Synchronization Recovery Techniques for Binary Cyclic Codes*

By S. Y. TONG

(Manuscript received December 17, 1965)

A class of binary block codes capable of simultaneous correction of additive errors and synchronization errors is presented. This class of codes consists of coset codes of binary cyclic or shortened cyclic codes, and retains the implementation advantages of binary cyclic codes. In most cases, the redundancy required to provide additive-error correction is sufficient to give synchronization-error correction so that no additional redundant bits are required.

Synthesis procedures to construct such codes are also presented, along with an upper bound on the number of synchronization errors which can be corrected by codes in this class.

I. INTRODUCTION

In serial-type data transmission systems, alpha-numeric characters are ordinarily represented by groups of binary symbols. To get meaningful information transfer, it is necessary at the receiver to correctly partition the incoming bit sequence, i.e., to establish and maintain "Character Timing". It is well known that channel noise not only produces additive errors but also can cause timing errors; consequently, methods to correct timing errors have been suggested by many authors. Usually these methods require special coding of the messages, as in comma-free codes,[1,2,3,4,5] or the insertion of synchronization sequences between blocks of messages.[6]

A similar timing problem exists in systems where error control is employed; the problem is transformed from character timing to word synchronization, or, equivalently, the ability to distinguish information bits from check bits. Codes that protect word synchronization as well

* This is a part of the Ph.D. dissertation submitted by the author to the Department of Electrical Engineering, Princeton University, Princeton, N. J.

as correct additive errors have been investigated. Sellers[7] has proposed a scheme where a burst-error-correcting code interlaced with additional check bits is used to give limited protection against synchronization loss, or provide burst-error correction when synchronization is maintained. Stiffler[8] has derived a necessary and sufficient condition for the existence of coset codes such that the sequences produced by slipping the word framing by r bits will not be code words. Such coset codes are useful for systems where coding is used for error detection only.

In order to utilize such a coset code for both additive-error correction and synchronization-error detection it is necessary that synchronization errors not result in decodable sequences. A condition sufficient for this has been obtained by Levy.[9] However, a more useful result would be a condition which would enable *correction* of both types of errors.

In this paper, a technique for obtaining codes capable of correcting synchronization errors as well as additive errors is presented. Furthermore, it is shown that in many cases correction of synchronization errors is possible even in the presence of additive noise. Generally, the scheme requires no additional check bits and the implementation is simple.

In addition to these fundamental results, a set of coset codes, optimal with respect to synchronization error detecting ability is obtained; this represents an improvement of Levy's results.[9]

1.1 *Definitions and Preliminaries*

To correct word-sync loss for an error-correcting code there are two conditions which must be met in order not to reduce the normal error-correcting capability of the code. The first condition is that an error pattern caused by sync loss must not be in any of the cosets which the code utilizes for correction of additive errors. This will ensure that the loss of sync will not be interpreted by the decoder as additive noise, and vice versa. The second condition is that the set of error patterns caused by misframing in one direction must be disjoint from the set of error patterns caused by misframing in the other direction. If the second condition is met, one can proceed to correct the word framing error iteratively. For the detection of sync loss, the first condition is necessary and sufficient, while for the correction of sync loss, the second condition must be satisfied. If the code is used for detection only, the first condition reduces to the requirement that the overlapping of any two code words should not be another code word, so that sync errors can always be detected as an erroneous word, although one would not be able to distinguish sync loss from additive errors.

Let

$$\mathbf{A} = (a_1, a_2, \cdots, a_n)$$
$$\mathbf{B} = (b_1, b_2, \cdots, b_n)$$
$$\mathbf{C} = (c_1, c_2, \cdots, c_n)$$
$$\mathbf{D} = (d_1, d_2, \cdots, d_n)$$

be code words, not necessarily distinct, then:

Definition 1: A synchronization bit loss (or bit loss) of r bits in word framing is said to occur if the receiver bit counter is r bits behind what it should be. That is to say, if the sequence $a_1, \cdots, a_n, b_1, \cdots, b_n$ is framed by the receiver as $a_{n-r+1}, \cdots, a_n b_1, \cdots, b_{n-r}$ where $r < [n/2]$ and the message is taken in such a way that b_n is the first bit of the sequence to arrive at the receiver.

Definition 2: A synchronization bit gain (or bit gain) of r bits in word framing is said to occur if the receiver bit counter counts r bits more than it should. Thus, $a_1, \cdots, a_n, b_1, \cdots, b_n$ is framed by the receiver as $a_{r+1}, \cdots, a_n b_1, \cdots, b_r$ where $r \leqq [n/2]$.

Definition 3: If $(a_{n-i+1}, \cdots, a_n b_1, \cdots, b_{n-i})$ and $(a_{i+1}a_{i+2}, \cdots, a_n b_1 b_2, \cdots, b_i)$ are not code words for every pair, **A**, **B** and all i,

$$0 < i \leqq r,$$

then the code is said to have comma-free freedom r.

Definition 4: A correctable coset of a code is defined as a coset whose leader is one of the error patterns the decoder corrects.

Definition 5: If the sequences $a_{i+1}a_{i+2}, \cdots, a_n b_1 b_2, \cdots, b_i$ and $a_{n-i+1}, \cdots, a_n b_1, \cdots, b_{n-i}$ do not belong to any of the correctable cosets of the code for every pair **A**, **B** and all i, $0 < i \leqq r$, then the code is said to have sync-detection capability r.

Definition 6: A code is said to have sync-recovery capability r if the code has sync-detection capability $q \geqq r$ and if the cosets containing $a_{i+1}, \cdots, a_n b_1, \cdots, b_i$ are disjoint from the cosets containing $c_{n-j+1}, \cdots, c_n d_1, \cdots, d_{n-j}$ for all $0 < i, j \leqq r$ and for every set of **A**, **B**, **C**, **D** of the code.

Definition 7: A code is said to have guaranteed noise tolerance of (k,r) bits, if the code is guaranteed to correct r bits of sync slippage with k or fewer additional bit errors in the received block.

Definition 8: A code is said to have expected noise tolerance of $E(r)$ bits, if with a probability of at least $\frac{1}{2}$, the code can correct r bits of sync slippage with $E(r)$ or fewer additional bit errors in the received block.

In the subsequent discussion all the error-correcting codes are assumed to correct random errors.

1.2 *Summary of the Results*

Consider a binary cyclic code which corrects $t \geqq 2r + 1$ errors, $r > 0$. If such a code is shortened by $2r + 1$ bits or more, it is shown in Section II that the code can be made to have sync-recovery capability r with expected noise tolerance of $E(\beta)$ bits when a slippage of β bits occurs

$$E(\beta) = 2^{-2\beta} \sum_{i=1}^{2\beta} (t - i) \binom{2\beta}{i - 1}, \qquad 0 < \beta \leqq r$$

without added redundancy. A similar technique is developed for codes which correct more than one error. It is shown that by adding $2r$ zeros to each code word and shortening the code by $2r + 1$ bits or more that the code can be made to have sync-recovery capability of r bits.

A scheme which is applicable to a single error-correcting code is also developed. It is shown that for any error-correcting code without an even-parity check it is possible to have sync-recovery capability of one bit if two information bits of the code are replaced by two zeros. In Section III, techniques are developed for binary cyclic codes which are not shortened. A necessary and sufficient condition is derived for the existence of a coset code having specified sync-correcting ability. It is also shown that a cyclic code which corrects t errors and has minimum distance d_m can be made to have sync-recovery capability

$$r \leqq d_m - 2t - 2$$

without additional redundancy and an optimal set of codes that assures $r = d_m - 2t - 2$ is given.

For cyclic codes with some special properties, it is shown that a synthesis procedure can be used to construct coset codes of the given code so that one bit out-of-sync can always be corrected. This procedure applies to almost all of the Bose-Chaudhuri[10] Hocquenghem[11] codes that corrects more than two errors.

Bounds on the amount of slippage which can be corrected are derived. It is shown that sync-recovery capability for any binary cyclic code cannot exceed $[(n - k - 1)/2]$* bits and sync-detection capability cannot

* $[x]$ denotes the greatest integer less than or equal to x.

exceed $n - k - 1$ bits, where n is the block length of the code and k is the number of information bits in the code. A brief discussion on the sync-detection capability of error-detecting codes is included in Section IV.

1.3 *Properties of Cyclic Codes*

A subspace V of l-tuples is called a cyclic code if for each vector $v = (a_0, a_1, \cdots, a_{l-1})$ in V, the vector $v' = (a_{l-1}, a_0, \cdots, a_{l-2})$ is also in V.

By considering each l-tuple as an element of the algebra A_l of polynomials modulo $x^l = 1$, one may associate each l-tuple $(a_0, \cdots, a_{l-1})$ with a polynomial $f(x) = a_0 + a_1x, \cdots, a_{l-1}x^{l-1}$ in the residue class modulo $x^l - 1$.* It can be shown that a subspace is a cyclic code if and only if it is an ideal in the algebra of polynomials modulo $x^l - 1$.[12] The generator $g(x)$ of the ideal is known as the generator polynomial of the cyclic code. It follows that l, which represents the natural length of the code, must be the least common multiple of the roots of the generator polynomial of the cyclic code. Given an (l,k) cyclic code, it is always possible to form an $(l - i,k - i)$ code by making the i leading information bits identically zero and omitting them from all code vectors. Such a code is no longer cyclic, and is called a shortened cyclic code. Denote the code space of a cyclic code by C_0 and the code space of a shortened cyclic code by C_i, where i is the number of bits shortened. Let n be the block length of a shortened cyclic code (i.e., $n = l - i$), the higher order $l - n$ bits are identically zero and hence are not transmitted. We can imagine that the receiver will decode the shortened code by first augmenting the received n-bit code word by $l - n$ zeros and then decoding it as if it were a full-length cyclic code. (Note that the actual receiver need not perform these precise functions, but in any case it has to do something equivalent to this.)

