REPORT R-640 FEBRUARY, 1974 UILU-ENG 74-2203

CSL COORDINATED SCIENCE LABORATORY

AN ERROR-CONTROL SYSTEM BASED ON MAJORITY-LOGIC DECODING

R.B. BROWN
C.L. CHEN
R.T. CHIEN
S. NG

UNIVERSITY OF ILLINOIS – URBANA, ILLINOIS

AN ERROR-CONTROL SYSTEM BASED ON MAJORITY-LOGIC DECODING

by

R. B. Brown, C. L. Chen, R. T. Chien and S. Ng

This work was supported in part by the Rome Air Development Center under Contract F30602-72-C-0031 and in part by the Joint Services Electronics Program (U.S. Army, U.S. Navy, and U.S. Air Force) under Contract DAAB07-72-C-0259.

Table of Contents

ACKNOWLEDGEMENT

The authors wish to thank Professor L. D. Rudolph of Syracuse University for the calculation of polynomials p(x) used in the design of the system. The authors are also indebted to their colleague Dr. D. Sarwate for his help in checking and testing the system.

AN ERROR-CONTROL SYSTEM BASED ON MAJORITY-LOGIC DECODING

R. B. Brown, C. L. Chen, R. T. Chien and S. Ng

I. INTRODUCTION

During the past two decades, many classes of error-correcting codes have been invented. However, these codes are rarely used in data transmission systems for combating channel noise. The reason for this is that the implementation of these codes is usually very complex and costly. From a practical point of view, the problem of finding efficient decoding algorithms that can be simply implemented is, therefore, the most important problem in coding applications today.

Among the codes that have been discovered, the class of BCH codes is perhaps the best known class of constructive codes [1,2]. These codes are efficient for error correction and error detection. In addition, due to the work by Peterson, Chien, and Berlekamp [1,2], there is a decoding algorithm which can be implemented with reasonable amount of equipment for this class of codes.

Codes based on finite geometries have been constructed by Rudolph [1,3]. These codes are called finite geometry codes. They include as subclasses Euclidean geometry codes and projective geometry codes. These codes can be made cyclic. Many of these codes are, in fact, BCH codes.

The important feature of finite geometry codes is that they are majority-logic decodable. That is, the decoding of a received code word can be simply implemented using majority-logic gates. In fact, many of the finite geometry codes can be more simply implemented than the BCH codes.

Recent developments in majority-logic decoding of finite geometry codes have greatly simplified the decoding complexity due to the reduction of the number of majority-logic gates [4,5]. This and other facts make finite geometry codes very attractive for practical error-control systems, especially for military applications.

An investigation was made on the problems in the design and implementation of finite geometry codes. As a result of the investigation, a binary coding system was constructed. The coding system meets the requirements stated in the work statement. In particular, four different error correcting codes are built into the system giving the user a choice of four different coding schemes with different error correcting capabilities. Three of the codes are 1/2-rate codes and the other one is a 1/4-rate code. The codes selected are all majority-logic decodable so that they can be implemented with a reasonable amount of hardware. The newest theory known to-date has been applied in the design of the majority-logic decoder. Each code is interleaved to enable the system to correct both random errors and burst errors. If an (n,k) code is capable of correcting t random errors, the interleaved code with interleaving degree ℓ is capable of correcting any errors that occur in a burst of ℓt positions. Many multiple burst-errors of shorter length are also correctable. The selection of interleaving degrees and codes was made primarily based on the requirements of the work statement. The system can operate at any channel rate between 300 and 19,200 bits per second. The system also includes automatic frame resynchronization and interface compatibility with MIL-STD-188C.

The system consists mainly of two parts, viz., an encoder and

decoder (Fig. 1). The encoder is to be inserted between the binary data source and the modulator (Fig. 1) of a regular digital data transmission system. Its function is to take every k binary information digits from the source and change them into n coded message digits, called a codeword, by adding (n-k) redundant check digits. The decoder is to be inserted between the demodulator and the binary data destinations (Fig. 1). Its purpose is to take each received codeword from the demodulator, corrupted by noise introduced by the transmission channel and the modem, and with the help of the redundant check digits recover from it the original k information digits.

Other supporting units of the system includes various control circuitries for the proper operation of the encoder and decoder, clock circuitries, and synchronization logic for establishing the starting point of each codeword.

The codes used are described in Section 2. These codes are selected based on the requirements of the work statement. With the exception of the (31,16) code (code D), these codes are less powerful in random error correction than the competitive BCH codes with the same rate. However, the decoding of these majority-logic decodable codes used are considerably simpler than that of the BCH codes. Code D is actually a BCH code. It has the same error correcting capability as the Golay (23,12) code. On a binary symmetric channel (BSC) the performance of code D is inferior but close to the performance of the Golay code. Implementation wise, code D is simpler. The performance of the codes used in the system and the Golay code on a BSC is shown by the curves in Fig. 12. Although it is not possible to make a direct comparison on the performance of these codes because of the differences among the code length and

the number of information digits, it can be seen that the (82,21) code consistantly outperforms code D and the Golay code in the sense that the probability of block decoding error is smaller for the same number of information digits in a block. This is due to the fact that the (82,21) code is a 1/4-rate while the other codes are 1/2-rate codes. We remark here that these codes may perform differently on a channel which cannot be modeled by BSC.

II. CODES USED

Four different codes are built into this system. A basic (n,k) t-error-correcting code encodes k information digits into a codeword of length n digits and can correct t or fewer errors occurring anywhere within each codeword. Interleaving the basic code b times results in a (nb, kb) code capable of correcting up to t bursts of errors, each burst of length b or less, occurring anywhere within the interleaved codeword of nb digits. Many patterns of random errors not guaranteed by the system are also correctable simply because majority-logic decoding does not perform bounded-distance decoding.

(a) The first code used is the punctured (253,127) 10 majority-error correcting EG code interleaved 25 times to yield a (6325,3175) code capable of correcting 8 bursts of length 25 each.* Twenty-five redundant bits are added to each codeword to make it an overall (6350,3175) 1/2 rate code.

(b) The second code used is the (127,64) 7-error correcting EG code

*The last two bits punctured in a codeword are assumed to be zero at the decoder. Thus the decoder actually guarantees the correction of 8 errors, although the original unpunctured code can correct 10 errors. However, the decoder is able to correct patterns of 9 errors 75% of the time, and correct patterns of 10 errors 25% of the time.

interleaved 50 times to yield a (6350,3200) code capable of correcting 7 bursts of length 50 each. Fifty redundant bits are added to each codeword to make it an overall (6400,3200) 1/2 rate code.

(c) The third code used is the shortened (82,21) 10-error-correcting PG code interleaved 75 times to yield a (6150,1575) code capable of correcting 10 bursts of length 75 each. One hundred and fifty redundant bits are added to each codeword to make it an overall (6300,1575) 1/4 rate code.

(d) The fourth code used is the (31,16) 3-error-correcting EG code interleaved 200 times to yield a (6200,3200) code capable of correcting 3 bursts of length 200 each. 200 redundant bits are added to each codeword to make it an overall (6400,3200) 1/2 rate code.

Twenty-five of the redundant bits in each interleaved codeword of the codes mentioned above are utilized for synchronization puprose. The synchronization scheme is discussed in Section 5. The following table is a summary of the codes used.

Basic Code (n,k,t)	Degree of Interleaving	Overall Code with Sync	Rate
A (253,127,10)**	25	(6350,3175)	1/2
B (127,64,7)	50	(6400,3200)	1/2
C (82,21,10)	75	(6300,1575)	1/4
D (31,16,3)	200	(6400,3200)	1/2

**See footnote on previous page.

III. ENCODER

A binary (n,k) cyclic code is completely specified by its parity check polynomial

$$h(x) = h_o + h_1 X + \cdots + h_k X^k \quad h_i \in GF(2)$$

such that any codeword $(a_o a_1 \cdots\cdots a_{n-1})$ satisfies

$$\sum_{j=0}^{k} h_j a_{j+i} = 0$$

for $0 \leq i \leq n-k-1$. For binary codes, $h_o = h_k = 1$. From here on a binary polynomial $A(x) = a_o + a_1 X + \ldots + a_{n-1} X^{n-1}$ will be represented by listing its exponent of X of nonzero terms. For example, the polynomial $1 + X + X^5 + X^{11}$ will be represented as (0, 1, 5, 11).

