Step-by-step Decoding of the Bose-Chaudhuri-Hocquenghem Codes

JAMES L. MASSEY, MEMBER, IEEE

Abstract—A new and conceptually simple decoding procedure is developed for all of the cyclic Bose-Chaudhuri-Hocquenghem codes. If t is the number of errors guaranteed correctable by the Bose-Chaudhuri bound, then any pattern of t or fewer errors can be corrected in a step-by-step manner using this procedure. In the binary case, the method requires only the determination of whether a $t \times t$ matrix is singular. In the general case, the method requires only the determination of whether a $t \times t$ matrix and a $(t + 1) \times (t + 1)$ matrix are simultaneously singular. Circuits to implement the algorithm are developed and two detailed examples are given. Finally, the step-by-step procedure is compared to other known methods for decoding the Bose-Chaudhuri-Hocquenghem codes.

I. Introduction

A NEW DECODING PROCEDURE for all of the cyclic Bose-Chaudhuri-Hocquenghem (BCH) codes is presented in Sections II and III of this paper. The method is called a step-by-step procedure after Prange [1] since it involves changing received symbols one at a time with testing to determine whether the weight of the error pattern has been reduced. In this section, we review several properties of the BCH codes that will be exploited in the sequel. The notation in Peterson [2] has been followed as closely as possible.

Let $g(X) = g_0 + g_1X + \cdots + g_{r-1}X^{r-1} + X^r$, $g_0 \neq 0$, be a monic polynomial of degree r with coefficients in $GF(q)$, the finite field of q elements. Let n be the least integer such that $g(X)$ divides $X^n - 1$. With each polynomial $f(X) = f_0 + f_1X + \cdots + f_{n-1}X^{n-1}$ of degree $n - 1$ or less with coefficients in $GF(q)$, associate the n-tuple or vector $\mathbf{f} = (f_0, f_1, \cdots f_{n-1})$. Then the *cyclic code* generated by $g(X)$ is the set of all vectors $\mathbf{f}$ such that $g(X)$ divides $f(X)$. The code length is n digits and the code redundancy is r digits.

The cyclic BCH codes may then be described in the following manner. Let α be any nonzero element of $GF(q^m)$ and take $g(X)$ as the monic polynomial of minimum degree having α^{m_0}, α^{m_0+1}, $\cdots$ α^{m_0+d-2} as roots, where m_0 is some integer and $d \geqq 2$ is any integer such that the specified roots are all distinct. The resulting code is a cyclic BCH code. When it is necessary to be explicit, we shall refer to such a code as a BCH(q, m_0, d) code. The Bose-Chaudhuri bound [3] states that every such code has minimum distance at least d. (The $q = 2$ cyclic codes were discovered by Bose and Chaudhuri [3], but in non-cyclic form had been found earlier, and independently, by Hocquenghem [4]. The nonbinary generalization was carried out by Gorenstein and Zierler [5].)

We now proceed to state some properties of general cyclic codes and of the BCH codes. Proofs, when not given here, may be found in Peterson [2]. We adopt the convention that code vectors $\mathbf{f}$ are transmitted with highest order digits first, i.e. f_{n-1} is the first transmitted digit, f_{n-2} is the second, etc. Let $i(X) = i_rX^r + i_{r+1}X^{r+1} + \cdots + i_{n-1}X^{n-1}$ be the polynomial corresponding to the $n - r$ information digits $i_r, i_{r+1}, \cdots i_{n-1}$ to be encoded in a cyclic code. The cyclic code is said to be *systematic* when $i(X)$ is identical to the terms of degree r and greater in the corresponding code word $f(X)$. Let $R_g[p(X)]$ denote the *remainder* when the polynomial $p(X)$ is divided by $g(X)$, then:

Property 1: Any cyclic code such that

$$f(X) = i(X) - R_g[i(X)]$$

is a systematic code.[1]

Hereafter we assume that all cyclic codes are in systematic form.

