g_1(x)$, and they are shown in Table 1. The zero element is, of course, $(0, 0, 0, 0)$. The roots of the primitive polynomial $g_1(x)$ are

$$\alpha, \alpha^2, \alpha^4, \alpha^8,$$

and the roots of $g_3(x)$ are

$$\alpha^3, \alpha^6, \alpha^{12}, \alpha^{24} = \alpha^9.$$

Thus,

$$\alpha, \alpha^2, \alpha^3, \alpha^4, \alpha^6, \alpha^8, \alpha^9, \alpha^{12}$$

are all the roots of $g(x)$, and the code is $(t = 2)$-additive-error-correcting, since the first $2t = 2 \cdot 2 = 4$ successive powers of α are roots of $g(x)$. To use the new synchronization technique, we must choose $\beta = \alpha^j \neq 1$ such that β is not a root of $g(x)$. Equivalently, we must choose for $f(x)$ a minimum function $g_i(x) \neq x - 1$ such that $g_i(x)$ is not a factor of $g(x)$. Now the factors of $x^{15} - 1$ other than $(x - 1)g(x)$ are $g_5(x)$ and $g_7(x)$. The factor $g_5(x)$ has roots α^5 and α^{10}, and the factor $g_7(x)$ has roots α^7, α^{14}, α^{13}, and α^{11}. Thus we can choose either $j = 5$ or $j = 7$, corresponding to $f(x) = g_5(x)$ or $g_7(x)$. Let us suppose further that we wish to correct a single left-shift or right-shift error, so that $t_l = t_r = 1$, and therefore $t_s = 2$. The final requirement on j is that $t_s < n_1$ where n_1 is the order of α^j. Now both $g_5(x)$ and $g_7(x)$ satisfy $2 = t_s < n_1$, since the order of α^5 is 3 and the order of α^7 is 15. Thus $g_5(x)$ and $g_7(x)$ are both acceptable choices for $f(x)$. However, in order to add as little redundancy as possible for synchronization purposes, we choose for $f(x)$ the acceptable polynomial of least *degree* satisfying $t_s < n_1$. Hence we take $f(x) = g_5(x)$. Thus the subcode C^* has the generator polynomial

$$\begin{aligned} g^*(x) &= g(x)g_5(x) \\ &= (1 + x^4 + x^6 + x^7 + x^8)(1 + x + x^2) \\ &= 1 + x + x^2 + x^4 + x^5 + x^8 + x^{10} \end{aligned}$$

and C^* is thus an $(n, k^*) = (15, 5)$ code. The subcode C^* has the roots

$$\alpha, \alpha^2, \alpha^3, \alpha^4, \alpha^5, \alpha^6, \alpha^8, \alpha^{10}, \alpha^{12}.$$

Such a code, if used solely for additive-error correction, would be a 3-additive-error correcting BCH code, since the first six successive powers of α are roots of C^*. The generator matrix of C^* is given by

$$G^* = \begin{bmatrix} 1 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 1 \end{bmatrix}.$$

The code C is the null space of the matrix H_0 given by

$$H_0 = \begin{bmatrix} 1 & \alpha & \alpha^2 & \alpha^3 & \cdots & \cdots & \alpha^{14} \\ 1 & \alpha^3 & (\alpha^3)^2 & (\alpha^3)^3 & \cdots & \cdots & (\alpha^3)^{14} \end{bmatrix}$$

$$= \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 1 & 1 \\ 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 \end{bmatrix}.$$

We note that H_0 has $r = n - k = \deg [g(x)] = 8$ rows, so that the parity check matrix of C is equal to H_0. The matrix H_1 is given by

$$\begin{aligned} H_1 &= [1, \beta, \beta^2, \cdots, \beta^{13}, \beta^{14}] \\ &= [1, \alpha^5, (\alpha^5)^2, \cdots, (\alpha^5)^{13}, (\alpha^5)^{14}] \\ &= \begin{bmatrix} 1 & 0 & 1 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 0 & 1 & 1 \\ 0 & 1 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}. \end{aligned}$$

We can see from this matrix that, as noted earlier, the order of β is 3, i.e., $\beta^3 = (1, 0, 0, 0)$. Note that the last row of H_1 is null, and the second and third rows are identical so that H_1 is not of full rank. The rank of H_1 is in fact, equal to deg $[g_5(x)] = 2$. To encode a k^*-coordinate information vector $\mathbf{s}' = (s_0, s_1, \cdots, s_{k^*-1})$ into a codeword of C^*, we make the vector $\mathbf{s}'$ correspond to the codeword $\mathbf{v}' = \mathbf{s}'G^*$. [Equivalently, we make the information polynomial

$$s(x) = s_0 + s_1x + s_2x^2 + \cdots + s_{k^*-1}x^{k^*-1}$$

correspond to the code polynomial $s(x)g^*(x)$.] For example, the vector (10110) is encoded into $(10110)G^* = (110010100001110)$. Table 2 contains a list of the 32 possible information vectors, $\mathbf{s}'$, and the corresponding codewords of C^*.