A general encoder for a cyclic (n, k) code with parity check polynomial h(x) is shown in Fig. 2. With the switch at position A, k information digits are shifted into the shift register. Next, with the switch moved to position B, the shift register is shifted n-k times. The n digits that come out of the output during this cycle form a codeword and it is in systematic form. If a basic (n, k) cyclic code is interleaved b times, one needs only to replace each stage of register in the encoder for the basic code by a b-stage shift register, and the resulting encoder is that of the (nb, kb) interleaved code. Thus as far as logic circuitry is concerned, one only needs to consider the basic code.

(a) Code A: This is a punctured cyclic code obtained by omitting two parity check digits from the (255, 127) cyclic code. The encoding circuit of Fig. 2 can still

be used, with the modification that after the switch has moved to position B the shift register needs only be shifted n-k-2 = 126 times instead of the n-k = 128 times required for the cyclic code. This will yield a total of 253 digits during each cycle of operation. The parity check polynomial of the original cyclic code is

$$h(x) = (0, 1, 2, 5, 6, 7, 8,$$
$$11, 14, 15, 16, 17, 20, 21,$$
$$22, 23, 24, 26, 28, 31, 33,$$
$$42, 45, 46, 49, 52, 53, 54,$$
$$55, 56, 60, 61, 63, 64, 67,$$
$$68, 69, 71, 73, 78, 79, 81,$$
$$82, 83, 84, 85, 87, 90, 93,$$
$$95, 97, 99, 100, 102, 104, 105,$$
$$106, 109, 111, 112, 113, 115, 117,$$
$$118, 120, 121, 123, 125, 126, 127)$$

(b) Code B: This is a cyclic code and hence can be implemented by the encoder of Fig. 2. Its parity check polynomial is

$$h(x) = (0, 2, 5, 6, 8, 9, 10, 11, 12,$$
$$15, 16, 21, 23, 24, 26, 30, 31,$$
$$32, 33, 34, 37, 38, 43, 45, 46, 47,$$
$$48, 49, 51, 53, 58, 60, 61, 62, 63, 64)$$

(c) Code C: This is a shortened cyclic code obtained by omitting three information digits from the cyclic (85,24)

code. The encoder circuit of Fig. 2 has to be modified to that of Fig. 3. At the beginning of each cyclic, switches S1, S2 and S3 (Fig. 3) are in position C and the "0" generator shifts out two 0's into the register. Next with both switches in position A the k-3 = 21 information digits are shifted into the register and sent out to the modem. The cycle is then completed by moving switch S1 to position B and shifting the register n-k = 61 additional times. This will yield a total of 82 digits during each cycle of operation. The parity check polynomial of the original cycle code is

$$h(x) = (0, 2, 3, 4, 7, 10, 11, 12, 14, 15, 17, 19, 24)$$

(d) Code D: This is a cyclic code. Thus its encoder is exactly of the form given in Fig. 2. Its parity check polynomial is

$$h(x) = (0, 4, 5, 6, 7, 12, 15, 16)$$

(e) The complete system encoder: Since the numbers of times the four different codes are interleaved are all multiples of 25, and their interleaved block lengths are approximately the same, it is possible for the encoders of the four different codes to share the same shift register, with 25 stage shift registers as the basic building block. The complete encoder for the system is as shown in Fig. 4. The switch S4 permits selection of any one of the four codes.

The operation of this system encoder can best be understood with an example. Consider the (127, 64) code interleaved 50 times to become the

(6350, 3200) code. For this code both the buffer registers B1 and B2 and the encoder register R (Fig. 4) are each used as a 3200-stage shift register. The actions of switch S1 and switch S2 are always complementary, i.e., if S1 is up, then S2 is down, and vice versa. Assume B1 is already completely filled with data bits, then with S1 in position B input data is shifted into B2 at a speed of 1/2T bits per sec. (bps), i.e., it takes 2T sec. to shift each bit in. At the same time with S2 in position C, S3 in E, and S5 in G, the data stored in B1 are shifted into R at twice the speed, i.e., 1/T bps. At the end of 3200T sec. R is completely filled with the 3200 data bits originally in B1, and 3200 output bits have already been generated. In the meantime 1600 bits have been shifted into B2. Next S3 is moved to F and R is again shifted at 1/T bps, while data continues to be shifted into B2 at 1/2T bps. At the end of 3150T sec. all 3150 parity check bits have been generated by the encoder to yield a total of 6350 output bits. At the same time B2 is now filled with 1575 additional data bits to contain a total of 3175 bits. S5 is now moved to H and 50 redundant bits are generated by the sync pattern generator and shifted out at 1/T bps, while 25 more data bits are shifted into B2 at 1/2T bps to completely fill B2. The same cycle can now be repeated by moving S1 to A and S2 to D.

The encoder operates in very much the same manner for the other three codes, except that for the (6150, 1575) 1/4 rate code, R has to be operated at four times the speed of B1 and B2, as compared to two times for the other three 1/2 rate codes.

IV. DECODER

As in the encoder, the decoder logic circuit of a (nb, kb) code,

obtained by interleaving a (n, k) code, can easily be derived from that of the basic (n, k) code by replacing each stage of its shift register by a b-stage shift register. Hence once again as far as logic circuitry is concerned, one only needs to consider the basic code.

(a) Code C: The decoding of this code is simpler than the other three codes and will be described first. It is different from the other three codes in that it requires only one level of majority decoding.

A block diagram of the decoder is shown in Fig. 5. It consists of an 85-stage shift register and a 20-input majority gate. The decoder register supplies the inputs to the majority gate by forming twenty parity check sums according to the following parity check equations:

(0, 1, 41, 25, 72)
(0, 2, 59, 50, 82)
(0, 3, 62, 53, 5)
(0, 4, 79, 15, 33)
(0, 6, 10, 21, 39)
(0, 7, 49, 27, 19)
(0, 8, 66, 30, 73)
(0, 9, 37, 35, 32)
(0, 11, 29, 75, 81)
(0, 12, 78, 42, 20)
(0, 13, 38, 14, 54)
(0, 16, 61, 60, 47)
(0, 17, 34, 68, 51)
(0, 18, 64, 70, 74)
(0, 22, 77, 65, 58)
(0, 23, 28, 26, 76)

(0, 24, 40, 84, 71)
(0, 31, 69, 45, 44)
(0, 36, 63, 55, 43)
(0, 46, 67, 52, 56)
(0, 48, 57, 83, 80)

Each check sum is the modulo 2 sum of the outputs of those stages of the register corresponding to each parity check equation. The 20-input majority gate is realized by a number of half-adders and full-adders as shown in Fig. 6. Its function is to output a 0 if ten or more of its inputs are 0, and output a 1 otherwise.

The decoder operates as follows. Assume the first three stages of the register is initially cleared. With the switch S at position A (Fig. 5), the 82 bits of the received codeword is shifted into the first 82 stages of the register. Next the switch S is moved to B and the register is shifted 21 times. The 21 information digits will then be decoded and appear as the output. S is then moved to C to shift in 0's, and the register is shifted 61 more times. This ensures that the first three stages of the register is cleared. The same cycle can now be repeated over again. Thus each codeword is decoded in two complete shifts of the decoder register. The decoding delay is one complete shift of the decoder register.

The remaining three codes all require two levels of majority decoding. A block diagram of a general decoder for a (n, k) t-error correcting cyclic code which requires two levels of majority decoding is shown in Fig. 7. It consists of two n-stage shift registers and two 2t-input majority gates. The first level register R1 supplies the inputs to the

first level majority gate by forming 2t parity check sums according to a set of 2t parity check equations. The output of the first level majority gate is fed into the second level register through a multiplying circuit which multiplies it by a given polynomial p(x). If the given polynomial $p(x) = p_o + p_1 X + \cdots + p_{n-1} X^{n-1}$, the multiplying circuit is as shown in Fig. 7. The second level register R2 supplies the inputs to the second level majority gate also by forming 2t parity check sums.

The decoder operates as follows. With switch S1 in position A and S2 open, the codeword to be decoded is shifted into R1 while R2 is also shifted at the same speed in the same time. After the first k shifts, R2 is broken at a number of places and 0's start to be shifted in to clear R2. After n-k additional shifts R1 is completely filled with the n bits of the codeword and R2 is completely cleared. S1 is then moved to B and S2 is closed, and both R1 and R2 are shifted n times. This completely fills R2 with the results of the first level of majority decoding. With S1 moved back to A and S2 open, the next cycle of operation can now be repeated. During the first k shifts of the next cycle the first k error digits of the previous codeword are determined by the second level of majority decoding and these are subtracted from the first k received digits of the previous codeword to yield the k information digits. Thus each codeword is decoded in two complete shifts of the decoder registers. The decoding delay is also two complete shifts of the decoder register, as compared to one complete shift for the one level majority decodable codes.