Now let us investigate what will happen if an r-bit loss occurs as shown in Fig. 1. Given that the transmitted message is $R(x)$, the received message is

$$x^rR(x) + A(x) + x^nB(x) \tag{1}$$

where $A(x)$ is the portion of next word entered into the framing and $B(x)$ is the higher order r-bit portion of $R(x)$ out of framing. In general,

* This paper discusses codes over binary field only so that the coefficients of polynomials are either 0 or 1 and + or − signs are used interchangeably.

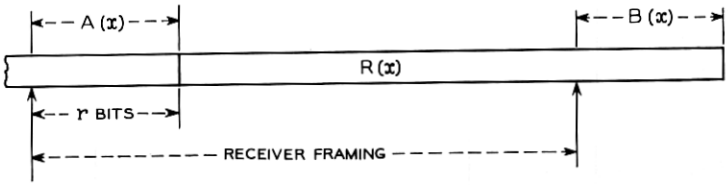

Fig. 1 — The situation of synchronization loss.

$A(x)$ and $B(x)$ are not predictable and one may write the received message in the following form

$$x^r R(x) + \delta_1^r(x) + x^n \delta_2^r(x) \tag{2}$$

where $\delta_i^r(x)$ represents a polynomial of degree at most $r - 1$ with random coefficients. For cyclic codes, $n = l$ and $x^l = 1$. Therefore,

$$\delta_1^r(x) + x^l \delta_2^r(x) = \delta_1^r(x) + \delta_2^r(x) = \delta^r(x), \tag{3}$$

so that the received message is

$$x^r R(x) + \delta^r(x). \tag{4}$$

Since $x^r R(x)$ is an element of C_0, it is divisible by $g(x)$ so that its syndrome is 0. The coset leader of the coset to which $\delta^r(x)$ belongs must have a weight of no more than r.* For the same reason, with a shortened cyclic code the coset leader of the coset of $\delta'(x) = [\delta_1^r(x) + x^n \delta_2^r(x)]$ must have a weight of not more than $2r$. It can be shown that a similar result holds for an r-bit gain. It follows that:

Theorem 1: A slippage of r bits in synchronization can result in a vector at most distance 2r from a nearest code vector if the code is a shortened cyclic code and at most distance r from a nearest code vector if the code is cyclic, where code vectors are elements of C_0.

II. SCHEME FOR SHORTENED CYCLIC CODES

2.1 *Codes that Correct at Least Three Random Errors*

Suppose a fixed polynomial, $P(x)$ is added to every code word at the encoder and the same polynomial is subtracted from the received message at the decoder. If the sync is maintained properly, the effect of $P(x)$ will be canceled. However, if sync is not maintained, an error

* We assume that the coset leader is the minimum weight element of the coset, which is the desired case for a random-error correcting code.

pattern may be detected. By choosing $P(x)$ in a suitable way, it is conceivable that one may be able to detect or even correct sync slips.

Let the code word be $R(x) \epsilon C_i = C_{l-n}$, where $x^l = 1$ and n is the length of shortened code word. The transmitted word is $R(x) + P(x)$. At the receiver, assume the word framing has an r-bit loss as shown in Fig. 1. The received word, $Q(x)$, according to (2), takes the following form:

$$Q(x) = x^r[R(x) + P(x)] + \delta_1{}^r(x) + x^n\delta_2{}^r(x). \tag{5}$$

The receiver then subtracts $P(x)$ from $Q(x)$, i.e.,

$$Q_1(x) = Q(x) + P(x). \tag{6}$$

The syndrome of $Q_1(x)$ is the remainder of $Q_1(x)/g(x)$, i.e.,

$$\begin{aligned} \{Q_1(x)\} &= \{Q(x) + P(x)\} \\ &= \{(1 + x^r)P(x) + \delta_1{}^r(x) + x^n\delta_2{}^r(x)\} \end{aligned} \tag{7}$$

where $\{x\}$ represents either the residue class of x modulo $g(x)$ or the polynomial of at least degree in that class; the context should make it clear which is meant. If $P(x)$ is chosen so that the coset $\{(1 + x^r)P(x)\}$ coincides with $\{x^i + \theta_1{}^r(x) + x^n\theta_2{}^r(x)\}$, where $n \leqq i < l$, and $\theta_j{}^r(x)$ is a polynomial of degree at most $r - 1$, for $j = 1, 2$, then,

$$\begin{aligned} &\{(1 + x^r)P(x) + \delta_1{}^r(x) + x^n\delta_2{}^r(x)\} \\ &\qquad = \{x^i + \theta_1{}^r(x) + x^n\theta_2{}^r(x) + \delta_1{}^r(x) + x^n\delta_2{}^r(x)\} \\ &\qquad = \{x^i + \delta_3{}^r(x) + x^n\delta_4{}^r(x)\}\ . \end{aligned} \tag{8}$$

Note that $x^i + \delta_3{}^r(x) + x^n\delta_4{}^r(x)$ has at most $2r + 1$ nonzero terms. Thus, if the code C_0 corrects $t \geqq 2r + 1$ errors, the polynomial

$$x^i + \delta_3{}^r(x) + x^n\delta_4{}^r(x)$$

must be a coset leader of C_0 for all possible $\delta^r(x)$'s. Thus, if $n + r \leqq i$, x^i cannot be canceled by the $\delta^r(x)$'s and the decoder will indicate that the $i + 1$ position of the received word is in error. But $n \leqq i < l$, so the ith $+ 1$ bit was not transmitted; this can be used to indicate that a misframing has occurred.

Similarly, it can be shown that if the receiver has an r-bit gain, then the message to the decoder is

$$Q_2(x) = x^{-r}[R(x) + P(x)] + x^{-r}\delta_1{}^r(x) + x^{n-r}\delta_2{}^r(x) + P(x) \tag{9}$$

$$\begin{aligned} \{Q_2(x)\} &= \{x^{i-r} + x^{-r}\delta_3{}^r(x) + x^{n-r}\delta_4{}^r(x)\} \\ &= \{x^{-r}Q_1(x)\} \qquad \text{for some } Q_1(x). \end{aligned} \tag{10}$$

Since each possible $Q_1(x)$ is a coset leader for all $\delta^r(x)$, so is $x^{-r}Q_1(x)$. Thus, by reasoning similar to that employed in the bit loss case, an r-bit gain can be detected.

In order to recover synchronization, one must be able to distinguish the syndrome due to bit loss from the syndrome due to bit gain. This implies:

$$\{x^i + \delta_3^r(x) + x^n\delta_4^r(x)\} \neq \{x^{i-r} + x^{-r}\delta_3^r(x) + x^{n-r}\delta_4^r(x)\} . \quad (11)$$

That is, the error patterns must not be in the same coset.

$$\{x^i + x^{i-r} + \delta_3^r(x) + x^n\delta_4^r(x) + x^{-r}\delta_3^r(x) + x^{n-r}\delta_4^r(x)\} \neq 0. \quad (12)$$

The weight of the polynomial within the bracket is at most $4r + 2$. Since the code is $t \geqq 2r + 1$ error correcting, any code word must have a weight of at least $4r + 3$. Hence, the inequality is always satisfied provided that either x^i or x^{i-r} is not canceled by the terms of the $\delta_i(x)$ polynomials. Otherwise, it would be possible for all other terms of $\delta_i(x)$ to be zero, resulting in a zero polynomial and thereby contradicting (12).

It is clear that in order to do this it is necessary and sufficient to have either x^i or x^{i-r} or both not in those positions where the δ's may take the value of 1's; that is to say, either

$$n + r - 1 < i < l - r \quad (13)$$

or

$$n + r - 1 < i - r < l - r \quad (14)$$

must be satisfied. Both conditions require $l - n \geqq 2r + 1$ which is the minimum number of bits required to be eliminated.

Thus, we have shown that if: (*i*) a cyclic code corrects $t \geqq 2r + 1$ errors; (*ii*) the number of eliminated bits is at least $2r + 1$; and (*iii*) it is possible to find a polynomial $P(x)$ constrained by (8); then,upon an r-bit loss or gain, the syndrome generated will be different from that of any of the correctable cosets and every bit loss syndrome will be different from any of the syndromes caused by bit gain, and vice versa. To complete the analysis, one must show the existence of polynomials $P(x)$ constrained by (8).

Recall that the requirement on $P(x)$ is that

$$\{(1 + x^r)P(x)\} = \{x^i + \theta_1^r(x) + x^n\theta_2^r(x)\} . \quad (15)$$

Since the θ's are arbitrary polynomials, [as they will be combined with δ's, see (8)] we may treat them as variables in determining the simplest possible $P(x)$. In particular, if one elects to satisfy condition (13), i.e.

$$n + r - 1 < i < l - r$$

and to minimize the number of bits to be eliminated, the smallest i should be selected. Therefore, let $i = n + r$. For reasons which will become apparent later,* we set $\theta_1{}^r(x) = 0$ and $\theta_2{}^r(x) = 1$. Then

$$\{(1 + x^r)P(x)\} = \{x^{n+r} + x^n\}$$

which implies

$$\{P(x)\} = \{x^n\}. \tag{16}$$

Thus, we have constructed a $P(x)$ which satisfies (15). This completes the derivation of $P(x)$ for the detection and correction of an r-bit gain or loss. To see if this pattern is also good for any β-bit sync slippage ($0 < \beta \leqq r$), we note that the error pattern for a β-bit loss is

$$\begin{aligned}(1 + x^\beta)P(x) + \delta_1{}^\beta(x) + x^n\delta_2{}^\beta(x) \\ &= x^n + x^{n+\beta} + \delta_1{}^\beta(x) + x^n\delta_2{}^\beta(x) \\ &= x^{n+\beta} + \delta_1{}^\beta(x) + x^n\delta_3{}^\beta(x).\end{aligned} \tag{17}$$

The number of nonzero terms of the polynomial is at most

$$2\beta + 1 < 2r + 1 \leqq t, \qquad \text{for all} \qquad \beta < r,$$

hence, identification of an error in position $n + \beta + 1$ is always possible. Since $n < n + \beta + 1 < l$, and position $n + \beta + 1$ was not transmitted, the fact that the decoder will show the $n + \beta + 1$ position to be in error can be used to indicate that an out-of-sync situation exists. By a similar argument, it can be shown that the detection of any β-bit gain is also possible for $\beta < r$.

