AD NUMBER

ADB007015

LIMITATION CHANGES

TO:

Approved for public release; distribution is unlimited.

FROM:

Distribution authorized to U.S. Gov't. agencies only; Test and Evaluation; DEC 1973. Other requests shall be referred to Air Force Armament Lab., Eglin AFB, FL 32542.

AUTHORITY

AFATL ltr 13 Oct 1976

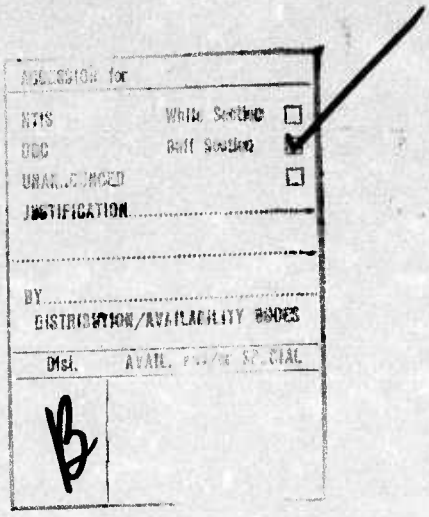
AFATL-TR-73-245

CODING FOR FREQUENCY-SHIFT-KEYED (FSK) COMMUNICATION SYSTEM

ELECTRICAL ENGINEERING DEPARTMENT
UNIVERSITY OF MISSOURI

TECHNICAL REPORT AFATL-TR-73-245
DECEMBER 1973

AIR FORCE ARMAMENT LABORATORY
AIR FORCE SYSTEMS COMMAND • UNITED STATES AIR FORCE
EGLIN AIR FORCE BASE, FLORIDA


ACCESSION for
NTIS White Section
DDC Buff Section
UNANNOUNCED
JUSTIFICATION
BY
DISTRIBUTION/AVAILABILITY CODES
Dist. AVAIL. and/or SPECIAL
B

Coding For Frequency-Shift-Keyed (FSK) Communication System

Dr. John J. Komo

FOREWORD

This report documents work performed during the period from January 1973 through December 1973 by the Electrical Engineering Department, University of Missouri, Columbia, Missouri, under Contract No. F08635-73-C-0078 with the Air Force Armament Laboratory, Eglin Air Force Base, Florida. Captain John A. Carnaghie (DLT) initiated the program, and Captain Charles W. Baker (DLMT) managed the program for the Armament Laboratory.

A note of thanks is extended to Captain Donald L. Steanson and Lieutenant Michael T. Corye, Air Force Institute of Technology, Civilian Institutions Division students at the University of Missouri for their assistance in coding analysis and circuit design and fabrication. A special acknowledgment is extended to Dr. James E. Ratke who provided leadership and assistance in the development and fabrication of the circuits.

This technical report has been reviewed and is approved.

JOHN W. JOHNSON
Deputy Chief, Guided Weapons Division

ABSTRACT

Coding schemes for the correction of random errors or burst errors for fixed block length digital data words are presented in this report. The word format considered is a 48-bit word with six 8-bit subwords, and the codes developed are for the correction of errors in two subwords with a third subword for the parity bits. Several codes are considered and analyzed for their error correcting capability. Also, computer simulations were carried out on several of these codes, and a burst length 5-error correcting coding system was mechanized. Performance gains based on comparisons with uncoded, coherent frequency-shift-keyed system are examined.

(The reverse of this page is blank)

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF FIGURES (CONCLUDED)

LIST OF TABLES

LIST OF TABLES (CONCLUDED)

(The reverse of this page is blank)

SECTION I

INTRODUCTION

In recent years the application of the theoretical aspects of coding theory has enhanced the attractiveness of digital communication systems. By using error control encoding and decoding, it is possible to achieve significant improvements in overall system performance, i.e., obtaining the same error performance as the uncoded system with less transmitter power or obtaining better error performance than the uncoded system with the same transmitter power. Noncyclic and cyclic type block codes are considered in this report. The amount of coding gain that is realizable for block codes depends on the particular code that is chosen.