(b) Code D: This is a cyclic code and hence can be

implemented by the decoding circuit of Fig. 7. The six parity check equations for the first level are:

(0, 4, 12, 15, 1, 8, 13, 17)
(0, 4, 12, 15, 2, 9, 22, 27)
(0, 4, 12, 15, 3, 11, 25, 28)
(0, 4, 12, 15, 5, 6, 7, 16)
(0, 4, 12, 15, 14, 18, 20, 30)
(0, 4, 12, 15, 19, 21, 26, 29).

The polynomial p(x) is

p(x) = (3, 7, 10, 15, 16, 19, 22, 26, 27, 28).

The six parity check sums for the second level are:

(30, 3, 11, 14)
(30, 6, 17, 27)
(30, 7, 23, 29)
(30, 8, 10, 12)
(30, 15, 16, 28)
(30, 19, 20, 21)

A 6-input majority gate can be realized by a number of half-adders and full adders as shown in Fig. 8.

(c) Code B: This is also a cyclic code and hence can be implemented by the decoding circuit of Fig. 7. The 14 parity check equations for the first level are:

(0, 4, 114, 15, 24, 30, 85, 81,
31, 8, 120, 7, 117, 102, 34, 43)
(0, 4, 114, 15, 24, 30, 85, 81,
62, 64, 50, 93, 70, 14, 94, 66)
(0, 4, 114, 15, 24, 30, 85, 81,
82, 35, 75, 42, 111, 91, 47, 48)

(0, 4, 114, 15, 24, 30, 85, 81,
3, 11, 61, 1, 76, 123, 121, 27)
(0, 4, 114, 15, 24, 30, 85, 81,
44, 86, 55, 23, 72, 49, 52, 107)
(0, 4, 114, 15, 24, 30, 85, 81,
37, 18, 103, 92, 16, 84, 68, 105)
(0, 4, 114, 15, 24, 30, 85, 81,
28, 125, 73, 19, 40, 100, 65, 29)
(0, 4, 114, 15, 24, 30, 85, 81,
6, 41, 104, 45, 51, 2, 109, 79)
(0, 4, 114, 15, 24, 30, 85, 81,
115, 32, 88, 46, 10, 22, 33, 83)
(0, 4, 114, 15, 24, 30, 85, 81,
69, 95, 113, 77, 20, 108, 39, 96)
(0, 4, 114, 15, 24, 30, 85, 81,
106, 67, 90, 12, 101, 53, 60, 21)
(0, 4, 114, 15, 24, 30, 85, 81,
74, 119, 71, 97, 17, 57, 87, 122)
(0, 4, 114, 15, 24, 30, 85, 81,
116, 110, 36, 9, 54, 26, 25, 98)
(0, 4, 114, 15, 24, 30, 85, 81,
118, 5, 99, 89, 58, 112, 80, 13).

The polynomial p(x) is

p(x) = (11, 27, 33, 37, 44, 45, 49, 51,
54, 56, 58, 61, 62, 65, 70, 74,
76, 77, 79, 84, 85, 86, 87, 90,
91, 92, 94, 96, 98, 100, 102, 103,
104, 106, 108, 110, 111, 113, 119, 120,
124)

The 14 parity check sums for the second level are

(126	61	108	66	55	73	68	99)
(126	6	94	49	85	89	91	110)
(126	123	104	64	32	24	105	43)
(126	81	118	107	82	62	36	79)
(126	13	54	11	41	17	19	33)
(126	2	116	57	58	27	100	46)
(126	50	4	69	51	97	125	90)
(126	88	102	60	70	5	52	71)
(126	111	83	28	96	25	7	93)
(126	35	10	42	95	77	56	22)
(126	117	21	20	112	109	26	92)
(126	48	16	65	122	72	34	45)
(126	9	101	38	103	106	98	18)
(126	119	59	44	74	86	31	8)

A 14-input majority gate can be realized by a number of half-adders and full adders as shown in Fig. 9.

(d) Code A: This is a punctured cyclic code obtain by omitting two parity check digits from the (255, 127) cyclic code. The decoding circuit of Fig. 7 can still be used, with the modification that the received codeword is shifted into the first level shift register R1 at the third stage instead of the first stage for cyclic codes. Then only 253 shifts are required to fill the 255-stage register with the received codeword. The second round of shifting in each cycle still consists of 255 shifts.

The 20 parity check equations for the first level are:

(252 246 198 190 178 163 144 127 126 73 72 30 23 2 1 0)
(253 250 238 147 145 142 126 102 72 61 47 34 5 3 1 0)
(249 225 208 148 126 98 72 43 39 33 14 10 6 4 1 0)
(247 235 199 156 126 104 81 72 58 49 40 24 11 7 1 0)
(243 229 212 193 182 167 126 105 76 72 66 50 26 8 1 0)
(244 214 207 188 154 137 126 72 70 67 56 45 32 9 1 0)
(200 172 158 151 135 126 113 111 108 82 78 72 20 12 1 0)
(233 224 203 176 140 126 124 121 100 79 72 38 21 13 1 0)
(209 191 164 139 128 126 123 99 95 89 72 63 36 15 1 0)
(231 218 187 173 165 153 136 132 126 90 83 72 53 16 1 0)
(232 230 227 223 217 186 175 168 133 126 119 72 51 17 1 0)
(202 201 159 152 131 130 129 126 120 72 64 52 37 18 1 0)
(228 210 192 169 160 134 126 114 112 109 96 72 65 19 1 0)
(254 251 245 197 189 177 162 143 126 125 72 71 29 22 1 0)
(241 219 194 181 166 150 126 107 92 84 77 75 72 27 1 0)
(248 236 220 196 161 126 115 97 93 87 72 59 42 28 1 0)
(240 226 222 216 205 179 149 126 117 74 72 69 44 31 1 0)
(239 237 234 215 155 141 126 122 116 103 94 72 46 35 1 0)
(213 206 185 184 183 180 174 126 118 106 91 72 55 54 1 0)
(221 204 146 138 126 101 88 80 72 68 62 60 57 48 1 0)

The polynomial p(x) is

p(x) = (254, 251, 249, 248, 245, 244, 243, 242, 241,
237, 236, 234, 232, 230, 227, 222, 221, 220,
218, 217, 214, 212, 209, 207, 205, 201, 198,
195, 193, 192, 189, 187, 186, 185, 183, 180,
178, 172, 159, 156, 154, 149, 147, 143, 125,
122, 119, 117, 116, 113, 112, 111, 109, 108,
105, 103, 101, 98, 97, 93, 92, 88, 80,
77, 73, 72, 71, 69, 65, 64, 61, 59,
51, 45, 39, 38, 35, 34, 32, 30, 12,
2, 0)

The 20 parity check sums for the second level are:

(125	71	0	254)
(251	143	1	254)
(146	4	2	254)
(248	32	3	254)
(38	9	5	254)
(103	48	6	254)
(242	65	7	254)
(206	31	8	254)
(198	80	10	254)
(77	19	11	254)
(223	37	12	254)
(207	97	13	254)
(163	88	14	254)
(230	131	15	254)
(118	50	16	254)
(158	63	17	254)
(133	111	18	254)
(175	123	20	254)
(161	142	21	254)
(245	197	22	254)

A 20-input majority gate is as shown in Fig. 6.

(e) The complete system decoder: As in the encoder, it is possible for the decoders of the four different codes to share the same shift registers R1 and R2, with 25-stage shift registers as the basic building block. The complete decoder for the system is as shown in Fig. 10. For convenience of representation, only one 1/2 rate code and the (6150, 1575) 1/4 rate code are explicitly drawn. The switches S select the proper decoding circuit to be

used. They are closed for that particular code to be used.