Since $t_l = 1$ and $t_r = 1$, the words of the augmented subcode C_a are obtained by adjoining the initial symbol of each word of C^* to the end of the word, and adjoining the final symbol to the beginning.

We now must determine a choice for the translation vector $\mathbf{c}'$. Suppose that we choose $\mathbf{c}' = \mathbf{g}'$, where $\mathbf{g}' = (100010111000000)$ is the coefficient vector of $g(x)$, considered as a polynomial of degree 14. Then we have

$$\mathbf{c}_a' = (01000101110000001),$$

and we may formally write

$$C_t = C_a + \mathbf{c}_a',$$

since each word of C_t corresponds to the sum of $\mathbf{c}_a'$ and a word of C_a. The words of the code C_t are shown in Table 2. The code C_t is an $(n_t, k_t) = (17, 5)$ code. It is the words of C_t that are sent over the channel.

The code C_t can correct $t = 2$ additive errors (since C is a 2-additive-error-correcting BCH code) and $t_s = 2$ synchronization errors, and may be called a $(t, t_s) = (2, 2)$-error-correcting code.

The error-correction procedure is illustrated below.

Suppose that the source has generated the information vector $\mathbf{s}' = (10110)$. The word in C^* corresponding to $\mathbf{s}' = (10110)$ is $\mathbf{v}' = (110010100001110)$, and the corresponding word in C_t is $\mathbf{v}_t' = (00100000110011100)$. Thus the word $\mathbf{v}_t' = (00100000110011100)$ is sent over the channel. Supopse that a left-shift error of order 1 has occurred, so that if no additive errors occurred, the sequence (01000001100111001) would be received, where we have assumed for

TABLE 2

CODEWORDS OF THE SUBCODE C^* AND THE TRANSLATED AUGMENTED SUBCODE C_t

Information vector $\mathbf{s}'$	Corresponding codeword of C^* $\mathbf{v}' = \mathbf{s}'G^*$	Corresponding codeword of C_t $\mathbf{v}_t' = \mathbf{v}_a' + \mathbf{c}_a'$
10000	111011001010000	00110011100100000
01000	011101100101000	01111110111010001
00100	001110110010100	01011000010101001
00010	000111011001010	01001011000010101
00001	000011101100101	11000010101001011
11000	100110101111000	00001000101110000
10100	110101111000100	00101110000001000
10010	111100010011010	00111101010110100
10001	111000100110101	10110100111101010
01100	010011010111100	01100011011111001
01010	011010111100110	01110000001001101
01001	011110001001101	11111001100011011
00110	001001101011110	01010110100111101
00101	001101011110001	11011111001100011
00011	000100110101111	11001100011011111
11100	101000011101100	00010101001011000
11010	100001110110010	00000110011100100
11001	100101000011101	10001111110111010
10110	110010100001110	00100000110011100
10101	110110010100001	10101001011000010
10011	111111111111111	10111010001111110
01110	010100001110110	01101101101101101
01101	010000111011001	11100100000110011
01011	011001010000111	11111111010001111
00111	001010000111001	11010001111110011
11110	101111000100110	00011011111001100
11101	101011110001001	10010010010010010
11011	100010011010111	10000001000101110
10111	110001001101011	10100111101010110
01111	010111100010011	11101010110100111
11111	101100101000011	10011100100000110
00000	000000000000000	01000101110000001

definiteness that the first symbol in the word following $\mathbf{v}_t'$ was 1. In addition to the synchronization error, however, let us suppose that two additive errors occurred, so that the third and twelfth symbols of (00100000110011100), or the second and eleventh symbols of

(01000001100111001), were complemented. Thus

$$\mathbf{y}_a' = (01000001100111001) + (01000000001000000)$$
$$= (00000001101111001)$$

is the received word. The receiver drops the first and last symbols of $\mathbf{y}_a'$ (i.e., the first t_r and last t_l symbols, with $t_r = t_l = 1$), to obtain

$$\mathbf{y}' = (000000110111100).$$

The receiver then calculates the additive-error syndrome

$$\mathbf{y}'H' = (11010111).$$

Since the additive-error syndrome is nonzero, the receiver interprets that an additive error has occurred, and proceeds to correct it. To do this, the receiver would employ one of the known procedures for correcting additive errors for the BCH code C, using the syndrome (11010111).