A block code of length n bits with k information bits is denoted a (n,k) code; thus, the difference n-k is the number of parity bits. For random error-correcting codes the important consideration is the number of errors that can be corrected, and a code is classified as a t error-correcting code if it can correct all error patterns of t or fewer errors. For burst error-correcting codes the important consideration is the length of burst that can be corrected, and a code is classified as an ℓ burst error-correcting if it can correct all burst of errors of length ℓ or less. A burst of length ℓ is considered as being a group of bits of length ℓ, of which at least the first and the $\ell^{\underline{th}}$ bit are in error.

For a (48, 40) block code the best random error-correcting code that was obtained was a single error-correcting code, and the burst error-correcting code that was obtained was a burst 3 error-correcting code. It is possible in many cases to find shorter length codes with better error-correcting capabilities than longer codes with the same number of parity bits. Another consideration here was in obtaining better error control on 3 of the 8-bit subwords with a third for parity, leaving 3 of the subwords without any error protection. This leaves the form of the desirable code to be a (24, 16) code. For this format the best random error-correcting code that was obtained was a single error-correcting and double error-detecting code (in fact, this can be obtained with 2 more information bits,

i.e., (24,18) code) and the best burst error-correcting code that was obtained was a burst 3 error-correcting code which can be obtained with one more information bit, i.e., (24,17) code.

The actual format decided upon was the correction of 2 of the 8-bit subwords with a third subword along with 1 bit from each of the information subwords for parity or a (24,14) code format. This format enabled us to obtain a double error-correcting code and a burst 5 error-correcting code. Coding gains for the double error-correcting (24,14) code and the single error-correcting (48,40) code compared to the uncoded, coherent frequency-shift-keyed (FSK) system performance are presented. The (24,14) burst 5 error-correcting code has been mechanized in a cyclic code configuration with encoder, decoder, and simulator circuitry for demonstration and feasibility illustration of its error-correcting capabilities. The cyclic code configuration was chosen for its simpler parts and smaller number of connections as opposed to the general block code configuration.

SECTION II

FREQUENCY-SHIFT-KEYED COMMUNICATION SYSTEM

The simplest form of frequency-shift keying (FSK) is one with rectangular frequency modulation and constant carrier amplitude.[1] This may be ideally described as transmitting signal pulses of the form

$$s(t) = \begin{cases} A\cos 2\pi f_1 t \text{ , for a 0} \\ A\cos 2\pi f_2 t \text{ , for a 1} \end{cases} \qquad 0 \leq t < T \qquad (1)$$

where f_1 and f_2 are constant over a single information pulse. The most common method of transmitting the FSK signals for coherent detection is the selective gating of one of a pair of locked oscillators with frequency and phase stability. The coherent detection of the binary FSK signals is shown in Figure 1.

After the coherent detector multiplies by $\cos 2\pi f_1 t$ and $\cos 2\pi f_2 t$ and lowpass filters, the output of the first lowpass filter is the sum of the signal amplitude. A and the lowpass noise $x_1(t)$ and the second lowpass filter output is just the lowpass noise $x_2(t)$. The channel noise n(t) can equivalently be written as either of the two bandpass representations.

$$\begin{aligned} n(t) &= x_1(t)\cos 2\pi f_1 t - y_2(t)\sin 2\pi f_1 t \\ &= x_2(t)\cos 2\pi f_2 t - y_2(t)\sin 2\pi f_2 t \end{aligned} \qquad (2)$$

where $x_1(t)$, $y_1(t)$, $x_2(t)$, and $y_2(t)$ are all lowpass processes. Thus, a correct decision is made when $A + x_1(t)$ is greater than $x_2(t)$ or, equivalently, an error is made when $A + x_1(t)$ is less than $x_2(t)$. The normal assumption on n(t) is that it is a zero mean Gaussian random process with variance N which implies that $x_1(t)$ and $x_2(t)$ are zero mean Gaussian random processes with variance N. The probability of bit error is then given by