As an example for the 1/2 rate code, assume the (6350, 3200) code is being used. For this code both input buffer registers B1 and B2 and decoder registers R1 and R2 are each used as a 6350-stage shift register, while the output buffer registers B3 and B4 are used as 3200-stage shift registers. Assume B1 is already completely filled with a received codeword and that the correct block sync has already been established from the 50 sync bits. Then with switch S1 in position B the next received codeword is shifted into B2 at a speed of 1/2T bps. At the same time with S2 in position C, S3 in E, and S4 open, the received codeword stored in B1 is shifted into R1 at twice the speed, i.e., 1/T bps. Also at the same time with S5 closed and S6 in H (assuming B3 is already filled with decoded information digits), R2 is shifted at 1/T bps. The decoded information digits of the previous received codeword, now shifting out of R1, will then be shifted into B4 also at 1/T bps. Also at the same time with S7 in I the decoded information digits in B3 are shifted out as output at 1/4T bps. After 3200 T sec. B2 contains 1600 bits of the codeword coming in from the line, while R1 has received the first 3200 bits of the codeword that has been stored in B1, and the 3200 information bits of the codeword previously contained in R1 have now been decoded and are stored in B4. Also 800 decoded bits have emerged from B3. Next with S5 open R2 is shifted to clear, while B1, B2, and R1 continues to be shifted as before. Thus at the end of 3150T additional sec. B2 is filled with 1575 additional bits, while the codeword originally stored in B1 has now been completely transferred to R1, and 787.5 additional bits have emerged from B3. R2 is now cleared and there is no change in B4. Now S3 is moved

to F, S4 is closed, and with S5 still open, both R1 and R2 are shifted at 1/T bps. B2 continues to be shifted as before. At the end of 6350 T additional sec. B2 is completely filled with the 6350 bits of a new codeword, and R2 is completely filled with the results of the first level of majority decoding. Also 1587.5 additional digits have emerged from B3 to make a total of 3175 bits. During the next 100 T sec. B1, B2, R1, R2, and B4 are idle while the 50 sync bits are being received and examined. B3 continues to be shifted as before and at the end of this 100 T sec. all 3200 bits have been shifted out of B3. Now with S1 moved to A, S2 to D, S3 to G, and S7 to J, the same cycle can be repeated over again.

Code D operates in precisely the same manner as described above. Code A also operates in the same manner, except that during the time the 25 sync bits are received R1 and R2 continue to be shifted.

Code C operates somewhat differently due to the fact that it is a 1/4 rate code and that it requires only one level of majority decoding. For this code B1 and B2 are used as 6150-stage shift register. Assume that both B1 and B3 have already been filled and that sync has already been established. The next received codeword is shifted into B2 at 1/2T bps., while the codeword stored in B1 is shifted into R1 at 1/T bps with S3 in position E and S5 open. At the same time the decoded digits in B3 are shifted out at 1/8T bps. At the end of 6150 T sec. the entire codeword originally stored in B1 is now transferred to R1, while B2 is filled with the first 3075 bits of the next codeword. Also 768.75 decoded digits have emerged from B3. Next with S3 in position F and S5 closed R1 is shifted at 1/T bps. and the resulting decoded digits are shifted into B4. At the

end of 1575 T sec. B4 is completely filled, and R1 and B4 become idle. B2 and B3 continue to be shifted as before. At the end of 4575 T additional sec. B2 is completely filled with a new codeword, while 1537.5 bits have emerged from B3. During the next 300 T sec the 150 sync bits are being received and examined. B3 continues to be shifted as before and at the end of this 300 T sec. all 1575 bits have been shifted out of B3. Now the same cycle can be repeated over again.

V. SYNCHRONIZATION SCHEME

The synchronization scheme described below is for frame synchronization. The purpose of frame synchronization is for the decoder to recognize the beginning of each block of coded bits. Bit synchronization is assumed to be provided by the modem or other external equipment.

The fixed pattern 11111 00101 10111 00010 00000 of 25 bits is selected as the sync pattern for frame synchronization.*

Data blocks are arranged as shown in Fig. 11, where N is a sequence of coded bits, and L is a sequence of alternating 1's and 0's. The number of bits in N and L depend on the codes used. These numbers are

(a) For code A, $N = 6325$, $L = 0$

(b) For code B, $N = 6350$, $L = 25$

(c) For code C, $N = 6150$, $L = 125$

(d) For code D, $N = 6200$, $L = 175$

In the decoder a sync pattern scanner is used to test the received

---

*It is shown in [6] that the probability of false sync when part of the received sync is in the 25-bit register is 7×10^{-5} if the channel errors are assumed to be independent with bit error rate of 10^{-1}.

bit stream for the presence of the 25-bit frame synchronizing signal and initiate the decoding process for the next N bits. The scanner consists of a 25-bit serial input shift register, 25 exclusive OR gates and a 25 input adder with five binary outputs: 2^0, 2^1, 2^2, 2^3 and 2^4.

The decoder input contains gating circuits which can gate the incoming bit stream to the sync shift register or to the input buffer. A 14-bit counter is used to count the N bits, and at the count of N, provides an "end-of-block" (EOB) signal to reset the counter and to switch the received signal from the buffer input back to the sync register input.

The exclusive-OR gates in the scanner section are "hard-wired" so that when the sync word is completely contained in the 25-bit shift register, each gate output is zero. In the event that one or more bits do not match the "hard-wired" word, the corresponding gates will produce a "1" output. Since each gate output drives one input to the 25 bit adder, the adder output is the number of mis-matches in the sync register. The five adder outputs are gated so that a "1" in positions 2^2, 2^3 or 2^4 will inhibit generation of a sync signal. In other words, an acceptable sync pattern may contain as many as three bits in error out of the 25. When outputs 2^2, 2^3 and 2^4 go to zero, a pulse is generated which causes the input signal to be switched from the sync register input to the data buffer input and resets the counter. The next N bits are accepted by the decoder as a valid data block.

The decoder input circuits also contain a sync error counter. A sync error is defined as follows:

$$\text{Error}=\text{SYNC} \cdot \overline{\text{EOB}} + \overline{\text{SYNC}} \cdot \text{EOB}$$

The presence of a sync signal or an EOB signal will cause the decoder to accept the next N bits as a valid block, even when such signals also produce a sync error, provided no more than three such errors have occurred in succession. A valid data block is defined as:

$$\text{SYNC} + \text{EOB} \cdot (\text{Number of Errors} < 4)$$

Four sync errors in succession will cause a red panel light to be turned "ON" displaying a sync error condition. If the sync search switch on the panel is in the "MANUAL" position, the indicator will remain "ON" until it is reset manually or until the switch is placed in "AUTO", and both the SYNC and EOB signals occur in coincidence.

If four successive sync errors have not occurred, and the sync search switch is in either the AUTO or MANUAL positions, the first time both EOB and SYNC signals are in coincidence the sync error counter is reset to zero. When in the AUTO mode, the sync scanner circuits cycle continuously, and the signal SYNC · EOB will cause the error counter to be reset regardless of the number of sync errors recorded.