When $\mathbf{f}$ is transmitted, $\mathbf{r} = \mathbf{f} + \mathbf{e}$ is received where $\mathbf{r} = (r_0, r_1, \cdots r_{n-1})$ is the received code word and $\mathbf{e} = (e_0, e_1, \cdots e_{n-1})$ is the error pattern. e_i is nonzero if and only if $r_i \neq f_i$. The total number of nonzero components in $\mathbf{e}$ is the (Hamming) weight of the error pattern, i.e. the total number of errors. It will prove convenient to distinguish errors in the *information* positions from errors in the *parity* positions by defining

$$e_i(X) = e_rX^r + e_{r+1}X^{r+1} + \cdots + e_{n-1}X^{n-1} \qquad (1)$$

and

$$e_p(X) = e_0 + e_1X + \cdots + e_{r-1}X^{r-1}. \qquad (2)$$

The *syndrome*, $s(X) = s_0 + s_1X + \cdots + s_{r-1}X^{r-1}$, is the remainder when $r(X)$ is divided by $g(X)$ and hence can always be formed at the receiver. Since $r(X) = f(X) + e(X)$ and since $g(X)$ divides $f(X)$, it follows that

$$s(X) = R_g[e(X)]. \qquad (3)$$

Property 2: Using (1) and (2), it follows immediately from (3) that for any systematic cyclic code,

$$s(X) = R_g[e_i(X)] + e_p(X).$$

Manuscript received March 20, 1965. Part of the research reported in this paper was performed by the author while he was a consultant to Codex Corp., Watertown, Mass.

The author is with the Dept of Electrical Engineering, University of Notre Dame, Notre Dame, Ind.

[1] See Peterson, [6].

For the cyclic BCH codes, the quantities S_j are elements of $GF(q^m)$ defined by

$$S_j = e(\alpha^{m_0+j-1}) \qquad j = 1, 2, \cdots \tag{4}$$

and have played an important role in previous decoding algorithms for the BCH codes [2], [5]–[7]. In general, only a restricted set of these quantities can be formed at the receiver.

Property 3: For any BCH(q, m_0, d) code,

$$S_j = s(\alpha^{m_0+j-1}) \qquad j = 1, 2, \cdots, d-1.$$

Proof: Let $p(X)$ be the quotient when $e_i(X)$ is divided by $g(X)$, then

$$e_i(X) = g(X)p(X) + R_g[e_i(X)].$$

Thus

$$e_i(\alpha^{m_0+j-1}) = R_g[e_i(\alpha^{m_0+j-1})], \qquad j = 1, 2, \cdots, d-1,$$

since by the definition of a BCH code, all of these arguments are roots of $g(X)$. The remainder of the proof follows immediately from property 2.

Since $s(X)$ is computable at the receiver, it follows from property 3 that so are $S_1, S_2, \cdots S_{d-1}$. In general, however, S_j for $j \geqq d$ is not calculable at the receiver. (We note in passing that the calculation of the S_j according to property 3 is always simpler than the calculation $S_j = r(\alpha^{m_0+j-1})$ that has been suggested elsewhere [2], [7].) For our purposes, the importance of the S_j lies in the following properties which are essentially theorems 9.3 and 9.4 in Peterson [2].

Property 4: For any BCH(2, 1, d) code and any j such that $2 \leqq j \leqq n$, the $j \times j$ matrix

$$M_j = \begin{bmatrix} 1 & 0 & 0 & 0 & \cdots & 0 \\ S_2 & S_1 & 1 & 0 & \cdots & 0 \\ & & \vdots & & & \\ S_{2j-2} & S_{2j-3} & S_{2j-4} & S_{2j-5} & \cdots & S_{j-1} \end{bmatrix}$$

is singular if the weight of $\mathbf{e}$ is $j-2$ or less, and is nonsingular if the weight of $\mathbf{e}$ is $j-1$ or j.[2]

Clearly, $\det(M_j) = \det(L_{j-1})$ where L_{j-1} is the $(j-1) \times (j-1)$ matrix formed by eliminating the first row and first column from M_j. Thus, property 4 may be rephrased as

Properly 4′: For any BCH(2, 1, d) code and any j such that $1 \leqq j \leqq n-1$, the $j \times j$ matrix

$$L_j = \begin{bmatrix} S_1 & 1 & 0 & 0 & \cdots & 0 \\ S_3 & S_2 & S_1 & 1 & \cdots & 0 \\ & & \vdots & & & \\ S_{2j-1} & S_{2j-2} & S_{2j-3} & S_{2j-4} & \cdots & S_j \end{bmatrix}$$

is singular if the weight of $\mathbf{e}$ is $j-1$ or less, and is nonsingular if the weight of $\mathbf{e}$ is j or $j+1$.