In order to recover synchronization, the set of syndromes corresponding to bit loss must be different from those corresponding to bit gain; that is,

$$\begin{aligned}\{x^{n+\alpha} + \delta_1{}^\alpha(x) + x^n\delta_3{}^\alpha(x)\} \\ &\neq \{x^n + x^{-\beta}\delta_1{}^\beta(x) + x^{n-\beta}\delta_3{}^\beta(x)\}\end{aligned} \tag{18}$$

for all

$$0 < \alpha, \beta \leqq r$$

or

$$\{x^{n+\alpha} + x^n + \delta_1{}^\alpha(x) + x^n\delta_3{}^\alpha(x) + x^{-\beta}\delta_1{}^\beta(x) + x^{n-\beta}\delta_3{}^\beta(x)\} \neq 0. \tag{19}$$

* To maximize expected noise tolerance, it is desirable to keep the degree of $\theta^r(x)$ small.

The number of nonzero terms within the brackets (19) is at most

$$2\alpha + 2\beta + 2 < 4r + 3 < 2t + 1 \leqq d_m$$

where d_m is the minimum distance of the code. Equation (19) cannot represent a code word unless both $x^{n+\alpha}$ and x^n are canceled by some terms of the δ's in such a way that all an-zero polynomial results. It is seen from (19) that $n + \alpha < l - \beta$ is a sufficient condition for $x^{n+\alpha}$ not to be canceled by any of the δ's. Since $l - n \geqq 2r + 1 > \alpha + \beta$, this is always satisfied. Thus, we have shown that a shortened cyclic code which corrects t random errors will have sync-recovery capability of r bits, if $l - n \geqq 2r + 1$, $t \geqq 2r + 1$, and if the *syndrome* of x^n is added to each word after encoding and before decoding.

The implementation of this scheme is obviously easy. The generation of $\{x^n\}$ can be accomplished by adding one bit corresponding to position x^n at encoder and decoder. (Since this position is not actually transmitted over the channel, only the syndrome of x^n is added to the encoded word.) Usually only a slight loss in efficiency results from the shortening of the code.

The decision rule can be formed as follows:

(*i*) If the decoder indicates that the bit corresponding to x^n is in error, but not any x^k, $n < k \leqq l - 1$, then one assumes the system has gained a few bits in bit count. By extending the sync count one bit at a time, the system will regain sync in at most r word times.

(*ii*) If the decoder indicates that the bit corresponding to $x^{n+\beta}$ is in error, $1 \leqq \beta \leqq r$, then one assumes the system has lost β bits. The word sync can be recovered either by using step-by-step correction as in (*i*), or by a one step correction.

Note that the $P(x)$ derived here is but one of many possibilities; for example, by letting $\{P(x)\} = \{x^{l-1}\}$ one would come up with a similar decision rule which favors bit-gain correction rather than the bit-loss correction as we have done.

Example: Consider the (23,12) Golay code shortened to a (20,9) code. Since it is a triple error-correcting code and since $l - n = 3$, it should have sync-recovery capability of 1 bit. Let us demonstrate this fact by step-by-step computation. The generator polynomial for the code is:

$$g(x) = 1 + x^2 + x^4 + x^5 + x^6 + x^{10} + x^{11}$$

$$= 101011100011.$$

The syndrome of $x^n = x^{20}$ is 00101101111 $= \{P(x)\}$ and of $x^{n+1} = x^{21}$ is 10111000110. Assume the information 000000001 is to be transmitted. After encoding, the code word is

$$\begin{array}{rl} & 1{,}01011011110000000001{,}01 \\ \text{Add } P(x) & \underline{\quad 00101101111000000000} \\ & 1{,}01110110001000000001{,}01. \end{array}$$

The above sequence is the transmitted message.

(*i*) Assume one bit loss has occurred. Thus, the received message is

$$\begin{array}{l} 1{,}01110110001000000001{,}01. \qquad \text{message flow} \longrightarrow \\ \uparrow \qquad\qquad\qquad\qquad \uparrow \\ \text{Receiver Frame} \end{array}$$

To this word the receiver adds $P(x)$ and three 0's

$$\begin{array}{rl} & 10111011000100000000 \\ & \underline{00101101111000000000} \\ A(x) = & 10010110111100000000(000) \end{array}$$

at the high-order end, as shown in the parentheses, to make a cyclic code. The syndrome of the message, $A(x)$, is the remainder of $A(x)/g(x)$.

$$\begin{aligned} \{A(x)/g(x)\} &= 00111000110 \\ &= 10111000110 + 100000000 \\ &= \{x^{21}\} + \{x^0\}. \end{aligned}$$

The decoder will indicate that bits corresponding to x^{21} and x^0 are in error, but x^{21} was not transmitted and $x^{21} = x^{20+1} = x^{n+\beta}$. By the decision rule just derived, one decides that a one bit loss has occurred.

(*ii*) Suppose a one bit gain has occurred. Thus, the received message is

$$\begin{array}{l} 1{,}01110110001000000001{,}01; \\ \uparrow \qquad\qquad\qquad\qquad \uparrow \end{array}$$

add $P(x)$ and three missing zeros. We have

$$\begin{array}{rl} & 11101100010000000010 \\ P(x) = & \underline{00101101111000000000} \\ & 11000001101000000010(000). \end{array}$$

The syndrome is

$$01110110001 = 00101101111 + 01011011110 = x^{20} + x^{19}.$$

By the decoding rule we see that $x^n = x^{20}$ is in error but not x^k for all k such that $20 = n < k \leqq l - 1 = 22$, hence the decoder decides that a bit gain has occurred. Notice that in this case, only two errors are indicated by the decoder due to a misframing of one bit. Because the code is triple error-correcting, sync recovery of a one bit of misframe is possible even if there is an error due to additive noise. This feature of noise tolerance will be discussed in the following paragraph.

2.1.1 *Noise Tolerance*

Assume a slippage of $\beta \leqq r$ bits has occurred. According to (17) the error pattern is

$$x^{n+\beta} + \delta_1^{\beta}(x) + x^n\delta_3^{\beta}(x).$$

The weight of this error polynomial is at most $2\beta + 1$, but the code corrects t errors, so that at least $t - 2\beta - 1$ additional errors can be in the received block without disturbing the sync-correcting process. This is so because every error-bit position can be identified correctly provided the total number of errors does not exceed t; hence the guaranteed noise tolerance is $(t - 2\beta - 1,\beta)$.

Since channel noise cannot affect any of the bit positions $n < i \leqq l$ directly, the vital bits for detection and correction of sync are not likely to be corrupted by noise. In fact, it can be shown that at least $2r + 2$ additional errors are required at specific locations to change the bit in position i for $n < i \leqq l$.

If we assume that the probability of occurrence of any nonzero term of the δ's is $1/2$, we can compute the expected noise tolerance $E(\beta)$ as follows:

If β bits of slippage occurs, there is at least one error caused by the error pattern purposely generated by $P(x)$, but not more than $2\beta + 1$ errors, in accordance with (17). The probability of occurrence of a total of i errors due to sync slippage of β bits is

$$P_i = \binom{2\beta}{i-1} \Big/ 2^{2\beta}; \tag{20}$$

the number of additional errors which can be tolerated with i errors is $t - i$, so the expectation is

$$E(\beta) = 2^{-2\beta} \sum_{i=1}^{2\beta+1} (t - i) \binom{2\beta}{i-1}. \tag{21}$$

That is to say, on the average, for a β-bit sync slippage, one can tolerate $E(\beta)$ additional errors. For example, consider a (255,215) BCH code

which corrects five random errors. Since $t \geqq 2r + 1$, $r_{\max} = 2$, $l - n \geqq 2r + 1 = 5$, $l = 255$, $n = 250$; therefore, the (250,210) shortened cyclic code can have sync-recovery capability up to 2 bits per word.

The guaranteed noise tolerances are $(t - 2\beta - 1,\beta) = (2,1)$ and $(0,2)$, and the expected noise tolerances are

$$E(1) = 2^{-2} \sum_{i=1}^{2} (5 - i) \binom{2}{i - 1} = 3$$

$$E(2) = 2^{-4} \sum_{i=1}^{4} (5 - i) \binom{4}{i - 1} = 2.$$

2.2 *Double Error-Correcting Codes*

The scheme just proposed cannot be used for double error-correcting codes because, in the worst case, for only one bit of sync slippage, it is possible to have three errors generated as a result of sync loss. (However, the probability of recovering sync loss is still high.) In this section the scheme is modified so that it is guaranteed to correct sync loss for double error-correcting codes; however, extra redundancy is required.

Let m zeros be placed on each end of a coded message of an n-bit shortened cyclic code. The transmitted word is then a stream of binary messages interlaced with $2m$-zero bits between messages (i.e., the added zeros are actually transmitted). A typical word is of the form shown below

$x^0 x^1 \cdots x^{m-1}$	$x^m \cdots x^{m+n-1}$	$x^{m+n} \cdots x^{2m+n-1}$
all zero	$x^m R(x)$	all zero

where the first and the last m bits are identically zero and the center portion is the shortened code word, $R(x)$.