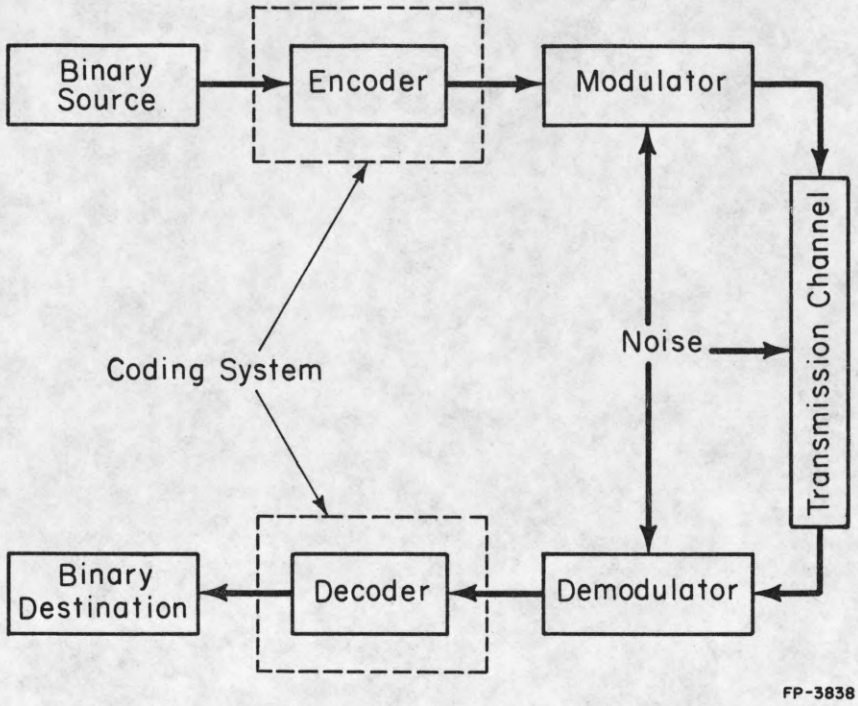

Fig. 1. Schematic of Coding System

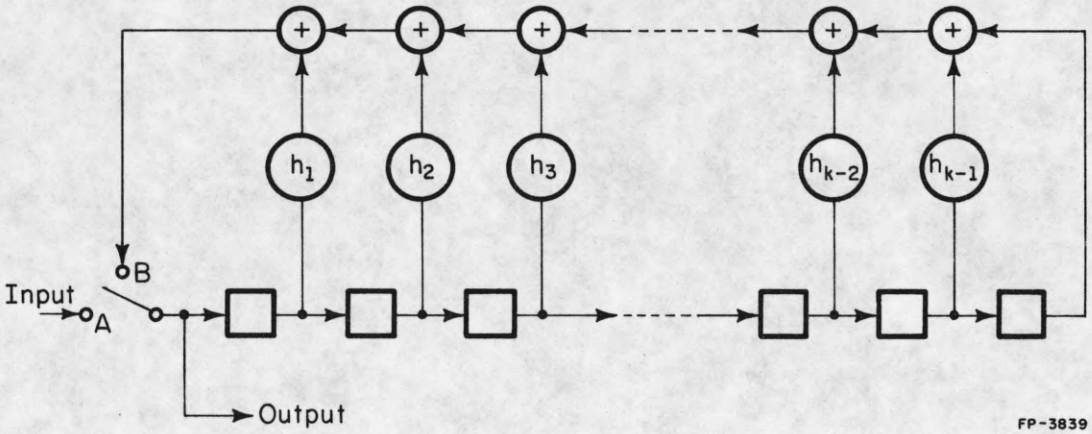

Fig. 2. General Encoder for Cyclic Codes

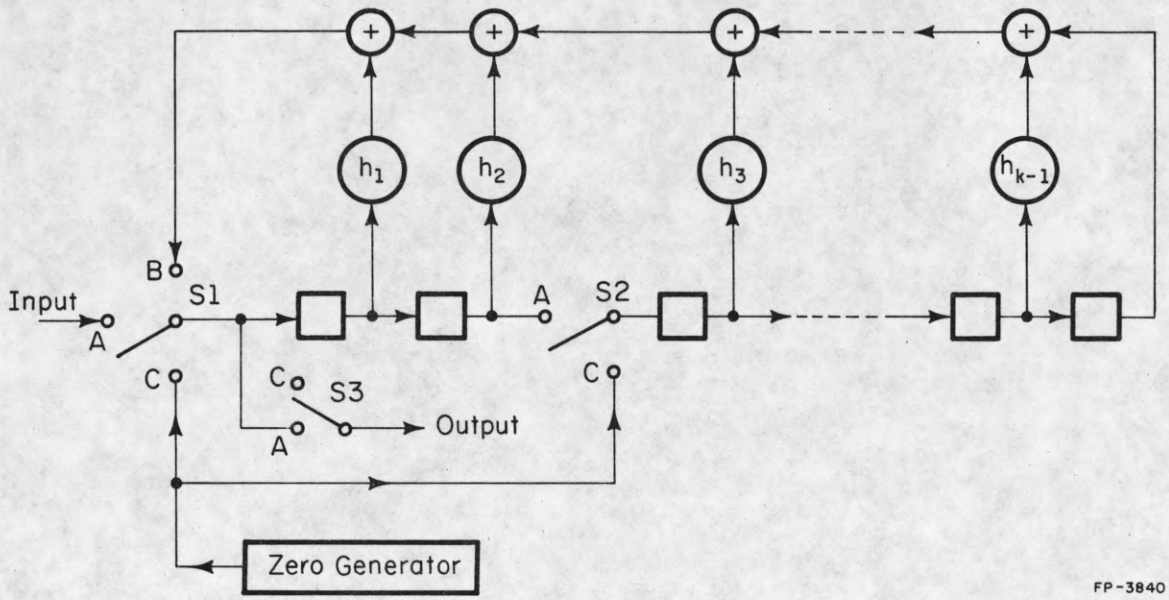

Fig. 3. Encoder for Shortened (82,21) Code

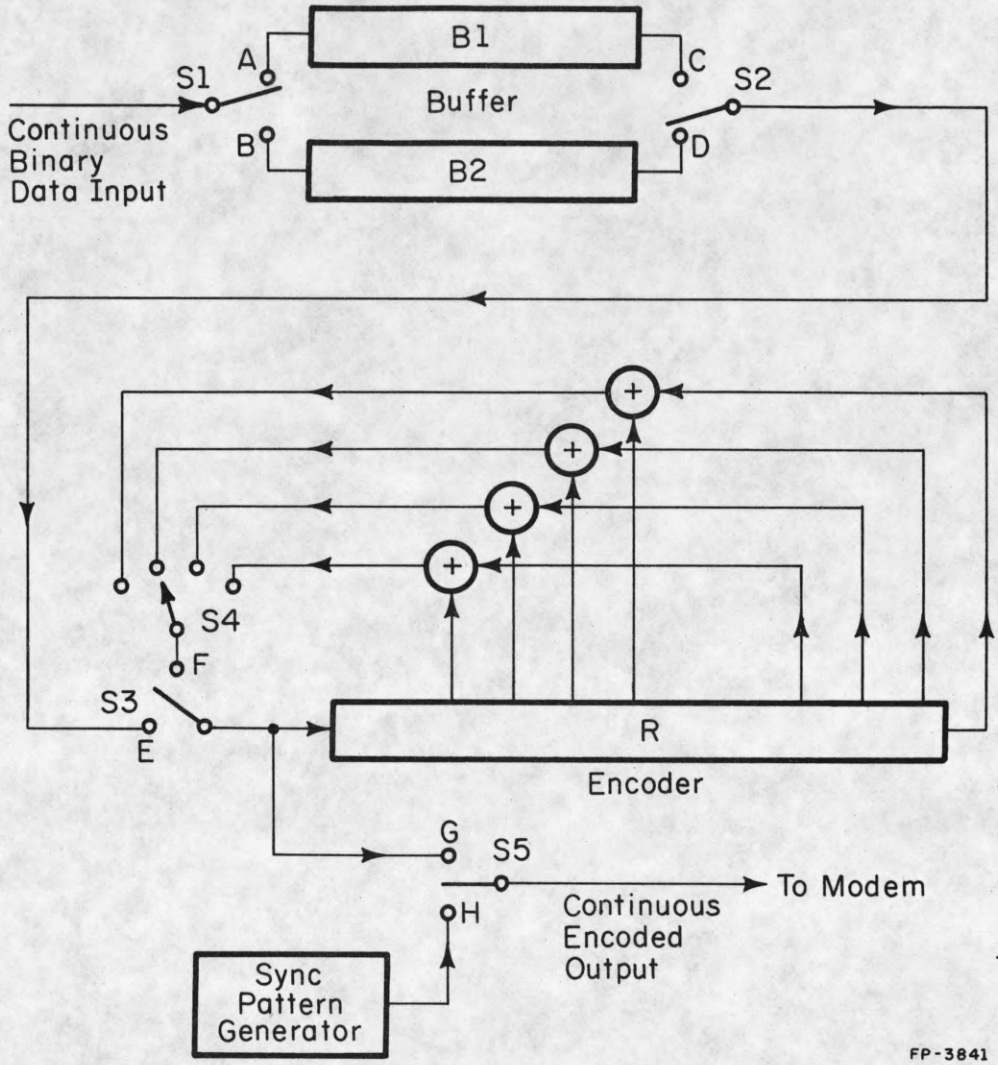

Fig. 4. Complete System Encoder