Property 5: For any BCH(q, m_0, d) code and any j such that $1 \leqq j \leqq n$, the $j \times j$ matrix

$$N_j = \begin{bmatrix} S_1 & S_2 & \cdots & S_j \\ S_2 & S_3 & \cdots & S_{j+1} \\ & & \vdots & \\ S_j & S_{j+1} & \cdots & S_{2j-1} \end{bmatrix}$$

is singular if the weight of $\mathbf{e}$ is $j-1$ or less, and is nonsingular if the weight of $\mathbf{e}$ is j.[3]

It should be stressed at this point that, by the remarks made after property 3, only certain matrices L_j and N_j can be constructed at the receiver. In general, L_t and N_t are the largest such matrices that can be constructed for a BCH code with $d = 2t + 1$.

For ease of future reference, a short tabulation of $\det(L_j)$ is given in Table I.

TABLE I

j	det (L_j)
1	S_1
2	$S_1^3 + S_3$
3	$S_1^6 + S_1^3S_3 + S_1S_5 + S_3^2$
4	$S_1^{10} + S_1^7S_3 + S_1^5S_5 + S_1^3S_7 + S_1^2S_3S_5 + S_1S_3^3 + S_3S_7 + S_5^2$

II. Decoding the Binary BCH Codes with $m_0 = 1$

The most important binary BCH codes are those for $m_0 = 1$ and $d = 2t + 1$. From the Bose-Chaudhuri bound, it follows that these codes are at least t-error-correcting. We now prove two theorems that result in a conceptually simple step-by-step decoding algorithm for these codes which will correct all error patterns of weight t or less.

Theorem 1: Assume an error pattern $\mathbf{e}$ of weight t or less in a BCH(2, 1, $2t+1$) code. If $\det(L_t) = 0$, change in order digits $s_0, s_1, \cdots s_{2t-2}$ of the syndrome $s(X)$ until (as will always occur) $\det(L_t) \neq 0$. Then this new syndrome corresponds to an error pattern of weight exactly t with the same $e_i(X)$ as for the original error pattern.

Proof: By property 2, changing s_j is equivalent to changing e_j in $e_p(X)$. If $e_j = 0$, changing it increases the error pattern weight by one; if $e_j = 1$, changing it decreases the error pattern weight by one. If any changing is required, at most $t-1$ of the e_j's could be "1" and hence at most $2t-1$ changes will be needed to increase the error pattern weight to t. Since the error pattern weight changes in unit steps, it follows by property 4′ that the first occurrence of $\det(L_t) \neq 0$ will signal the presence of exactly t errors. Only digits in $e_p(X)$ are altered so $e_i(X)$ remains unchanged. Finally, there are always enough digits to be changed since the fact that $g(X)$ has $2t$ dis-

[2] See Peterson, [6].

[3] See Gorenstein and Zierler, [5].

tinct specified roots guarantees that $2t - 2 < r - 1$.

Theorem 2: Assume an error pattern $\mathbf{e}$ of weight exactly t in a BCH$(2, 1, 2t + 1)$ code. Consider changing temporarily and one at a time the received information digits $r_{n-1}, r_{n-2}, \cdots r_r$ and testing each time whether $\det(L_t)$ vanishes. When and only when $\det(L_t) = 0$, the corresponding information digit was received in error (and hence its correct value is the complement of the received digit).

Proof: Suppose $r_j \neq i_j$. Then changing r_j reduces the error pattern weight to $t - 1$ and hence, by property 4′, $\det(L_t) = 0$. Conversely, suppose $r_j = i_j$. Then changing r_j increases the error pattern weight to $t + 1$ and hence, by property 4′, $\det(L_t) \neq 0$.

Theorems 1 and 2 together establish the validity of the following algorithm for the correction of t or fewer errors in a BCH$(2, 1, 2t + 1)$ code:

Step 0) Set $j = 0$.
Step 1) Determine from $s(X)$ whether $\det(L_t) = 0$.
Step 2) If $\det(L_t) = 0$, complement s_j, increase j by one, and go to step 1. Otherwise, set $j = 1$ and go to step 3.
Step 3) Temporarily complement r_{n-j} and determine from the modified syndrome whether $\det(L_t) = 0$.
Step 4) If $\det(L_t) = 0$, set $i_{n-j} = r_{n-j} + 1$. Otherwise, set $i_{n-j} = r_{n-j}$.
Step 5) If $j = n - r$, stop. Otherwise, increase j by one and go to step 3.