Since we have

$$(100000000100000)H' = (11010111),$$

the receiver would reach the conclusion that the additive-error pattern in $\mathbf{y}'$ is

$$\mathbf{e}' = (100000000100000).$$

The receiver then calculates the corrected vector

$$\mathbf{z}' = \mathbf{y}' - \mathbf{e}'$$
$$= (000000110111100) - (100000000100000)$$
$$= (100000110011100).$$

Next it calculates the synchronization-error syndrome

$$\mathbf{z}'H_1' = (0110) = \alpha^5.$$

The receiver must now use this synchronization-error syndrome to determine which synchronization error, if any, has occurred.

To do this it calculates

$$\zeta = \mathbf{c}'H_1' = (1110) = \alpha^{10}.$$

Now

$$\mathbf{z}'H_1'/\zeta = \alpha^5/\alpha^{10} = \beta^2;$$

thus

$$\beta^u = \beta^2 \quad \text{or} \quad u = 2 > 1 = t_r .$$

Hence the receiver interprets that a left-shift error of order $L = n_1 - u = 3 - 2 = 1$ has occurred, and moves the word marks one place to the left.

Hence the truncated received word, corrected for additive and synchronization errors, is

$$\mathbf{z}_c' = (010000011001110).$$

Subtracting $\mathbf{c}'$, the receiver correctly interprets that

$$\mathbf{v}' = \mathbf{z}_c' - \mathbf{c}' = (010000011001110) - (100010111000000)$$
$$= (110010100001110)$$

is the word of C^* corresponding to the transmitted word. Hence the information symbols are correctly interpreted as $\mathbf{s}' = (10110)$.

Because of the seriousness of synchronization errors, the receiver should not take corrective action upon observing the first indication (from the synchronization-error syndrome) that a synchronization error has occurred. For proper use of the technique, the truncated received word must not itself contain the synchronization error. Thus the first word after the word containing the synchronization error gives the first reliable indication of the occurrence of the synchronization error. Words actually containing symbol gains or losses are severely altered and may result in nonzero, but false, synchronization error syndromes. Also, if more than t additive errors occur, a nonzero synchronization-error syndrome may result even though there has been no synchronization error. The importance of making correct decisions regarding synchronization errors warrants the observation by the receiver of the *same* synchronization-error syndrome for several successive words before correcting the apparent synchronization error. Of course, if the receiver destroys synchronization by taking corrective action corresponding to a spurious nonzero synchronization-error syndrome resulting from the occurrence of more than t additive errors, then this mistake will be rectified with the next received word containing not more than t additive errors.

RECEIVED: January 6, 1967; revised April 21, 1967

REFERENCES

BARKER, R. H. (1953), Group synchronizing of binary digital systems. *In* Communication Theory" (W. Jackson, Ed.), pp. 273–287. Butterworths, London, England. (U.S. edition: Academic Press, New York.)

BOSE, R. C., AND RAY-CHAUDHURI, D. K. (1960a), On a class of error correcting binary group codes. *Inform. Control* **3**, 68–79.

BOSE, R. C., AND RAY-CHAUDHURI, D. K. (1960b), Further results on error correcting binary group codes, *Inform. Control* **3**, 279–290.

CALDWELL, J. G. (1966), Synchronizable error-correcting codes, Doctoral dissertation, Department of Statistics, University of North Carolina, Chapel Hill.

HOCQUENGHEM, A. (1959), Codes correcteurs d'erreurs, *Chiffres* **2**, 147–156.

LEVY, J. E. (1966), *Self-Synchronizing Codes Dervied from Binary Cyclic Codes, IEEE Trans. Inform. Theory IT* **12**, 286–290.

PETERSON, W. W. (1961), "Error-Correcting Codes." The M.I.T. Press, Cambridge, Massachusetts.

STIFFLER, J. J. (1965), Comma-free error correcting codes, *IEEE Trans. Inform. Theory IT* **11**, 107–112.

TONG, S. Y. (1966), Synchronization recovery techniques for binary cyclic codes, *Bell Syst. Tech. J.* **45**, 561–596.