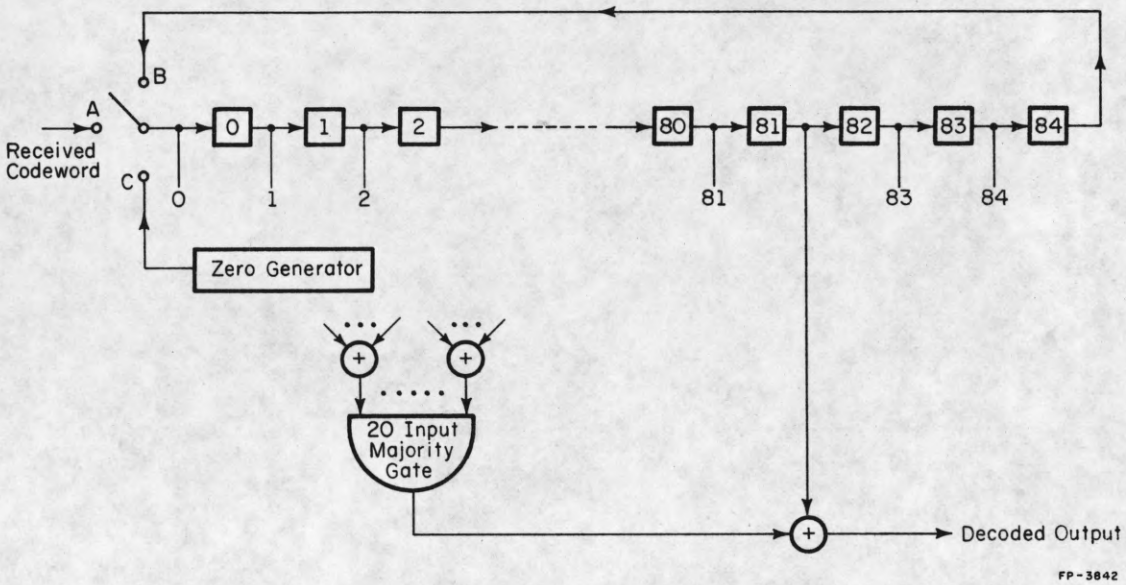

Fig. 5. Decoder for (82,21) Code

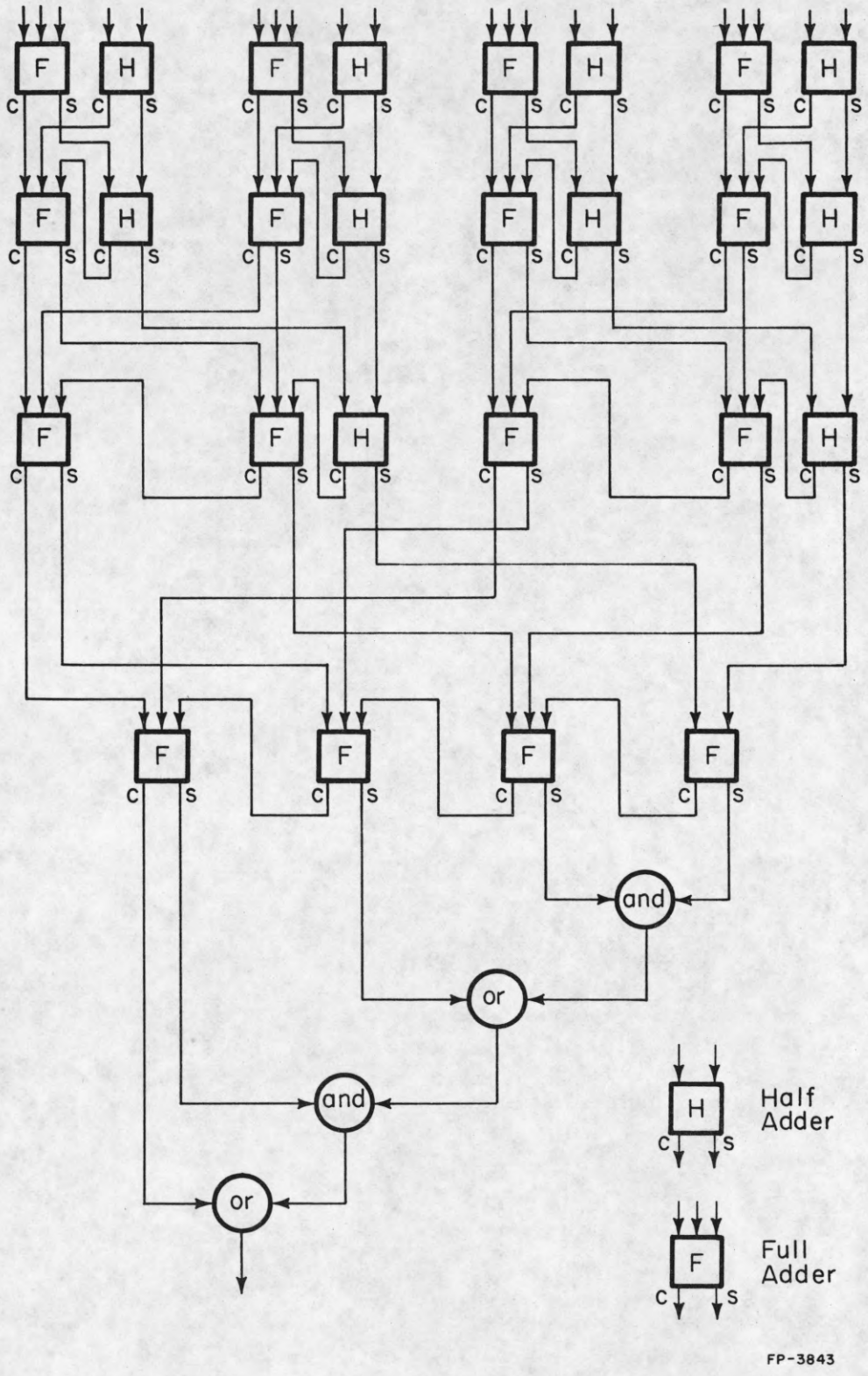

Fig. 6. 20 Inputs Majority Gate

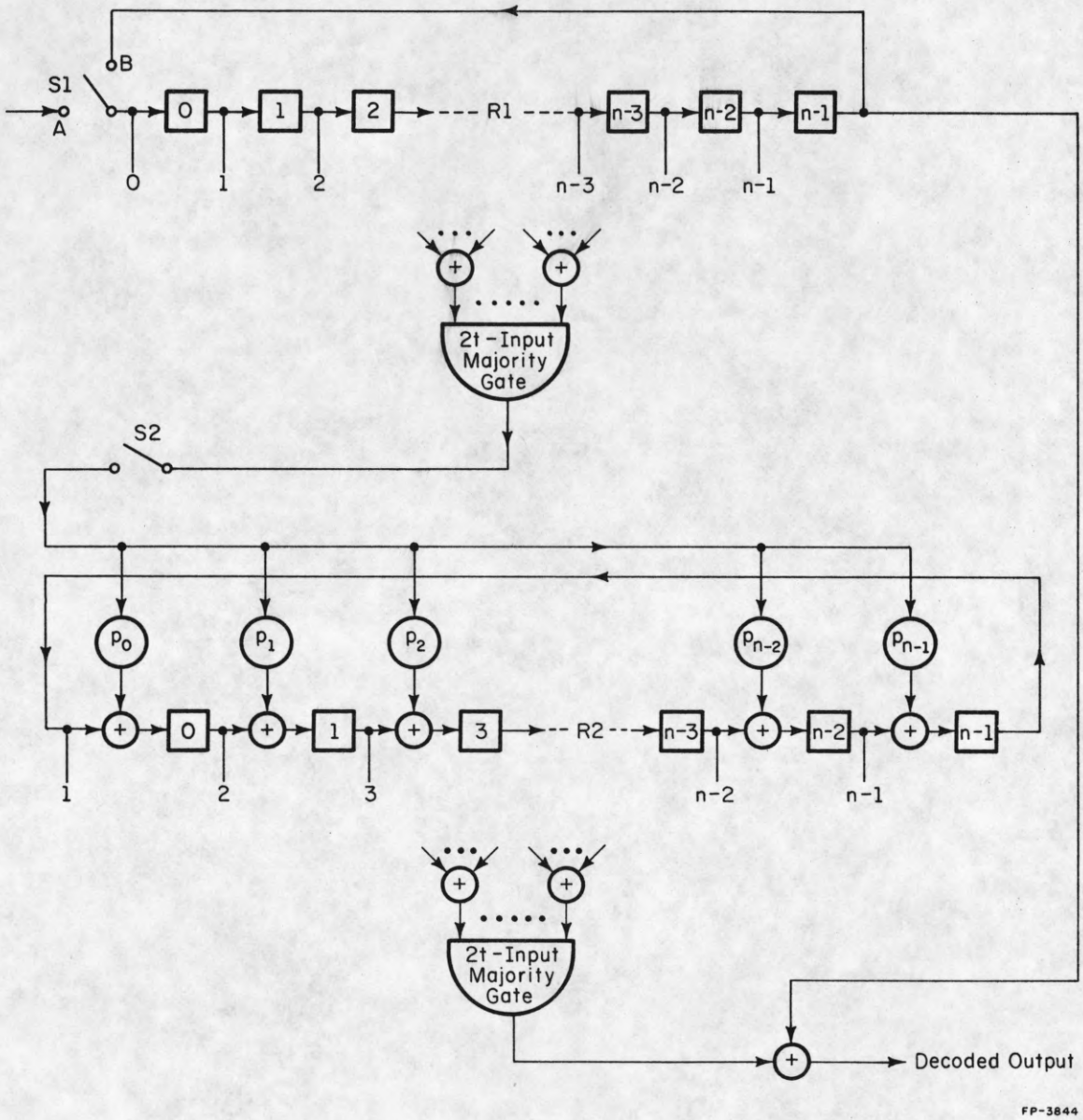

Fig. 7. General Decoder for Cyclic (n,k) t-error Correcting Code Requiring 2 Levels of Majority Decoding