(This algorithm decodes the information digits only. If it is desired to decode the received parity digits also, this may be done by modifying step 2 to read "$\cdots$ complement s_j and $r_j, \cdots$" and modifying step 5 to read "If $j = n, \cdots$") It is noteworthy that the only significant computation is that required to determine whether $\det(L_t) = 0$. At most, this must be done $(n - r) + (2t - 1) < n$ times, i.e. once for each of the $n - r$ information digits and at most $2t - 1$ times before $\det(L_t) \neq 0$ in step 2.

The entire algorithm may be implemented efficiently by a modification of the general cyclic decoder proposed by Meggitt [8]. A block diagram of such a decoder, with explanatory notes, is shown in Fig. 1. The lower shift-register in Fig. 1 is a well-known syndrome calculator for a cyclic code.[4] The cyclic structure of the code makes it possible to treat successive received digits as though each were r_{n-1}. Then changing each such digit is equivalent to adding a fixed polynomial $a(X)$ to the syndrome where $a(X)$ is the syndrome when r_{n-1} is the only digit in error, i.e. $a(X) = R_g[X^{n-1}]$ by property 2.

The principal component of the decoder in Fig. 1 is the element whose input is the modified syndrome and whose output is a "one" when and only when $\det(L_t) = 0$. The design of this element will be discussed in Section IV.

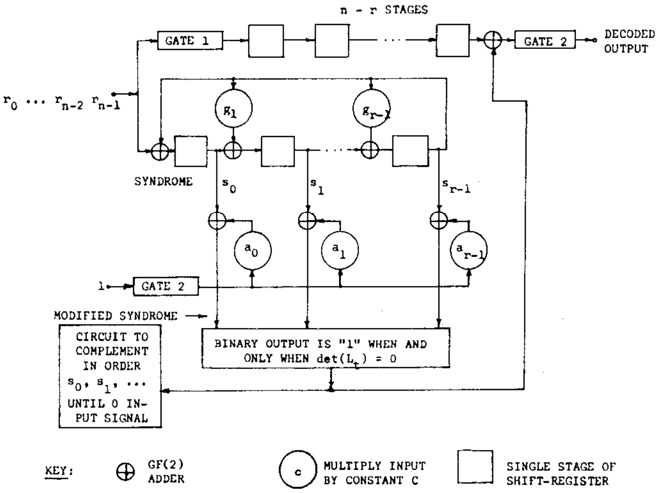
n - r STAGES
GATE 1
GATE 2
DECODED OUTPUT
r0 ··· rn-2 rn-1
g1
gr-1
SYNDROME
s0
s1
sr-1
a0
a1
ar-1
1
GATE 2
MODIFIED SYNDROME
CIRCUIT TO COMPLEMENT IN ORDER s0, s1, ··· UNTIL 0 INPUT SIGNAL
BINARY OUTPUT IS "1" WHEN AND ONLY WHEN det(Lt) = 0
KEY:
GF(2) ADDER
c
MULTIPLY INPUT BY CONSTANT C
SINGLE STAGE OF SHIFT-REGISTER

Fig. 1. General step-by-step decoder for binary BCH codes with $m_0 = 1$.

(Note 1: $(a_0, a_1, \cdots a_{r-1})$ is the syndome for $\mathbf{e} = (0, 0, \cdots 0, 1)$.
Note 2: Gate 1 is energized while the $n - r$ received information digits are read into the upper shift-register and is de-energized thereafter. Upper shift-register does not shift again until "decode" order is given.
Note 3: Complementing circuit begins to function after all n received digits are read into the decoder. It ceases to function when it receives a "0" input signal at which time the "decode" order is given.
Note 4: Gates 2 are energized by the decode order.
Note 5: Lower shift-register, after all n received digits are read in, does not shift again until "decode" order is given.)

III. Decoding Arbitrary BCH Codes

We begin by stating analogs of Theorems 1 and 2 that result in a step-by-step decoding algorithm for the nonbinary BCH codes (and the binary codes with $m_0 \neq 1$).

Theorem 3: Assume an error pattern $\mathbf{e}$ of weight t or less in a BCH (q, m_0, d) code where $d \geqq 2t + 1$. If $\det(N_t) = 0$, change (in any way) in order the digits $s_0, s_1, \cdots s_{2t-2}$ of the syndrome until (as will always occur) $\det(N_t) \neq 0$. Then this new syndrome corresponds to an error pattern of weight exactly t with the same $e_i(X)$ as for the original error pattern.

Proof: The proof of Theorem 1 applies with the only difference being that if $e_j \neq 0$ is changed to another nonzero value, then the error pattern weight is unchanged. This in no way alters the rest of the proof.