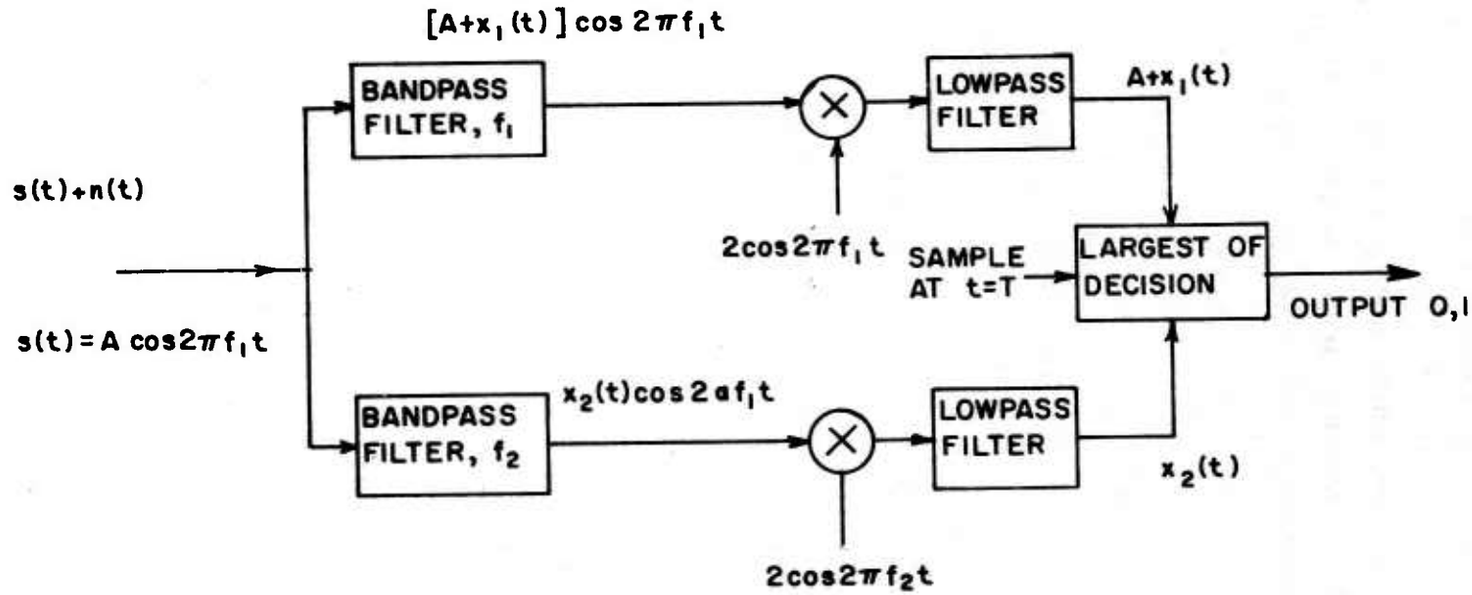


Figure 1. Coherent Detection of FSK, 0 Transmitted

$$P_b = \text{Prob}\ (A + x_1 < x_2) = \text{Prob}\ (x_2 - x_1 > A)$$
$$= \int_A^\infty \frac{1}{\sqrt{2\pi(2N)}} \exp[-\frac{u^2}{2(2N)}]\ du = Q\ (\frac{A}{\sqrt{2N}}) \tag{3}$$

where x_1 and x_2 are the samples of $x_1(t)$ and $x_2(t)$ at $t = T$ and $Q(\alpha)$ is the integral from α to ∞ of a zero mean unit variance Gaussian random variables. Similarly, for the noncoherent detection the probability of error is given as

$$P_b = \frac{1}{2} \exp\ [-A^2/4N] \tag{4}$$

As shown in Figure 2, there is very little difference in performance for large signal-to-noise ratio $A^2/2N$.

This signal-to-noise ratio is also conveniently written as E_b/N_0 where E_b is the energy per bit and N_0 is the noise spectral density.

To show the advantages of coding, it is desirable to compare the uncoded FSK system to several coded FSK systems. The system considered from now on will be the coherent FSK system. An uncoded FSK system is now compared, on a probability of information bit error basis, to an (n,k) t_c random error correcting code.