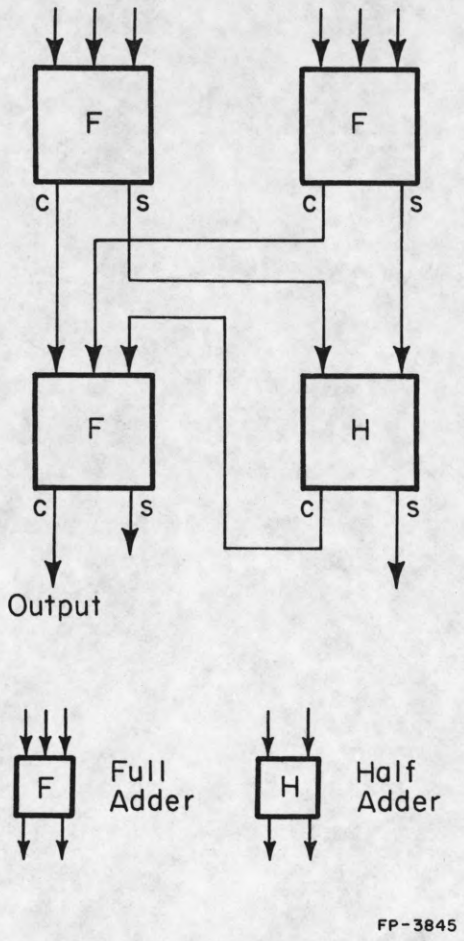

Fig. 8. 6 Input Majority Gate

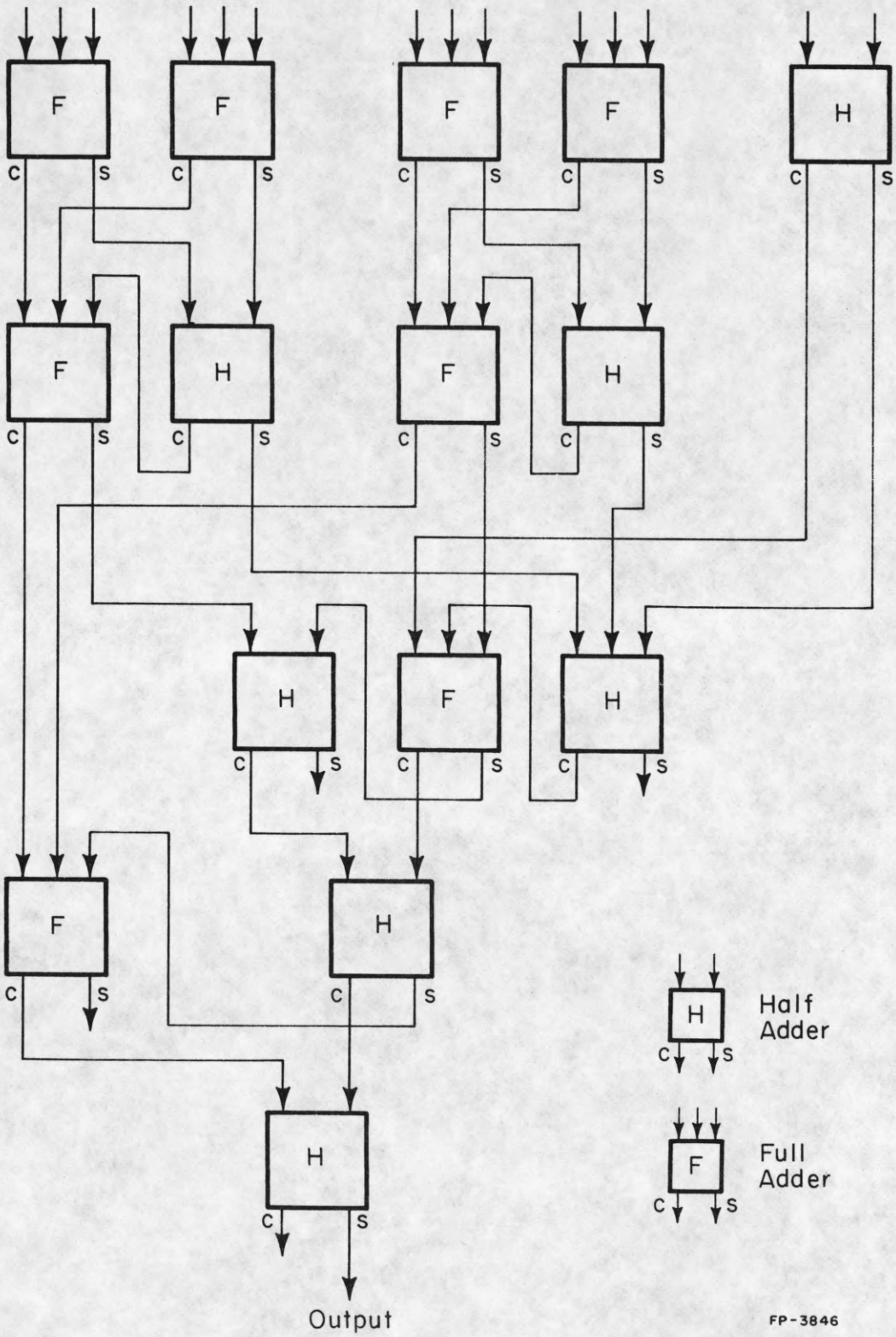

Fig. 9. 14 Inputs Majority Gate

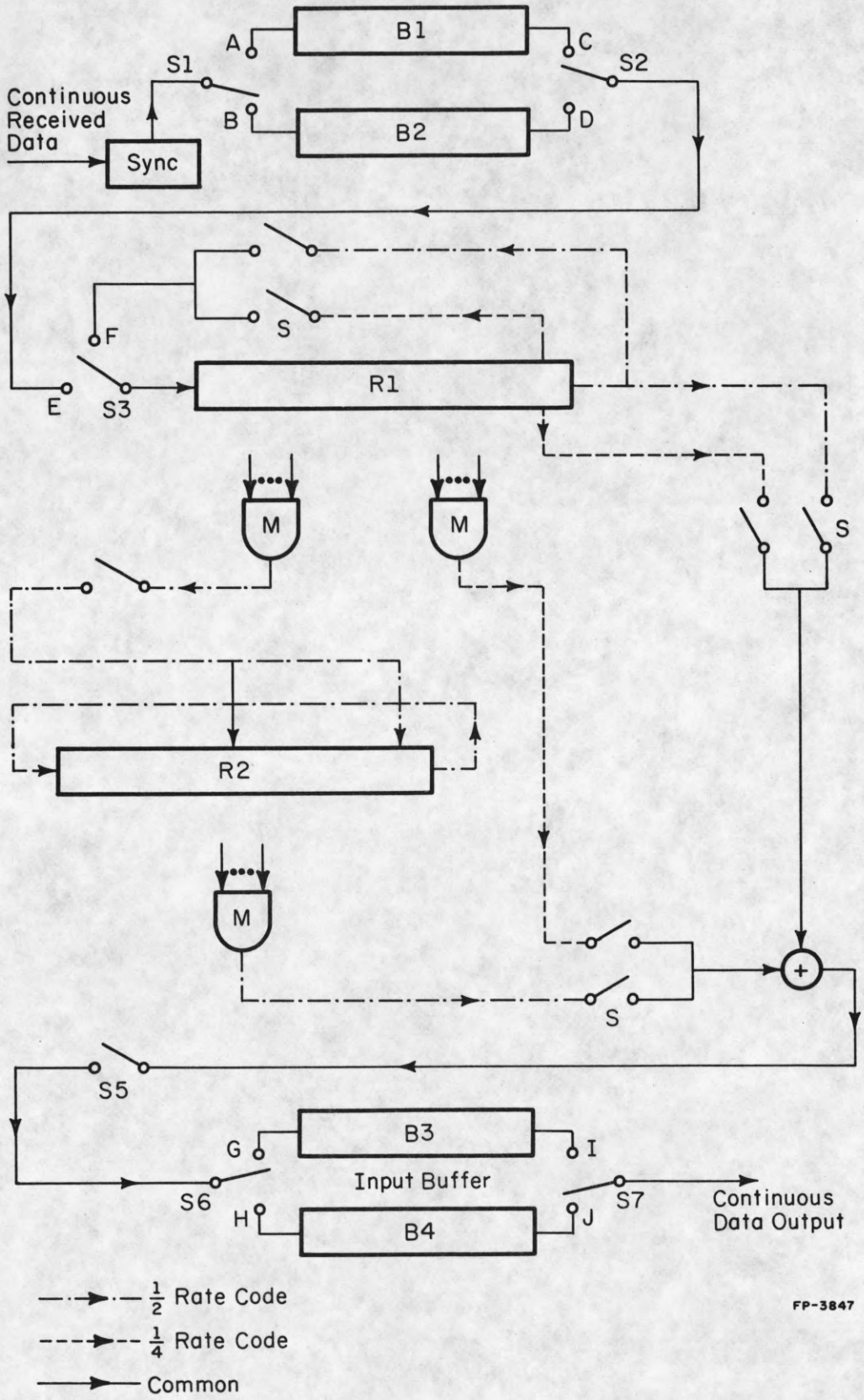

Fig. 10. Complete System Decoder

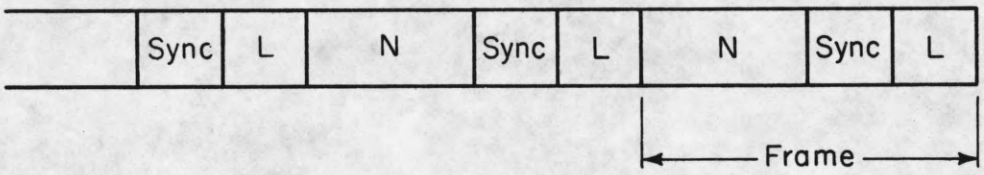

Fig. 11. Data Frames

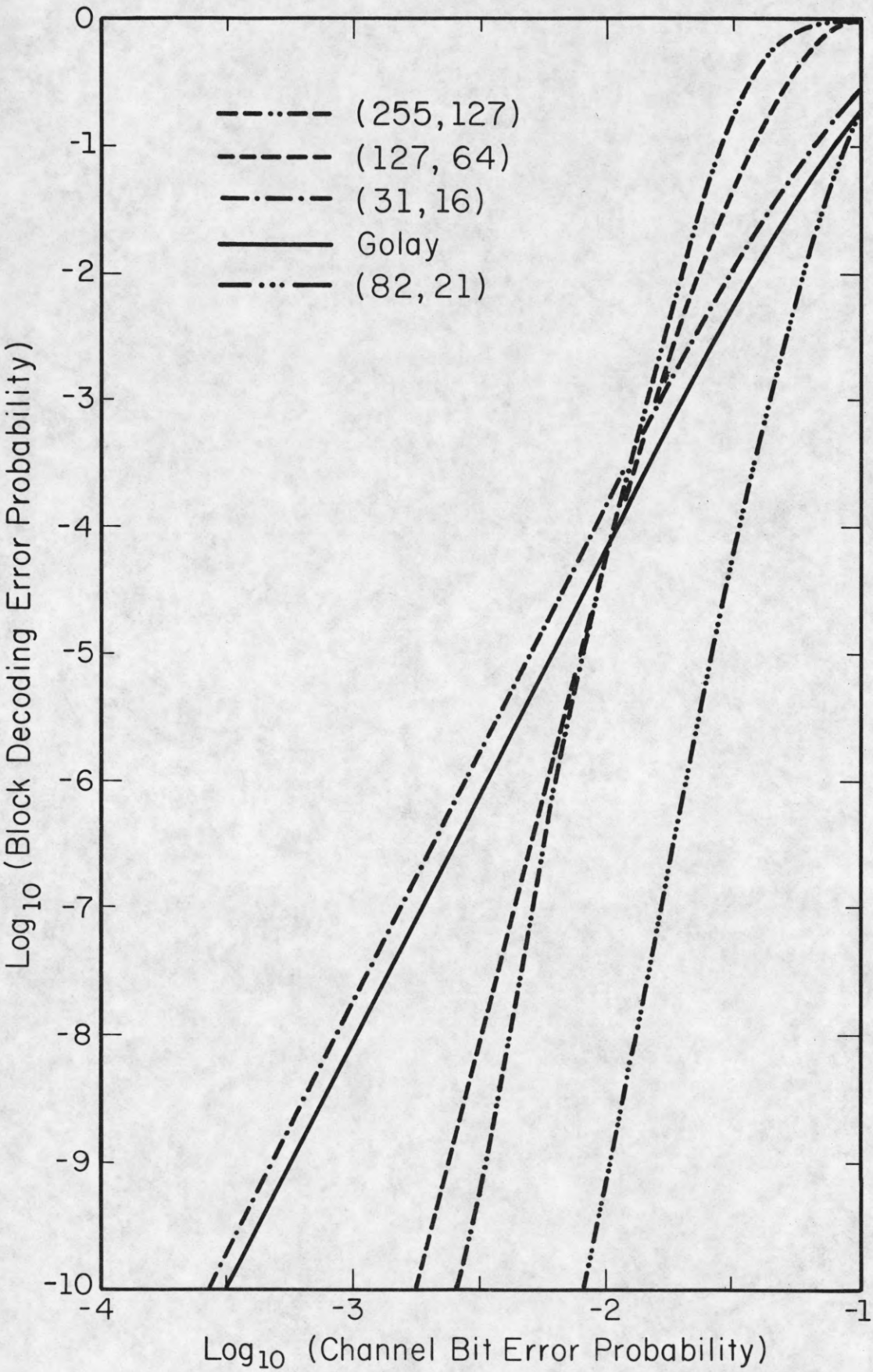

Fig. 12. Performance of Codes on BSC

FP-3849

REFERENCES

1. W. W. Peterson and E. J. Weldon, Jr., Error-Correcting Codes, Second Edition, MIT Press, Cambridge, Mass. 1972.

2. E. R. Berlekamp, Algebraic Coding Theory, McGraw-Hill, New York, 1968.

3. L. D. Rudolph, Geometric Configuration and Majority Logic Decodable Codes, MEG Thesis, University of Oklahoma, Norman, Oklahoma, 1964.

4. C. L. Chen, "Note on Majority-Logic Decoding of Finite Geometry Codes", IEEE Trans., IT-18, pp. 539-541, 1972.

5. L. D. Rudolph and C.R.P. Hartmann, "Sequential Code Reduction," IEEE Trans., IT-19, pp. 549-555, July 1973.

6. J. L. Maury, Jr., and F. J. Styles, "Development of Optimum Frame Sychronization Codes for Goddard Space Flight Center PCM Telemetry Standards," Proceedings of the NTC, 1964.

Security Classification

DOCUMENT CONTROL DATA - R & D

(Security classification of title, body of abstract and indexing annotation must be entered when the overall report is classified)

1. ORIGINATING ACTIVITY (Corporate author)	2a. REPORT SECURITY CLASSIFICATION
Coordinated Science Laboratory University of Illinois Urbana, Illinois 61801	UNCLASSIFIED 2b. GROUP