Let $x^m P(x)$ be added to such a code; the transmitted message is

$$x^m[R(x) + P(x)]. \tag{22}$$

It is clear that the word framing at the receiver must contain $2m + n$ bits. It follows that all terms of $x^{r+m}[R(x) + P(x)]$ will be within a single receiver frame for all $| r | \leqq m$. Thus, for an r-bit loss the received message, in the absence of noise, is

$$x^{r+m}[R(x) + P(x)].$$

After subtracting $x^m P(x)$ at receiver, the syndrome of the resultant message is

$$\{x^{r+m}[R(x) + P(x)] + x^m P(x)\} = \{x^m(1 + x^r)P(x)\}, \quad |r| \leqq m. \tag{23}$$

Let $P(x) = \{x^{l-m-1}\}$, then the syndrome due to an r-bit loss is

$$\{x^m(1 + x^r)P(x)\} = \{x^{l-1} + x^{l+r-1}\}. \tag{24}$$

If $2m + n \leqq l - 1$, the bit corresponding to x^{l-1} (which is the lth bit) is not transmitted. Hence, the detection of x^{l-1} in error can be used to indicate a bit loss. In order to correct the bit loss, note that the error at x^{l+r-1} corresponds to the bit at position $l + r$ modulo l. Therefore, if $r > 0$, the error bit position corresponds to one of the first m bits which is known to be zero so that the inconsistency between the calculated error bit and the actual bit value at position r can serve as an indication of an r-bit loss.

Similarly, for $r < 0$ the bit corresponding to the syndrome $\{x^{l+r-1}\}$ is $l + r \geqq 2m + n + r + 1 \geqq m + n + 1$ for $0 > r \geqq -m$ but the bits $m + n + 1, m + n + 2, \cdots, 2m + n$ are known to be zero, so that the r-bit gain can be detected in the same way. Notice that if $l + r \geqq 2m + n$, for some $r < 0$, the error indication directly shows an $|r|$-bit gain has occurred since the bit $l + r$ is not transmitted.

Recall that one requires

$$2m + n \leqq l - 1, \tag{25}$$

i.e.,

$$l - n \leqq 2m + 1 \tag{26}$$

so that at least $2m + 1$ bits of the information symbols must be eliminated.

Since exactly two errors (namely x^{l-1} and x^{l+r-1}) will be generated for every sync loss up to m bits, a code capable of correcting at least two errors must be used. It is obvious that for a t-error-correcting code, an addition of $t - 2$ errors anywhere in the data section (i.e., in the $x^m R(x)$ part) can be tolerated. However, errors in the zero section can affect the sync-correcting process, although the sync-detection capability will not suffer. Thus, the guaranteed noise tolerance for sync-recovery is zero while it is $t - 2$ for sync detection.

We summarize the sync-correcting rule as follows:

(*i*) If x^{l-1} is in error then one assumes that a sync-slip has occurred.

(*ii*) If, in addition, one and only one of the calculated error bits is in the bit position corresponding to x^{l+r-1}, $|r| \leqq m$, while the actual bit of that position is either not transmitted or has the value of zero then

one assumes an r-bit loss has occurred (if $r < 0$, $| r |$-bit loss = r-bit gain).

It is easy to see from the above argument that any $P(x) = \{x^{l-i}\}$, $1 \leqq i \leqq l - 2m - n$, can be used for this scheme.

Example: Consider a (15,7) BCH code shortened to (12,4) code with one zero adjoined at each end of the code to obtain (14,4) code. Here $l = 15$, $m = 1$, $n = 12$. The code is generated by

$$g(x) = 1 + x^4 + x^6 + x^7 + x^8.$$

Assume the message 1100 is to be sent. After encoding we have the (12,4) code word

$$\begin{aligned} R(x) &= 010001011100 \\ P(x) &= \{x^{l-m-1}\} = \{x^{13}\} = 001011100000 \\ R(x) + P(x) &= 011010111100. \end{aligned}$$

Add a zero at each end, the transmitted message is

$$0,001101011110000,0.$$

Notice that the neighboring bits of the message must be zero since each message must begin and end with zero, by construction.

(i) At the receiver assume a one-bit loss has occurred. Then we have the message, as the receiver sees it:

$$\begin{array}{c} 0,00110101111000,0. \\ \uparrow \qquad\qquad\quad \uparrow \end{array}$$

The receiver adds $xP(x)$ and the resulting message becomes

$$\begin{array}{rl} & 00011010111100 \\ \text{add } xP(x) = & 00010111000000 \\ \hline & 00001101111100(0). \end{array}$$

The syndrome is $10010111 = 00010111 + 10000000 = \{x^{14}\} + \{x^0\}$ which indicates x^{14} and x^0 are in error but x^{14} is never transmitted and x^0 is known to have been transmitted as zero. Therefore, according to the decoding rule just derived, we see that word sync has slipped. Since $x^{l+r-1} = x^{14+r} = x^0$, $r = 1$, which shows a gain of one bit has occurred.

(ii) Assume one-bit gain has occurred, the received message becomes as shown

$$\begin{array}{rl} & 01101011110000 \\ \text{add } P(x) = & 00010111000000 \\ \hline & 01111100110000(0). \end{array}$$

The syndrome is $00111001 = 00010111 + 00101110 = \{x^{14}\} + \{x^{13}\}$. x^{14} in error indicates a sync-slip condition and $x^{13} = 0$

$$x^{l+r-1} = x^{13} = x^{15-1-1},$$

so $r = -1$ which shows that a gain of one bit in word sync has occurred.

2.3 *Codes Without an Even-Parity Check*

Recall that, from (23), the syndrome is $\{x^m(1 + x^r)P(x)\}$ for an r-bit loss. It is desirable to have such a syndrome coincide with the syndrome of a single bit, say x^i *, and if the bit corresponding to x^i is either not transmitted, or if the value of the coefficient of x^i is known to the receiver by prearrangement, then it is possible to have sync recovery capability for a single error-correcting code. We shall show in the following that such a possibility does exist.

Assertion: For a cyclic error-correcting code without an even-parity check† it is always possible to modify the code in such a way as to have sync-recovery capability of one bit. The scheme is based on the following theorem.

Theorem 2: If $g(x)$ is a generator polynomial for a cyclic code of natural length l and if

$$\text{GCD } (g(x),1 + x) = 1$$

then

$$(1 + x) \mid \{x^{l-1}\}.$$

Proof:

$$(1 + x) \nmid g(x) \Rightarrow g(1) \neq 0$$

$$\therefore g(1) = 1 \quad \text{or} \quad 1 + g(1) = 0$$

or $1 + g(x)$ is divisible by $1 + x$.
Likewise $x \nmid g(x)$ because $g(x)h(x) = x^l + 1$ and if $x \mid g(x)$ then $x \mid x^l + 1$ which is impossible.

Therefore,

$$x \mid [1 + g(x)]. \tag{27}$$

It follows that $1 + g(x) = x(1 + x)F_3(x)$. In the ring of polynomials modulo $x^l + 1$

$$x^l \equiv 1$$

* i.e., the remainder of $x^i/g(x)$.
† Or, equivalently, the generator polynomial of the code is not divisible by $1 + x$.

or

$$x^{l-1} \equiv x^{-1},$$

by (27),

$$x^{-1} \equiv x^{-1}(1 + g(x)) \equiv (1 + x)F_3(x) \text{ [mod } g(x)],$$

i.e.,

$$\{x^{l-1}\} = (1 + x)F_3(x).$$

Q.E.D.

Theorem 2 shows the existence of $P(x) = \{x^{l-1}/1 + x\}$ for such codes. Thus, for any cyclic code which corrects one or more errors and whose generator polynomial, $g(x)$, is not divisible by $1 + x$, one may shorten the code by 2 bits, append one zero at each end of the encoded message and add the pattern $xP(x) = x\{x^{l-1}/1 + x\}$. The configuration is shown in Fig. 2. Note that the added zeros are actually transmitted.

When the system is in synchronization, the framing of the receiver is as shown in Fig. 2. The receiver first adds $xP(x)$ and then decodes the whole word.

From (23), one-bit loss gives the syndrome

$$\{x(1 + x)P(x)\} = \{x^{l-1}x\} = \{x^0\}$$

and one-bit gain gives the syndrome $\{x^{-1}(1 + x)xP(x)\} = \{x^{l-1}\}$. Note that both x^0 and x^{l-1} are inserted zero bits; hence, the decoder can use the decision rule as shown in Table I.

Example: Consider the single error-correcting Hamming code generated by $x^4 + x + 1$. For this code $l = 15$. From Table II:

$$P(x) = \left\{\frac{x^{l-1}}{1 + x}\right\} = \{x^{10}\}.$$

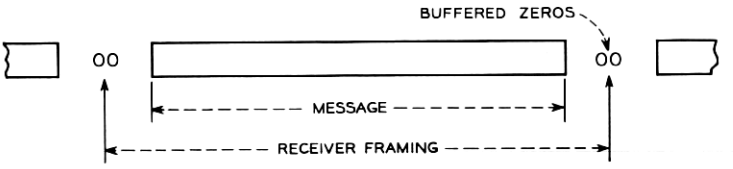

Fig. 2 — The relation between message and word framing.

TABLE I

Error Locator Indicates	Real Bit Value	Decision
x^0 in error	$x^0 = 0$	one-bit loss
x^0 in error	$x^0 = 1$	bit x^0 in error
x^{l-1} in error	$x^{l-1} = 0$	one-bit gain
x^{l-1} in error	$x^{l-1} = 1$	bit x^{l-1} in error

The code is modified from (15,11) to (15,9). Let the information bits be 001000101. After encoding it becomes 000100010001010.

Adding $xP(x) = x^{11}$, we have

$$\begin{array}{rl} xP(x) = & 000000000001000 \\ & \underline{000100010001010} \\ & 000100010000010. \end{array}$$

Case 1: One-bit loss:

The received message becomes

$$\begin{array}{rl} & 00001000100000100 \\ xP(x) = & \underline{00000000000100000} \\ & 00001000100100100. \end{array}$$

The syndrome is 1000; it indicates the first bit in error since the first bit is 0. From Table I we read "one-bit loss has occurred."