If there is a constraint on equal energy per information bit, then the energy per bit for the coded system becomes

$$E_b^c = \frac{k}{n} E_b \tag{5}$$

but if the energy is already available

$$E_b^c = E_b \tag{6}$$

The probability of bit error for the coded system is given by

$$P_b^1 = Q(\sqrt{E_b^c/N_0}\ ) \tag{7}$$

For the uncoded system a word error is made when 1 or more bits are in error, and the probability of word error is

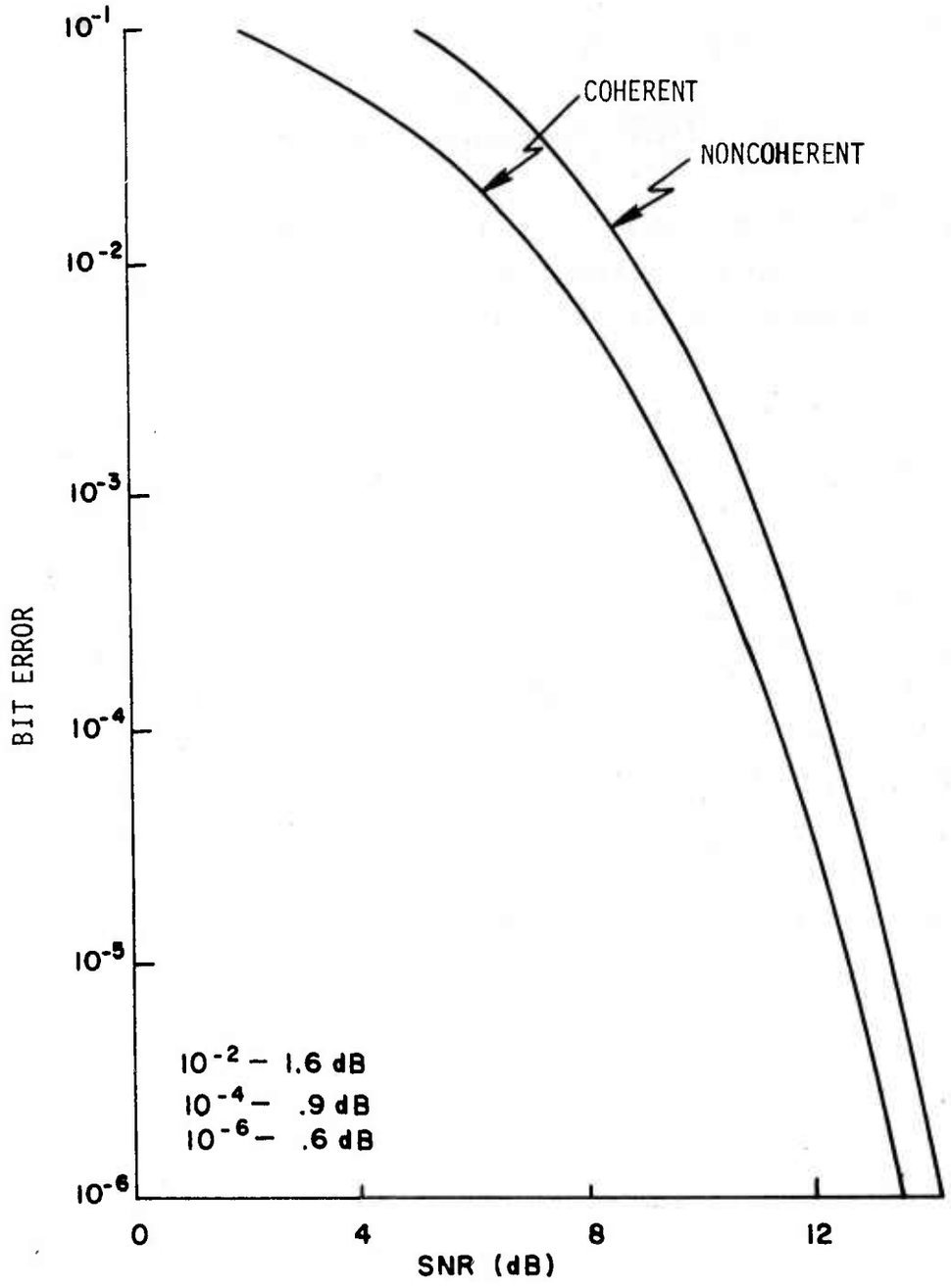


Figure 2. Coherent Versus Noncoherent FSK Comparison

$$P_w = \text{Prob (At least 1 bit error)}$$
$$= \text{1-Prob (No bit errors)} \tag{8}$$
$$= 1 - (1-P_b)^k$$