Theorem 4: Assume an error pattern of weight exactly t in a BCH (q, m_0, d) code where $d \geqq 2t + 1$. Consider varying temporarily and one at a time the received information digits $r_{n-1}, r_{n-2}, \cdots r_r$ through all $q - 1$ possible different values, and determining each time whether $\det(N_t) = \det(N_{t+1}) = 0$. When and only when $\det(N_t) = \det(N_{t+1}) = 0$, the corresponding information digit was received in error and its correct value is the value of the received digit that caused both determinants to vanish.

Proof: If $r_j \neq i_j$, then changing r_j to equal i_j will reduce the error pattern weight to $t - 1$ and hence cause $\det(N_t) = \det(N_{t+1}) = 0$ by property 5. For all other values

[4] See Peterson, [2].

of r_j, the error pattern still has weight t and hence det $(N_t) \neq 0$. Conversely, if $r_j = i_j$, then any change in r_j increases the error pattern weight to $t + 1$ and hence, by property 5, det $(N_{t+1}) \neq 0$.

Theorems 3 and 4 would form the basis of a step-by-step decoding algorithm except that, in general, when $d = 2t + 1$, S_{2t+1} and hence N_{t+1} are *not* calculable at the receiver. This difficulty can be obviated in the following manner. Note first that

$$N_{t+1} = \left[\begin{array}{ccc|c} & & & S_{t+1} \\ & N_t & & \vdots \\ & & & S_{2t} \\ \hline S_{t+1} & \cdots & S_{2t} & S_{2t+1} \end{array}\right]$$

so that the cofactor of S_{2t+1} is simply det (N_t). Thus, when det $(N_t) = 0$, det (N_{t+1}) is *independent* of S_{2t+1}. We are thus led to define the matrix N_{t+1}^0 as the matrix obtained from N_{t+1} by replacing S_{2t+1} with 0. It follows that det $(N_t) =$ det $(N_{t+1}) = 0$ when and only when det $(N_t) =$ det $(N_{t+1}^0) = 0$.

Theorem 5: Thus, we have established that Theorem 4 remains true if det (N_{t+1}) is replaced with det (N_{t+1}^0).

It follows from property 3 that N_{t+1}^0 can always be constructed at the receiver from the syndrome $s(X)$. Thus, there is sufficient information at the receiver to carry out the following algorithm for the correction of any error pattern of weight t or less in any BCH (q, m_0, d) code having $d \geqq 2t + 1$.

Step 0) Set $j = 0$.

Step 1) Determine from $s(X)$ whether det $(N_t) = 0$.

Step 2) If det $(N_t) = 0$, change digit s_j (say by adding to it the element 1 of $GF(q)$), increase j by one, and go to step 1. Otherwise, set $j = 1$ and go to step 3.

Step 3) Temporarily add in order all the $q - 1$ nonzero elements of $GF(q)$ to r_{n-j} and determine each time from the modified syndrome whether det $(N_t) =$ det $(N_{t+1}^0) = 0$.

Step 4) If β was the element which when added to r_{n-j} caused det $(N_t) =$ det $(N_{t+1}^0) = 0$, set $i_{n-j} = r_{n-j} + \beta$. Otherwise, set $i_{n-j} = r_{n-j}$.

Step 5) If $j = n - r$, stop. Otherwise increase j by one and go to step 3.

(If it is desired to decode the parity digits also, modify step 2 to read "...change digits s_j and r_j by adding the same element of $GF(q)$ to each ..." and modify step 5 to read "If $j = n$, ...")

A block diagram of a cyclic decoder to implement this algorithm is shown in Fig. 2. The operation is so similar to that of the decoder in Fig. 1 that no further explanation should be required. The main component of this decoder is the element whose binary output is a "1" when and only when det $(N_t) =$ det $(N_{t+1}^0) = 0$.