Case 2: One-bit gain:

The received message becomes

$$\begin{array}{rl} & 0010001000001000 \\ xP(x) = & \underline{0000000000010000} \\ & 0010001000011000. \end{array}$$

The syndrome is 1001; it indicates x^{14} in error but $x^{14} = 0$. So again by Table I we have detected one-bit gain.

TABLE II—GF(2^4)

x^0	1000	x^8	1010
x^1	0100	x^9	0101
x^2	0010	x^{10}	1110
x^3	0001	x^{11}	0111
x^4	1100	x^{12}	1111
x^5	0110	x^{13}	1011
x^6	0011	x^{14}	1001
x^7	1101	$x^{15} = x^0$	= 1000

2.3.1 *Noise immunity*

It is obvious that this scheme uses only one bit to detect and correct sync loss so that for codes which correct t errors we can afford $t - 1$ more additive errors in the code word provided that the bits x^0 and x^{l-1} are not in error.

III. FULL-LENGTH CYCLIC CODES

For cyclic codes, the technique used to distinguish synchronization loss from additive errors for shortened codes no longer applies. One must select $P(x)$ in such a way that over the range of synchronization slippage r, the error patterns so generated do not fall into any of the correctable cosets. This means

$$\delta^s(x) + P(x)(1 + x^s) \notin \mathcal{C} \tag{28}$$

and

$$x^{n-s}\delta^s(x) + P(x)(1 + x^{n-s}) \notin \mathcal{C}, \tag{29}$$

for all $\delta^s(x)$, $0 < s \leqq r$, where $\mathcal{C}$ is the set of all polynomials that belong to the union of all correctable cosets. By the following lemma it will be shown that conditions (28) and (29) are equivalent.

Lemma 1: $Q(x) \in \mathcal{C}$
if and only if

$$x^i Q(x) \in \mathcal{C} \qquad \textit{for all } i.$$

Proof: (*i*) Suppose $Q(x) \in \mathcal{C}$. Denote the coset leader to which $Q(x)$ belongs by $Q_c(x)$; then $Q(x) + Q_c(x) = w_Q(x)$ is a code word, but $x^i w_Q(x)$ is also a code word and the weight of $Q_c(x)$ is the same as the weight of $x^i Q_c(x)$; the weight of $Q_c(x)$ is no more than t, the maximum number of errors the code corrects. Therefore, $x^i Q_c(x)$ must also be a coset leader. It follows that $x^i Q_c(x) + x^i w_Q(x) = x^i Q(x)$ must belong to the coset whose leader is $x^i Q_c(x)$. That is to say $x^i Q(x) \in \mathcal{C}$.

(*ii*) $x^i Q(x) \in \mathcal{C} \Rightarrow x^{n-i}(x^i Q(x)) = Q(x) \in \mathcal{C}$ by (*i*). Q.E.D.

Since (29) can be rewritten as:

$$x^{n-s}[\delta^s(x) + P(x)(1 + x^s)] \notin \mathcal{C}, \tag{30}$$

by the above lemma and the law of contraposition, (28) and (29) are equivalent. The condition (28) can be restated as:

$$\bar{W}[Q_s(x)] \geqq t + 1 \tag{31}$$

where $Q_s(x)$ is the coset leader of the polynomial

$$\delta^s(x) + P(x)(1 + x^s)$$

and $\bar{W}[x]$ is defined as the weight of the polynomial x.

Equation (31) is a necessary and sufficient condition for a code to have sync-detection capability r. It is sufficient because if (31) is true, then no error pattern generated by $P(x)$ due to slippage of s bits,

$$0 < s \leqq r,$$

can be in a correctable coset; thus sync loss can be detected. It is necessary because otherwise there must exist at least one particular δ^s, say $\delta_\alpha{}^s$, such that (31) is not satisfied, then say

$$\bar{W}[Q_{s_\alpha}(x)] \leqq t, \qquad \text{for some } \alpha$$

where

$$Q_{s_\alpha}(x) = \delta_\alpha{}^s(x) + P(x)(1 + x^s), \qquad \text{for some } s, \qquad 0 < s \leqq r.$$

Then

$$\delta_\alpha{}^s(x) + P(x)(1 + x^s)$$

must be in one of the correctable cosets.

To have sync-correction capability r, the code must have sync-detection capability at least r and the syndromes of the error patterns for bit loss must be different than those for bit gain; that is

$$\{\delta^p(x) + P(x)(1 + x^p)\} \neq \{x^{n-q}[\delta^q(x) + P(x)(1 + x^q)]\}, \qquad 0 < p, q \leqq r \tag{32}$$

or

$$\{\delta^{p+q}(x) + P(x)(1 + x^{p+q})\} \neq 0 \tag{33}$$

or

$$\{\delta^{p+q}(x)\} \neq \{P(x)(1 + x^{p+q})\}. \tag{34}$$

If $p + q \geqq n - k$ then the degree of $\delta^{p+q}(x)$ can be as large as

$$p + q - 1 \geqq n - k - 1.$$

The right side of (34) has a degree at most $n - k - 1$. Thus, by proper choice of the coefficients of $\delta^{p+q}(x)$, one can always equate the two sides of (34). Hence, $p + q \leqq n - k - 1$ is a necessary condition to satisfy (34). But max p = max $q = r$ so that $2r \leqq n - k - 1$ is a necessary

condition for the code to have sync-correction capability r. Hence the theorem:

Theorem 3: The sync-correction capability of a (n,k) *coset code derived from a cyclic code cannot exceed* $(n - k - 1)/2$.

If $p + q \leqq n - k - 1$, $\{\delta^{p+q}(x)\} = \delta^{p+q}(x)$ so that if the degree of $\{P(x)(1 + x^{p+q})\}$ is at least $p + q$, then (34) will be satisfied; *i.e.*,

$$D\{P(x)(1 + x^{p+q})\} \geqq p + q \tag{35}$$

is sufficient for (34) where $D[G(x)]$ denotes the degree of polynomial $G(x)$.

Therefore, the necessary and sufficient conditions for a code to have sync-correction capability r are:

$$W[\delta^s(x) + P(x)(1 + x^s)] \geqq t + 1 \qquad \text{for all } 0 < s \leqq r \tag{36}$$

and

$$D\{P(x)(1 + x^{p+q})\} \geqq p + q \qquad \text{for all } 0 < p, \quad q \leqq r \tag{37}$$

where $Q_s(x)$ is the coset leader of the polynomial $\delta^s(x) + P(x)(1 + x^s)$.

3.1 *Scheme A — General Approach*

A decoder which corrects up to t errors while utilizing a code with minimum distance d_m can also detect up to $d = d_m - (t + 1)$ errors. Thus, if every error pattern generated by slippage $s \leqq r$ has weight between $t + 1$ and d, it will certainly be detected. It follows that these error patterns cannot be in $\mathcal{C}$. Based on this sufficient condition one can design $P(x)$ accordingly. Suppose

$$d \geqq W[\delta^s(x) + P(x)(1 + x^s)] \geqq t + 1 \tag{38}$$

for all random polynomials $\delta^s(x)$ for $0 < s \leqq r$. Let

$$P(x) = \sum_{i=0}^{n-1} P_i x^i \tag{39}$$

and

$$T_s(x) = \sum_{i=0}^{n-s-1} (P_i + P_{i+s})x^{i+s}, \qquad 0 < s \leqq r. \tag{40}$$

Then

$$d - s \geqq W(T_s(x)) \geqq t + 1, \qquad 0 < s \leqq r, \tag{41}$$

is necessary and sufficient for (38) to hold. In particular, it is necessary that $W[T_s(x)] \geqq t + 1$. It follows that the best choice for $P(x)$, in the sense of maximizing r, is for

$$W[T_s(x)] = t + 1, \qquad \text{for all } 0 < s \leqq r. \tag{42}$$

Those $P(x)$ that satisfy (42) will be called optimal. In general, for certain special codes it is possible to find other schemes which work even if $d - r \geqq t + 1$ is not satisfied. This possibility will be discussed later.

It can be shown by direct substitution* in (41) that the following $P(x)$ are optimal.

$$P(x) = \sum_{\sigma=\sigma_0}^{[t/2]} x^{\sigma(r+1)-\sigma_0} + x^{n-1} \tag{43}$$

if

$$r + [t/2](r + 1) - \sigma_0 < n - 1$$

where

$$\sigma_0 = 1 + 2[t/2] - t \tag{44}$$

and $[t/2]$ represents the integer part of $t/2$.

TABLE III

Code	d_m	Number of Errors Corrected t	Levy's Result (Max. Number of Sync Loss Detected)	Optimal Result (Max. Number of Sync Loss Detected)
(23,12)	7	1	2	3
(127,85)	13	1	8	9
		2	5	7
		3	2	5
(255,191)	17	1	12	13
		2	9	11
		3	6	9
		4	3	7
(255,163)	25	1	20	21
		2	17	19
		3	14	17
		4	11	15
		5	8	13

Levy[9] has found a set of $P(x)$ for the purpose of detecting synchronization loss. Table III compares Levy's results† with some of the optimal results just derived. It is interesting to note that appreciable improvements are obtained by optimal $P(x)$.

It remains to show that the polynomials $P(x)$ of (43) possess sync-recovery capability. The following theorem characterizes the extent of slippage the code can correct provided that $d - r \geqq t + 1$.

* See Appendix.

† See Ref. 9, page 11, Table 1. Note $\delta \geqq t + 1$ is necessary to detect sync loss. Table III is constructed by letting $\delta = t + 1$.

Theorem 4: To every t error-correcting cyclic code there exists a coset code with sync-recovery capability of at least

$$r < \frac{n - k - t + [t/2]}{2 + [t/2]} \text{ bits}$$

provided that:

(*i*) *the coset code is generated by*

$$P(x) = \sum_{\sigma=\sigma_0}^{[t/2]} x^{\sigma(r+1)-\sigma_0} + x^{n-1}$$

(*ii*) $d - r \geqq t + 1$

(*iii*) $r + [t/2](r + 1) - \sigma_0 < n - 1$

where

$$\sigma_0 = 1 + 2[t/2] - t.$$

Proof: Since $P(x)$ has sync-detection capability r when $d - r \geqq t + 1$, all one has to check is that the syndrome of error patterns due to bit loss are different from those due to bit gain.