For the t_c error-correcting code a word error is not made unless $t_c + 1$ or more bits are in error. The probability of word error for the coded system is

$$P_w^c = \text{Prob (At least } t_c + 1 \text{ bit errors)}$$
$$= \text{1-Prob } (t_c \text{ or less bit errors})$$
$$= 1 - \left[\sum_{i=0}^{t_c} \binom{n}{i} (P_b^1)^i (1-P_b^1)^{n-i}\right] \tag{9}$$

This word error can also be written as a function of the information bit error for the coded system as

$$P_w^c = 1 - (1-P_b^c)^k \tag{10}$$

The desired relationship for P_b^c can now be obtained as

$$P_b^c = 1 - \left[\sum_{i=0}^{t_c} \binom{n}{i} (P_b^1)^i (1-P_b^1)^{n-i}\right]^{1/k} \tag{11}$$

Because of their different structure two random error-correcting codes were considered in detail. These were the (48,40) single error-correcting code and the (24,14) double error-correcting code. For the (48,40) code

$$P_b^c = 1 - \left[(1-P_b^1)^{48} + 48P_b^1(1-P_b^1)^{47}\right]^{1/40} \tag{12}$$

and for the (24,14) code

$$P_b^c = 1 - \left[(1-P_b^1)^{24} + 24P_b^1(1-P_b^1)^{23} + 276(P_b^1)^2 (1-P_b^1)^{22}\right]^{1/14} \tag{13}$$

Normally, the most meaningful comparison between an uncoded and a coded system is obtained using an equal energy per information bit

constraint [Equation (5)] . This is true since there is usually kE_b energy allocated for k information bits, and when the extra parity bits are added there is a total energy, kE_b, allocated now for n bits. Using this equal energy per information bit constraint, the uncoded, (48,40) coded, and (24,14) coded system performance is compared in Figure 3. Both coded systems perform better than the uncoded system, and the (24,14) code, as would be expected (2 more parity bits), performs better than the (48,40) code. As indicated in Figure 3, if it is desired to operate the system with a bit error probability of 10^{-6}, this can be done with 1.7 dB less power for the (48,40) coded system than the uncoded system and an additional 1.1 dB or a total of 2.8 dB less power is required for the (24,14) coded system over the coded system. Another comparison that should be made is that if a signal-to-noise ratio (SNR) of, for example, 10 dB is available to all 3 systems, the probability of bit error for the uncoded, the (48,40) coded, and the (24,14) coded systems are 7.8×10^{-4}, 1.0×10^{-4}, and 9.2×10^{-6} respectively. There is almost 2 orders of magnitude improvement in the bit error probability of the (24,14) coded system over the uncoded system at 10 dB SNR.

Even though the equal energy per information bit constraint comparison is normally the most meaningful, for the case at hand the structure of the word is such that the energy is available for the parity bits even if they are not used. This results in an additional gain of 0.8 dB [$10 \cdot \log(48/40)$] for the (48,40) coded system and 1.6 dB [$10 \cdot \log (24/14)$] for the (24,14) coded system. This comparison is shown in Figure 4. To achieve a bit error probability of 10^{-6} requires 2.5 dB and 4.4 dB less for the (48,40) coded and (24,14) coded systems, respectively, over the uncoded system. Also, if a SNR of 9 dB is present in all three systems, the probability of bit error for the uncoded, the (48,40) coded, and the (24,14) coded systems are 2.4×10^{-3}, 1.5×10^{-4}, and 1.8×10^{-6}, respectively. There is more than three orders of magnitude improvement in the bit error probability of the (24,14) coded system over the uncoded system at a SNR of 9 dB.