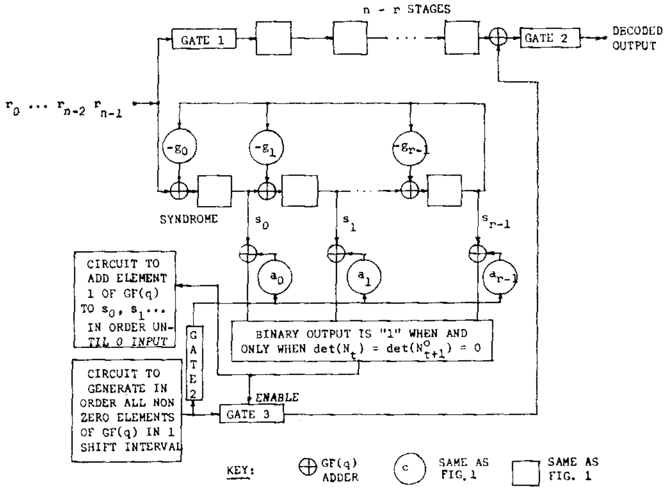
n - r STAGES
GATE 1
GATE 2
DECODED OUTPUT
r0 ··· rn-2 rn-1
-g0
-g1
-gr-1
SYNDROME
s0
s1
sr-1
a0
a1
ar-1
CIRCUIT TO ADD ELEMENT 1 OF GF(q) TO s0, s1... IN ORDER UNTIL 0 INPUT
GATE 2
BINARY OUTPUT IS "1" WHEN AND ONLY WHEN det(Nt) = det(N0t+1) = 0
CIRCUIT TO GENERATE IN ORDER ALL NON ZERO ELEMENTS OF GF(q) IN 1 SHIFT INTERVAL
ENABLE
GATE 3
KEY:
GF(q) ADDER
c
SAME AS FIG. 1
SAME AS FIG. 1

Fig. 2. General step-by-step decoder for all BCH (q, m_0, d) codes.

(Note 1: Notes 1–5 of Fig. 1 apply unchanged except that "complementing circuit" is replaced by "circuit to add element 1 of $GF(q)$."

Note 2: Gate 3 is energized when and only when a "1" is present on its enable input.)

IV. Two Examples

As has already been remarked, the only complex elements in the decoders of Figs. 1 and 2 are the "determinant computing" elements. To illustrate the design of these elements, and to show the wide choice available to the designer for their realization, we shall now give the detailed design of this element for the (15, 7) double-error-correcting and the (31, 16) triple-error-correcting binary BCH codes.

The (15, 7) code is a BCH (2, 1, 5) code where α is a primitive element of $GF(2^4)$. Let α be a root of the primitive polynomial $X^4 + X + 1$, i.e. $\alpha^4 = \alpha + 1$. Every element of $GF(2^4)$ may be represented as a binary 4-tuple over the basis $\alpha^0 = 1 = (1\ 0\ 0\ 0)$, $\alpha = (0\ 1\ 0\ 0)$, $\alpha^2 = (0\ 0\ 1\ 0)$, and $\alpha^3 = (0\ 0\ 0\ 1)$. For example, $\alpha^6 = \alpha^2\alpha^4 = \alpha^2(\alpha + 1) = \alpha^3 + \alpha^2 = (0\ 0\ 1\ 1)$. The minimum degree polynomial $g(X)$ having α, α^2, α^3, and α^4 as roots has degree 8. Thus $s(X) = s_0 + s_1X + \cdots + s_7X^7$ has degree 7 or less by property 2. Let $(a_0\ a_1\ a_2\ a_3)$ be the 4-tuple representation of $S_1 = s(\alpha)$, and let $(b_0\ b_1\ b_2\ b_3)$ be the 4-tuple representation of $S_3 = s(\alpha^3)$. Then it is readily verified from the representation scheme that

$$\begin{aligned} a_0 &= s_0 + s_4 + s_7 \\ a_1 &= s_1 + s_4 + s_5 + s_7 \\ a_2 &= s_2 + s_5 + s_6 \\ a_3 &= s_3 + s_6 + s_7 \\ b_0 &= s_0 + s_4 + s_5 \\ b_1 &= s_3 + s_4 \\ b_2 &= s_2 + s_4 + s_7 \\ b_3 &= s_1 + s_2 + s_3 + s_4 + s_6 + s_7, \end{aligned}$$

i.e. each a or b is a $GF(2)$ sum of selected syndrome digits. Thus the representations of S_1 and S_3 may be easily formed. (It will always be the case that each digit in the representation of any S_i will be a linear combination of syndrome digits whether the code is binary or not.)

At this point there is some freedom as to how det $(L_2) = S_1^3 + S_3$ will be calculated. In general, one might store the elements of the matrix L_t in a small special purpose computer which would then triangularize the matrix in the usual manner[5], det (L_t) vanishes when and only when a "0" appears on the main diagonal after triangularization. This process requires at most $t^3/3$ operations where an operation here is considered to be the calculation of an inverse or a multiplication in $GF(2^m)$. Alternatively, one could construct a combinatorial element whose inputs are the m-tuple representations of the S_i and whose binary output is a "1" if and only if det $(L_t) = 0$.