From (37) one requires

$$\begin{aligned} D\{P(x)(1 + x^{p+q})\} &= D\left\{\left(\sum_{\sigma=\sigma_0}^{[t/2]} x^{\sigma(r+1)-\sigma_0} + x^{n-1}\right)(1 + x^{p+q})\right\} \\ &= D\{1 + x^{p+q} + \cdots + x^{[t/2](r+1)-\sigma_0+p+q} \\ &\quad + x^{n+1}\} \leqq p + q, \qquad 1 \leqq p + q \leqq r. \end{aligned} \tag{45}$$

Let the generator polynomial be denoted

$$g(x) = \sum_{i=0}^{n-k} g_i x^i ;$$

then

$$x^{-1} g(x) = g_0 x^{-1} + \sum_{i=1}^{n-k} g_i x^{i-1} .$$

In general, one may assume $g_0 = g_{n-k} = 1$, and because $x^n = 1$, we have

$$x^{-1} g(x) = x^{n-1} + \sum_{i=1}^{n-k} g_i x^{i-1} = x^{n-1} + Q(x)$$

or

$$x^{n-1} = x^{n-1} g(x) + Q(x).$$

Hence

$$\{x^{n-1}\} = Q(x),$$

but as $Q(x)$ has a degree exactly $n - k - 1$; it follows that

$$D\{x^{n-1}\} = n - k - 1.$$

Therefore, (45) has a degree $n - k - 1$ if the next highest order term, $x^{[t/2](r+1)+p+q-\sigma_0}$ is always of a degree less than $n - k - 1$, for all $1 \leqq p, q \leqq r$; i.e.,

$$[t/2](r + 1) + p + q - \sigma_0 < n - k - 1, \quad 1 \leqq p, q \leqq r. \tag{46}$$

But, max $(p + q) = 2r$, i.e.,

$$[t/2](r + 1) + 2r - \sigma_0 < n - k - 1$$

or

$$r < \frac{n - k - 1 + \sigma_0 - [t/2]}{2 + [t/2]}.$$

Recall that $\sigma_0 = 1 + 2[t/2] - t$, and therefore

$$r < \frac{n - k - t + [t/2]}{2 + [t/2]}$$

is sufficient to satisfy (45).

Example: Find $P(x)$ for (15,6) BCH code where $d_m = 6$.

(*i*) If the code is used for single error correction, $t = 1$ and

$$d - r = d_m - (t + 1) - r \geqq t + 1.$$

Therefore, $r \leqq 2$, but

$$\frac{n - k - t + [t/2]}{2 + [t/2]} = (9 - 1)/2 = 4$$

so that by Theorem 4, $P(x) = 1 + x^{14}$ is a valid pattern for sync correction up to 2 bits.

(*ii*) If the code is used for double error correction $t = 2$ and $d = 3$. Thus, $d - r = 3 - r \geqq t + 1 = 3$. Therefore, $r = 0$, so that it is impossible to use this method but there are still other possibilities which will be discussed in the next section.

3.2 *Scheme B — Special Case*

The scheme just derived is applicable to all binary cyclic codes provided that $d_m > 2t + 2$, where t is the number of errors the decoder ac-

tually corrects, and that the scheme is not applicable to those systems in which the decoder is designed to correct $t = [(d_m - 1)/2]$ errors.

In this section, a different technique is developed. Instead of requiring that $d_m > 2t + 2$, this technique requires that a set of conditions on the code structure be satisfied. These conditions are almost always satisfied by *Bose-Chaudhuri-Hocquenghem* codes that correct more than two errors. For such codes it will be shown that it is always possible to obtain a coset code that has sync-recovery capability of at least one bit, even if the decoder is designed to correct errors up to the guaranteed error-correcting capability of the code, i.e., $t = [(d_m - 1)/2]$.

Definition 9: The weight of a coset is defined as the weight of its coset leader.

Definition 10: Code C_2 is said to be a descendant of code C_1 if

$$a \in C_1 \Rightarrow a \in C_2, \qquad C_1 \neq C_2$$

and is denoted by the notation

$$C_1 \subset C_2 .$$

Theorem 5: $C_1 \subset C_2$ *if and only if* $g_2(x) \mid g_1(x)$
where

$g_1(x)$ *and* $g_2(x)$ *are the generators of codes* C_1 *and* C_2, *respectively.*

Proof: (*i*) Since $g_2(x)$ itself is a code word of C_2, if $g_2(x)$ does not divide $g_1(x)$ then $g_1(x) \notin C_2 \Rightarrow C_1 \not\subset C_2$; a contradiction.

(*ii*) If $g_2(x) \mid g_1(x)$, then let

$$\begin{aligned} r(x) &= g_1(x)/g_2(x) \\ w(x) \in C_1 \Leftrightarrow w(x) &= g_1(x)f(x) \\ &= g_2(x)r(x)f(x) \\ &\Leftrightarrow w(x) \in C_2 . \end{aligned}$$

Hence, $C_1 \subset C_2$, by definition.

Theorem 6: Suppose code C_1 *corrects* t_1 *errors and code* C_2 *corrects* t_2 *errors. If* $C_1 \subset C_2$ *and* $t_1 > t_2$, *then the coset* Ω *of* C_1 *that the code word* $K(x) \in (C_2 - C_1)$ *belongs, must have a weight of at least* $2t_2 + 1$.

Proof: By definition $(C_2 - C_1)$ is nonempty, so there exists $K(x) \in C_2$ but $K(x) \notin C_1$. All the elements, of the coset Ω of C_1 to which $K(x)$ belongs, must be of the form

$$K(x) + w(x), \qquad w(x) \in C_1$$

but, $C_1 \subset C_2$, so

$$w(x) \in C_2 .$$

It follows that

$$K(x) + w(x) \in C_2 ;$$

thus, the weight of $K(x) + w(x)$ must be at least $2t_2 + 1$, as C_2 is a t_2-error-correcting code. It follows that Ω must have weight at least $2t_2 + 1$.

Theorem 7: Let Ω be a coset of a code C, $K(x)$ an element $\in \Omega$, and suppose the weight of Ω is R. Then the coset Ω_1 to which $K(x) + x^i$ belongs must have weight not less than $R - 1$.

Proof:

$$\begin{aligned}\Omega_1 &= \{(K(x) + x^i) + w_k(x) : w_k(x) \in C\} \\ &= \{(K(x) + w_k(x)) + x^i : w_k(x) \in C\} \\ &= \{\Omega(x) + x^i : \Omega(x) \in \Omega\}.\end{aligned}$$

Since Ω has weight R, each element of Ω_1 differs from an element in Ω by exactly one term. It follows that the weight of Ω_1 is at least $R - 1$.

The following several lemmas, in associated with Theorems 6 and 7, are essential to Theorem 8 which is the basis for Scheme B.

Lemma 2: $\{K(x)\} \not\equiv \{x^{n-1}K(x)\} \quad [\text{mod } g_1(x)]$

if

$$K(x) \in C_2 - C_1, \qquad C_1 \subset C_2$$

and

$$g_1(x)/g_2(x) = r(x) \not| (1 + x)$$

where $g_1(x)$ and, $g_2(x)$ are generator polynomials of C_1, C_2, respectively.

Proof: By hypothesis,

$$(1 + x) \not| r(x). \tag{47}$$

Assume

$$\{K(x)\} = \{x^{n-1}K(x)\} \qquad [\text{mod } g_1(x)]$$

i.e.,

$$\{(1 + x)K(x)\} = 0 \qquad [\text{mod } g_1(x)] \tag{48}$$

or

$$(1 + x)K(x) = f(x)g_1(x).$$

Now,

$$(1 + x) \mid f(x) \Rightarrow K(x) = f_1(x)g_1(x)$$

$$\Rightarrow K(x) \in C_1 \text{ ; a contradiction.}$$

It follows that $(1 + x) \nmid f(x)$. (49)

Since $K(x) \in C_2$,

i.e.,

$$K(x) = f_2(x)g_2(x),$$

by the assumption of (48)

$$(1 + x)K(x) = (1 + x)f_2(x)g_2(x) = f(x)g_1(x)$$
$$= f(x)g_2(x)r(x)$$

or

$$(1 + x)f_2(x) = f(x)r(x). \tag{50}$$

In view of (47) and (49), and by the unique factorization theorem, (50) cannot be satisfied. The lemma follows by contradiction.

Lemma 3: $\{K(x) + 1\} \neq \{x^{n-1}K(x) + x^{n-1}\} \quad [\text{mod } g_1(x)]$
if

$$K(x) \in C_2 - C_1 \quad \textit{and} \quad C_1 \subset C_2.$$

Proof: Assume the contrary.
Then

$$K(x) + 1 + x^{n-1}K(x) + x^{n-1} = f(x)g_1(x)$$
$$= f(x)g_2(x)r(x) \tag{51}$$

but

$$K(x) \in C_2 \Rightarrow g_2(x) \mid K(x).$$

The right-hand side is divisible by $g_2(x)$ but not the left-hand side unless

$$g_2(x) \nmid (1 + x^{n-1}).$$

The code generated by $g_2(x)$ has natural length n so that $g_2(x) \mid (1 + x^n)$ but not any $1 + x^k$, $k < n$. Therefore, (51) cannot hold and the lemma follows.

Lemma 4: $\{K(x)\} \neq \{x^{n-1}K(x) + x^{n-1}\} \quad [\text{mod } g_1(x)]$ *if*

$$K(x) \in C_2 - C_1, \qquad C_1 \subset C_2.$$

Proof: Assume the contrary.
Then

$$x^{n-1}K(x) + K(x) + x^{n-1} = f(x)g_2(x)r(x).$$

The left-hand side is not divisible by $g_2(x)$, since $g_2(x) \mid K(x)$, so the lemma follows.