3. REPORT TITLE

AN ERROR-CONTROL SYSTEM BASED ON MAJORITY-LOGIC DECODING

4. DESCRIPTIVE NOTES *(Type of report and inclusive dates)*

5. AUTHOR(S) *(First name, middle initial, last name)*

R. B. Brown, C. L. Chen, R. T. Chien and S. Ng

6. REPORT DATE	7a. TOTAL NO. OF PAGES	7b. NO. OF REFS
February, 1974	35	6

8a. CONTRACT OR GRANT NO.	9a. ORIGINATOR'S REPORT NUMBER(S)
DAAB07-72-C-0259; F30602-72-C-0031 b. PROJECT NO.	R-640
c. d.	9b. OTHER REPORT NO(S) *(Any other numbers that may be assigned this report)* UILU-ENG 74-2203

10. DISTRIBUTION STATEMENT

Approved for public release. Distribution unlimited.

11. SUPPLEMENTARY NOTES	12. SPONSORING MILITARY ACTIVITY
	Joint Services Electronics Program through U.S. Army Electronics Command; Rome Air Development Center, Rome, New York

13. ABSTRACT

During the past two decades, many classes of error-correcting codes have been invented. However, these codes are rarely used in data transmission systems for combating channel noise. The reason for this is that the implementation of these codes is usually very complex and costly. From a practical point of view, the problem of finding efficient decoding algorithms that can be simply implemented is, therefore, the most important problem in coding applications today.

DD FORM 1 NOV 65 1473

Security Classification

14 KEY WORDS	LINK A		LINK B		LINK C	
	ROLE	WT	ROLE	WT	ROLE	WT
Majority-Logic Decoding						
Error Correction						
Finite Geometry Codes						