Such a combinatorial circuit is shown for the (15, 7) code in Fig. 3(a). It is readily checked that the output is a "1" if and only if det $(L_2) = 0$. The "cube" element is a device whose output is the 4-tuple representation of the cube of its input. This element could be implemented as a device which performs a "look-up" in a table of cubes for $GF(2^4)$. Alternatively, it could be implemented as a pure binary logical element as shown in Fig. 3(b) where advantage has been taken of the fact that under the foregoing representation scheme for $GF(2^4)$ the cube of $(a_0\ a_1\ a_2\ a_3)$ is $(c_0\ c_1\ c_2\ c_3)$ where

$$c_0 = a_0a_2' + a_1(a_2 + a_3)$$

$$e_1 = a_0(a_1 + a_2) + a_2'a_3$$

$$c_2 = (a_0 + a_2)(a_1 + a_2 + a_3) + a_1a_3$$

$$c_3 = (a_1 + a_2)a_3' + a_3$$

and where $'$ denotes the complement of the binary number.

Similar techniques can be applied to implement the "determinant computing" element for any step-by-step decoder. As a second example, consider the (31, 16) code which is a BCH (2, 1, 7) code where α is a primitive element of $GF(2^5)$. Taking α as a root of $X^5 + X^2 + 1$, and letting $S_1 = s(\alpha) = (a_0\ a_1\ a_2\ a_3\ a_4)$, $S_3 = s(\alpha^3) = (b_0\ b_1\ b_2\ b_3\ b_4)$ and $S_5 = s(\alpha^5) = (c_0\ c_1\ c_2\ c_3\ c_4)$, we can find each of the a's, b's and c's as a modulo-two sum of syndrome digits, for example

$$a_0 = s_0 + s_5 + s_8 + s_{10} + s_{11} + s_{14}.$$

Assuming that all the a's, b's, and c's have been computed by the necessary adding circuits, the remainder of the "determinant computing" circuit may be realized as shown in Fig. 4. It is readily verified that the output of this circuit is a"1" when and only when det $(L_3) = S_1^6 + S_1^3S_3 + S_1S_5 + S_3^2 = 0$.

For a thorough discussion of the design of circuits to do arithmetic in a finite field, the reader is referred to [10].

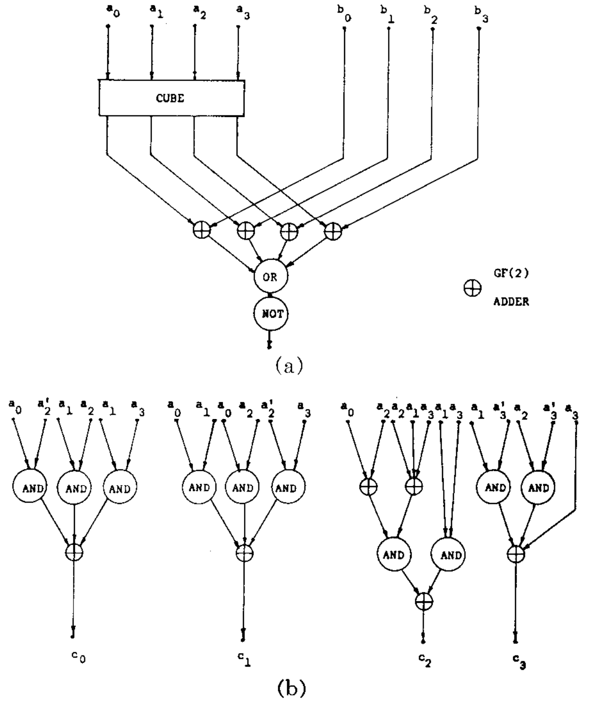

Fig. 3. (a) Determinant-computing circuit for BCH (15, 7) code. (b) Detail design of "cube" element.