Lemma 5: $\{K(x) + 1\} \neq \{x^{n-1}K(x)\} \quad [\text{mod } g_1(x)]$ *if*

$$K(x) \in C_2 - C_1, \qquad C_1 \subset C_2.$$

Proof: Similar to Lemma 4.

Theorem 8: Suppose code C_1 and C_2, with generator $g_1(x)$ and $g_2(x)$, correct t_1 and t_2 errors, respectively, and $2t_2 > t_1$. If $C_1 \subset C_2$,

$$g_2(x)/g_1(x) = r(x) \not\mid (1 + x)$$

and if $K(x) \in C_2 - C_1$, then the pattern

$$P(x) = \left\{\frac{K(x)}{1 + x}\right\},$$

if $K(x)$ has even weight, or

$$P(x) = \left\{\frac{K(x) + 1}{1 + x}\right\},$$

if $K(x)$ has odd weight, defines a coset code of C_1 with sync-recovery capability of at least one bit.

Proof: First note that such $P(x)$ always exists. Now with the $P(x)$ used to define the coset code, the error pattern for one-bit loss in sync is

$$\{P(x)(1 + x) + \delta^1(x)\}$$

$$= \{K(x) + \delta^1(x)\} \qquad \text{if } K(x) \text{ is even}$$

$$= \{K(x) + 1 + \delta^1(x)\} = \{K(x) + \delta^1(x)\} \qquad \text{if } K(x) \text{ is odd.}$$

That is, the syndromes are either $\{K(x)\}$ or $\{K(x) + 1\}$. By Theorem 6, $\{K(x)\}$ does not belong to a correctable coset of C_1 since $2t_2 > t_1$. The coset corresponding to $\{K(x) + 1\}$ has weight equal to or greater than $2t_2$, by Theorem 7, so that it is not correctable.

Similarly, the error pattern for one-bit gain is

$$\begin{aligned}
&\{P(x)(1 + x^{n-1}) + x^{n-1}\delta^1(x)\} \\
&\quad = \{x^{n-1}[P(x)(1 + x) + \delta^1(x)]\} \\
&\quad = \{x^{n-1}K(x) + x^{n-1}\delta^1(x)\} && \text{if } K(x) \text{ is even} \\
&\quad = \{x^{n-1}(K(x) + 1) + x^{n-1}\delta^1(x)\} \\
&\quad = \{x^{n-1}K(x) + x^{n-1}\delta^1(x)\} && \text{if } K(x) \text{ is odd.}
\end{aligned}$$

That is, the syndromes are either

$$\{x^{n-1}K(x) + x^{n-1}\} \quad \text{or} \quad \{x^{n-1}K(x)\}$$

By Theorem 6 and Lemma 1, $\{K'(x)\} = \{x^{n-1}K(x)\}$ is not in any of the correctable cosets of C_1, and by Theorem 7 and Lemma 1,

$$\{x^{n-1}K(x) + x^{n-1}\} = \{x^{n-1}(K(x) + 1)\}$$

is not correctable either. It follows that $P(x)$ satisfies the first condition that all the error patterns of one-bit slippage generated by $P(x)$ can not be in any of the correctable cosets of C_1 so that C_1 has sync-detection capability of at least one bit.

By Lemmas 2, 3, 4, and 5 the cosets corresponding to bit-loss patterns can not be the same as the bit-gain patterns. Thus the second condition is satisfied. It follows that code C_1 has sync-recovery capability of at least one bit.

The search for $K(x)$ is very simple since $g_2(x) \in C_2$, $g_2(x) \notin C_1$ so that we may use $g_2(x)$ for $K(x)$ in every instance.

Thus the procedure to find a coset code for use with Scheme B is as follows.

(*i*) Find a code, C_2, of the given code C_1 such that $g_2(x)/g_1(x) = r(x) \not| (1 + x)$ and that $2t_2 > t_1$.

(*ii*) Use

$$\left\{\frac{g_2(x)}{1 + x}\right\} = P(x)$$

to generate the desired coset code if the weight of $g_2(x)$ is even and use

$$\left\{\frac{g_2(x) + 1}{1 + x}\right\} = P(x)$$

if $g_2(x)$ has odd weight.

From Theorem 8, it is easy to see that this procedure applies to all Bose-Chaudhuri-Hocquenghem codes whenever $2t_2 > t_1$, and many other algebraic codes.

The following example shows how to apply Theorem 8 to Bose-Chaudhuri-Hocquenghem codes.

The polynomial and the associated sync-loss error syndromes $P(x)$ for a (15,5) BCH triple-error-correcting code generated by

$$g_1(x) = (x^4 + x + 1)(x^4 + x^3 + x^2 + x + 1)(x^2 + x + 1)$$

can be found as follows:

If α is a root of $x^4 + x + 1$ then α^5 is a root of $x^2 + x + 1$ (see tables of Marsh[13] or Peterson[12]) so that

$$r(x) = (x^2 + x + 1) \nmid (1 + x)$$

$$g_2(x) = \frac{g_1(x)}{r(x)} = (x^4 + x + 1)(x^4 + x^3 + x^2 + x + 1)$$

generates a double error-correcting BCH code.

Observe that

$$t_1 = 3, \qquad t_2 = 2$$

therefore,

$$2t_2 > t_1,$$

hence, all the requirements for Scheme B are satisfied. Set

$$K(x) = g_2(x),$$

and since $g_2(x)$ has odd weight use

$$P(x) = \left\{\frac{K(x) + 1}{1 + x}\right\} = \left\{\frac{x^8 + x^7 + x^6 + x^4}{1 + x}\right\}$$
$$= x^7 + x^5 + x^4.$$

The syndromes are:

(*i*) bit loss

$$\{K(x) + 1\} = x^8 + x^7 + x^6 + x^4$$

or

$$\{K(x)\} = x^8 + x^7 + x^6 + x^4 + 1.$$

(*ii*) bit gain

$$\{x^{n-1}K(x)\} = x^9 + x^6 + x^5 + x^4 + x + 1$$

or

$$\{x^{n-1}K(x) + x^{n-1}\} = x^7 + x^6 + x^5 + x^3.$$

It can be verified that all the syndromes listed do not belong to correctable cosets of the given code, and the syndromes are obviously all different, so the modified (15,5) code has sync-correction capability of one bit.

Notice that for this code

$$d_m = 7, \qquad t_1 = 3;$$

therefore,

$$d = d_m - (t + 1) = 7 - 4 = 3$$

and

$$t_1 + 1 = 4.$$

The condition, $d - r \geqq t + 1$, is not satisfied so that Scheme A is not applicable.

3.3 *Implementation*

By the Euclidean division algorithm, one writes

$$(1 + x^s)P(x) = g(x)F(x) + R_s(x)$$

where $D[R_s(x)] < D[g(x)] = n - k$.

From (28), the s-bit-loss syndromes are

$$\{\delta^s(x) + (1 + x^s)P(x)\} = \{\delta^s(x)\} + \{(1 + x^s)P(x) = \delta^s(x) + Rs(x)$$

(Since $D[\delta^s(x)] < D[g(x)]$) for all $0 < s \leqq r$.

It follows that all possible s-bit-loss syndromes have the same high order $n - k - s$ terms [namely, the high order $n - k - s$ terms of $Rs(x)$] and the remaining low order s-terms assume all possible 2^s combinations. Thus, the 2^s possible syndromes for an s-bit-loss can be detected by a single *and* gate that recognizes the high order $n - k - s$ terms of $R_s(x)$.

According to (28) and (30), s-bit-gain syndromes are the same as the s-bit-loss syndromes multiplied by x^{n-s}. It follows that a bit-loss recognition device, as mentioned above, can also be used to test bit-gain syndromes. This can be done by transforming bit-gain syndromes to bit-loss syndromes through multiplication by x^s, $0 < s \leqq r$, then testing the resultant syndromes with the bit-loss recognition device. Such a device takes r *and* gates and is applicable to both Scheme A and B.

IV. CYCLIC CODES FOR DETECTION ONLY

Some parts of the subject discussed in this section have been investigated in the literature.[8] Here, a different point of view is presented.

If a cyclic code is used for error detection only, then any error pattern due to sync loss or gain can be detected so long as the erroneous words generated by sync loss are not in the code space. If r is the maximum amount of slippage possible (such that the misframed words are not code words), the code is usually said to possess comma-free freedom r. Codes having the property that $|r| = [(n + 1)/2]$, are called "comma free".

Consider a coset code C generated by $P(x)$ and designed to detect sync loss or gain. To assure that such error patterns are detectable, one requires, by (28) and lemma 1,

$$\{(1 + x^{\beta})P(x) + \delta^{\beta}(x)\} \neq 0 \tag{52}$$

since $\mathfrak{C} = 0$ because every coset is not correctable for error-detecting codes.

As any syndrome has a degree which is at most $n - k - 1$, and $\delta^{\beta}(x)$ is an arbitrary polynomial of degree $\beta - 1$, it is always possible, for any $P(x)$ to have

$$\{(1 + x^{\beta})P(x) + \delta^{\beta}(x)\} = 0 \qquad \text{if} \qquad \beta > n - k - 1.$$

It follows that the comma-free freedom can never exceed $n - k - 1$. Hence, we have the following theorem.

Theorem 9: The comma-free freedom of any cyclic code cannot exceed $n - k - 1$.

One sees that by letting $P(x) = 1$; (52) now reads

$$\{(1 + x^{\beta})\cdot 1 + \delta^{\beta}(x)\} = \{1 + x^{\beta} + \delta^{\beta}(x)\} = x^{\beta} + \delta^{\beta}(x). \tag{53}$$

The term x^{β} cannot be canceled by $\delta^{\beta}(x)$ provided $\beta \leqq n - k - 1$. It follows that the upper bound on comma-free freedom described by Theorem 9 can be met. Hence, we have the following result.

Theorem 10: The comma-free freedom of any cyclic code can be made as large as $n - k - 1$.

According to Theorem 10, the comma-free freedom r cannot be greater than $n - k - 1$. Thus, if $k \geqq (n - 1)/2$, it is impossible to detect all the sync loss for $n - k \leqq r \leqq k$; on the other hand, if $k < (n - 1)/2$ then the interval between $n - k$, k does not exist. Hence, every slippage can be detected. It follows that

Corollary 1: An (n,k) *cyclic code can be made comma-free if and only if* $k < (n-1)/2$.

The above results have been proved in a different manner by J. J. Stiffler.[8]

Clearly, for strictly error-detecting codes, if a received message is not a code word, no distinction can be made to decide whether the error is caused by additive noise or is due to sync loss. However, statistical decisions still can be made accurately by observing the number and frequency of word errors, and, if it is concluded that a sync loss has occurred, sync can be recovered by sliding the word frame until the number of errors observed abruptly reduces to a predetermined level. The penalty for such a process is, of course, the time delay and the loss of data.

V. CONCLUSIONS

Techniques for automatic word synchronization recovery are presented. The techniques are useful if the slippage of word framing is not large, which is presumably the usual case.

An upper bound on the synchronization recovery capability for any cyclic code is found. It is shown that, for recovery to be possible, the amount of slippage in bits, r, cannot exceed $(n-k-1)/2$. It is also shown that the synchronization-loss detection capability of any cyclic code is upper-bounded by $n-k-1$ bits and furthermore, the bound can be met.

For shortened cyclic codes, the technique has five valuable features:

(*i*) No additional redundancy is required if the code corrects more than two errors, and only two additional check bits are required if the code corrects one or two errors. The generation of such check bits is simple.

(*ii*) The normal error-correcting ability of the code is not reduced when synchronization is maintained.

(*iii*) The correction of synchronization loss can be accomplished even in the presence of additive noise.

(*iv*) The time delay required before proper framing is restored is small, usually one-word time.

(*v*) The implementation is very simple.

The technique has been successfully applied to an existing error control unit[14] which utilizes a triple-error-correcting (200,175) code. The net cost of the additional hardware is about 20 transistors. The circuit corrects at least one-bit synchronization loss and, with diminishing probabilities, corrects larger sync losses as well.

For cyclic codes which are not shortened, necessary and sufficient conditions for the existence of a suitable coset code without additional redundancy for the recovery of synchronization loss are derived; and a class of optimal codes is given.

Two schemes are presented for finding such coset codes. The first scheme, called Scheme A, applies to any cyclic code whose minimum distance is greater than $2t + 2$, where t is the number of random errors the decoder actually corrects.

Such a scheme is usually applicable to data systems with a reverse channel in which case high error-detecting ability is utilized to obtain very-high-accuracy transmission. This requires that the error-correcting ability of the code be reduced in favor of the detecting ability. That is to say, in such a case, the minimum distance of the code is often greater than $2t + 2$.

For systems using forward-acting error correction only, the error-correcting ability of the code is usually exploited fully so that the requirements of Scheme A may not be met. A special technique, called Scheme B, are developed for such situation. Instead of requiring $d_m > 2t + 2$, Scheme B requires a set of conditions which are almost always satisfied by Bose-Chaudhuri-Hocquenghem codes that correct more than two errors. Both schemes have the advantages mentioned earlier for shortened cyclic codes except there is no noise tolerance.

Word synchronization loss is a catastrophic failure in error-control systems. The techniques herein described offer a solution to this problem for all binary cyclic codes with negligible cost in hardware, in time delay, and without loss in transmitting efficiency in many cases.

VI. ACKNOWLEDGMENTS

The author would like to express his gratitude to his thesis advisors, Professor E. J. McCluskey, Jr. and Professor A. McKellar, for their encouragement and advice. The author is also deeply indebted to H. O. Burton for many stimulating discussions and for assisting with the preparation of this manuscript.

Thanks are due to E. J. Weldon, Jr. for many references he provided and to P. G. Neumann for his valuable suggestions.

APPENDIX

The Verification that $P(x)$ is Optimal

$$P(x) = \sum_{i=0}^{n-1} P_i x^i = \sum_{\sigma=\sigma_0}^{[t/2]} x^{\sigma(r+1)-\sigma_0} + x^{n-1} \tag{54}$$

where

$$r + [t/2](r + 1) - \sigma_0 < n - 1 \tag{55}$$

and

$$\sigma_0 = 1 + 2[t/2] - t.$$

Now

$$T_s(x) = \sum_{i=0}^{n-s-1} (P_i + P_{i+s})x^{i+s} \qquad 0 < s \leqq r. \tag{56}$$

Consider

$$\begin{aligned}(1 + x^s)P(x) &= \left(\sum_{\sigma=\sigma_0}^{[t/2]} x^{\sigma(r+1)-\sigma_0} + x^{n-1}\right)(1 + x^s) \\ &= \sum_{\sigma=\sigma_0}^{[t/2]} (x^{\sigma(r+1)-\sigma_0} + x^{\sigma(r+1)-\sigma_0+s}) \\ &\quad + x^{n-1} + x^{s-1}.\end{aligned} \tag{57}$$

Note that:

(*i*) $\sigma(r + 1) - \sigma_0 \neq \sigma(r + 1) - \sigma_0 + s$ for all $0 < s \leqq r$.

(*ii*) The exponent of the highest order term under the summation sign, $[t/2](r + 1) - \sigma_0 + s < [t/2](r + 1) - \sigma_0 + r < n - 1$ by (55).

Therefore, no terms of (57) will be canceled for all $0 < s \leqq r$.

Note $T_s(x)$ is the same as the polynomial (57) with the lower order terms $x^0, x^1, \cdots, x^{s-1}$ removed so that the number of nonzero terms (hence the weight) of $T_s(x)$ is equal to the total number of terms of (57) minus the number of nonzero terms with exponent $s - 1$ or less.

Case 1: t is odd.

$$\sigma_0 = 1 + 2[t/2] - t = 1 + 2\frac{t - 1}{2} - t = 0.$$

There are two nonzero terms in (57) which have exponents no greater than $s - 1$, namely $x^{\sigma_0(r+1)-\sigma_0} = x^0$ and x^{s-1}. Therefore, the weight of $T_s(x)$

$$\begin{aligned}&= 2([t/2] + 2) - 2 = 2\left(\frac{t - 1}{2} + 2\right) - 2 \\ &= t + 1.\end{aligned}$$

Case 2: t is even.

$$\sigma_0 = 1 + 2[t/2] - t = 1 + 2(t/2) - t = 1.$$

There is only one nonzero term, namely, x^{s-1} that has an exponent no greater than $s - 1$. Therefore, the weight of $T_s(x)$

$$= 2([t/2] + 1) - 1 = 2(t/2 + 1) - 1 = t + 1.$$

REFERENCES

1. Gilbert, E. N., Synchronization of Binary Messages, IEEE Trans. Inform. Theor., *IT-6*, No. 4, September, 1960, pp. 470–477.
2. Neumann, P. G., On a Class of Efficient Error Limiting Variable Length Codes, IEEE Trans. Inform. Theor., *IT-8*, No. 5, September, 1962, pp. 260–260.
3. Neumann, P. G., Efficient Error Limiting Variable Length Codes, *Ibid, IT-8*, No. 4, July, 1962, pp. 292–304.
4. Neumann, P. G., Error Limiting Coding Using Information Loseless Sequential Machine, *Ibid, IT-10*, No. 2, April, 1964, pp. 108–115.
5. Stiffler, J. J., Synchronization of Telemetry Codes, IEEE Trans. Space Electron. and Telemet., *SET-8*, No. 3, June, 1962, pp. 112–117.
6. Barker, R. H., Group Synchronizing of Binary Digital Systems, *Communication Theory*, W. Jackson, Ed., Academic Press, 1953, pp. 273–287.
7. Sellers, F. F., Bit Loss and Gain Correction Code, IEEE Trans. Inform. Theor., *IT-8*, No. 1, January, 1962, pp. 35–38.
8. Stiffler, J. J., Comma-Free Error-Correcting Codes, IEEE Trans. Inform. Theor., *IT-11*, No. 1, January, 1965, pp. 107–112.
9. Levy, J. E., Self-Synchronizing Codes Derived from Binary Cyclic Codes, Unpublished Report, ADCOM, Inc., Cambridge, Mass., March, 1965.
10. Bose, R. C. and Ray-Chaudhuri, D. K., On a Class of Error-Correcting Binary Group Codes, Information and Control, No. 3, 1960, pp. 68–79.
11. Hocquenghem, A., Codes Correcteurs d'erreurs, Chiffres 2, 1959, pp. 147–156.
12. Peterson, W. W., *Error-Correcting Codes*, MIT Press, 1961.
13. Marsh, R. W., *Table of Irreducible Polynomials Over GF(2) Through Degree 19*, NSA, Washington, D.C., 1957.
14. Burton, H. O., and Weldon, E. J., An Error Control System for Use With a High-Speed Voiceband Data Set, 1st IEEE Communication Conference Record, June, 1965, 485–488.
15. Frey, Jr., A. H., Message Framing and Error Control, IEEE Trans. Mil. Electron., *MIL-9*, No. 2, April, 1965, pp. 143–147.
16. Golomb, S. W., et al., Comma-Free Codes, Canada J. Math., *10*, No. 2, 1958.
17. Gorenstein, D., Peterson, W. W., and Zierler, N., Two Error Correcting Bose-Chaudhuri Codes are Quasi-Perfect, Information and Control, *3*, 1960, pp. 291–294.
18. Ullman, J. D., Near Optimal Single Synchronization Error-Correcting Codes, Technical Report No. 45, Digital System Labs, Department of E.E., Princeton University, March, 1965.