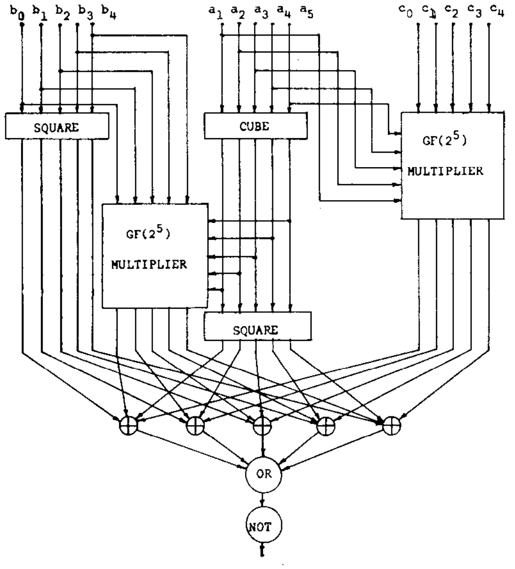

Fig. 4. Determinant-computing circuit for BCH (31, 16) code.

V. Remarks and Comparisons

An algebraic decoding procedure for the BCH (2, 1, $2t + 1$) codes was first developed by Peterson [2], [6]. Chien [7], by clever arguments that exploited the cyclic nature of these codes, simplified the Peterson algorithm so that for each decoded digit the decoder is required to determine only whether some determinant vanishes; however, the determinant to be investigated depends upon the weight of the error pattern. The step-by-step procedure developed here offers some advantage in that only det (L_t)

[5] See Pennington, [9].

need be investigated (thus avoiding a 'multiple-mode decoder"). Also det (L_t) is slightly simpler than the determinant Δ which must be computed in the Chien algorithm.

For the general cyclic BCH codes, the only known algebraic decoding procedure is that developed by Gorenstein and Zierler [2], [5]. This procedure requires extensive algebraic computation and the step-by-step procedure developed here would appear to be significantly easier to implement. Moreover, it is not clear that the Chien simplification can be applied to the Gorenstein-Zierler method with as much success as with the Peterson method.

The concept of step-by-step decoding originated with Prange [1], [2] who investigated its use in maximum-likelihood decoding. In Prange's formulation, the method requires the calculation of the weight of the minimum-weight error pattern consistent with the syndrome obtained as received digits are varied. The viewpoint here is simplified by the fact that the error pattern weight is always increased (if necessary) to be exactly t, thereafter one need only determine whether a change reduces the weight of the minimum-weight error pattern consistent with the syndrome.

Finally, we note that Berlekamp [11] has shown that det (M_{t+1}) is the same as the determinant of a parasymmetric matrix whose dimensions are (approximately) half those of M_{t+1}. Since det (L_t) = det (M_{t+1}), Berlekamp's result would considerably reduce the time required to calculate det (L_t) when a matrix triangularization is employed.

References

[1] E. Prange, "The use of coset equivalence in the analysis and decoding of group codes," USAF Cambridge Research Center, Bedford, Mass., Tech. Rept. AFCRC-TR-59-164, June 1959.

[2] W. W. Peterson, *Error-Correcting Codes.* New York: M.I.T. Press-Wiley, 1961.

[3] R. C. Bose and D. K. Ray-Chaudhuri, "On a class of error-correcting binary group codes," *Inform. Control*, vol. 3, pp. 68–79, March 1960.

[4] A. Hocquenghem, "Codes correcteurs d'erreurs," *Chiffres*, vol. 2, pp. 147–106, September 1959.

[5] D. Gorenstein and N. Zierler, "A class of cyclic linear error-correcting codes in p^m symbols," *J. SIAM*, vol. 9, pp. 207–214, June 1961.

[6] W. W. Peterson, "Encoding and error-correction procedures for the Bose-Chaudhuri codes," *IRE Trans. on Information Theory*, vol. IT-6, pp. 459–470, September 1960.

[7] R. T. Chien, "Cyclic decoding procedures for the Bose-Chaudhuri-Hocquenghem codes," *IEEE Trans. on Information Theory*, vol. IT-10, pp. 357–363, October 1964.

[8] J. E. Meggitt, "Error-correcting codes and their implementation for data transmission systems," *IRE Trans. on Information Theory*, vol. IT-7, pp. 234–244, October 1961.

[9] R. H. Pennington, *Introductory Computer Methods and Numerical Analysis.* New York: MacMillan, 1965, pp. 281–283.

[10] T. C. Bartee and D. I. Schneider, "Computation with finite fields," *Inform. Control*, vol. 6, pp. 79–98, March 1963.

[11] E. Berlekamp, "On decoding binary Bose-Chaudhuri-Hocquenghem codes," *IEEE Trans. on Information Theory*, vol. IT-11, pp. 577–579, October 1965.