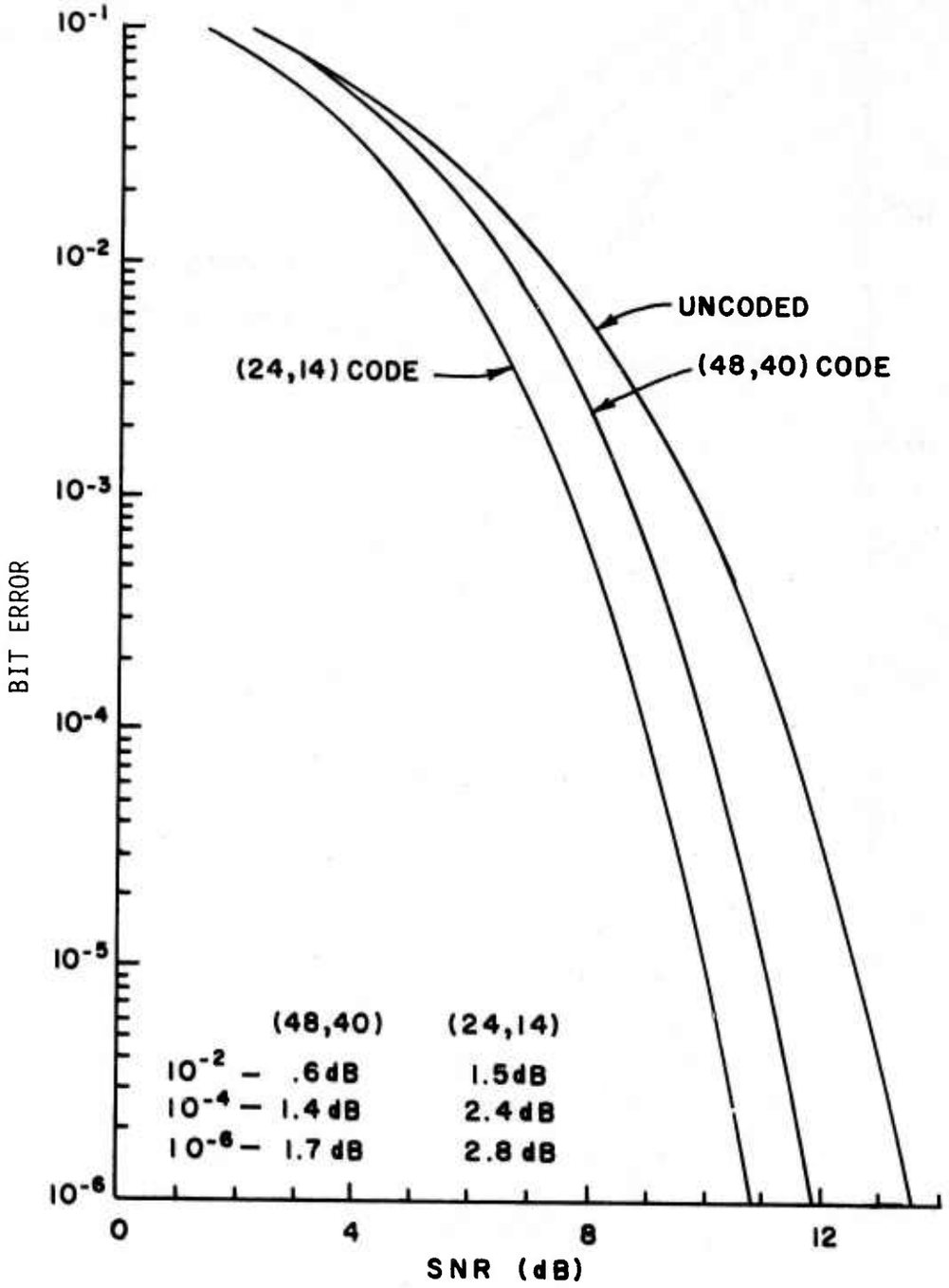


Figure 3. Coded Versus Uncoded Comparison ($E_b^c = \frac{k}{n} E_b$)

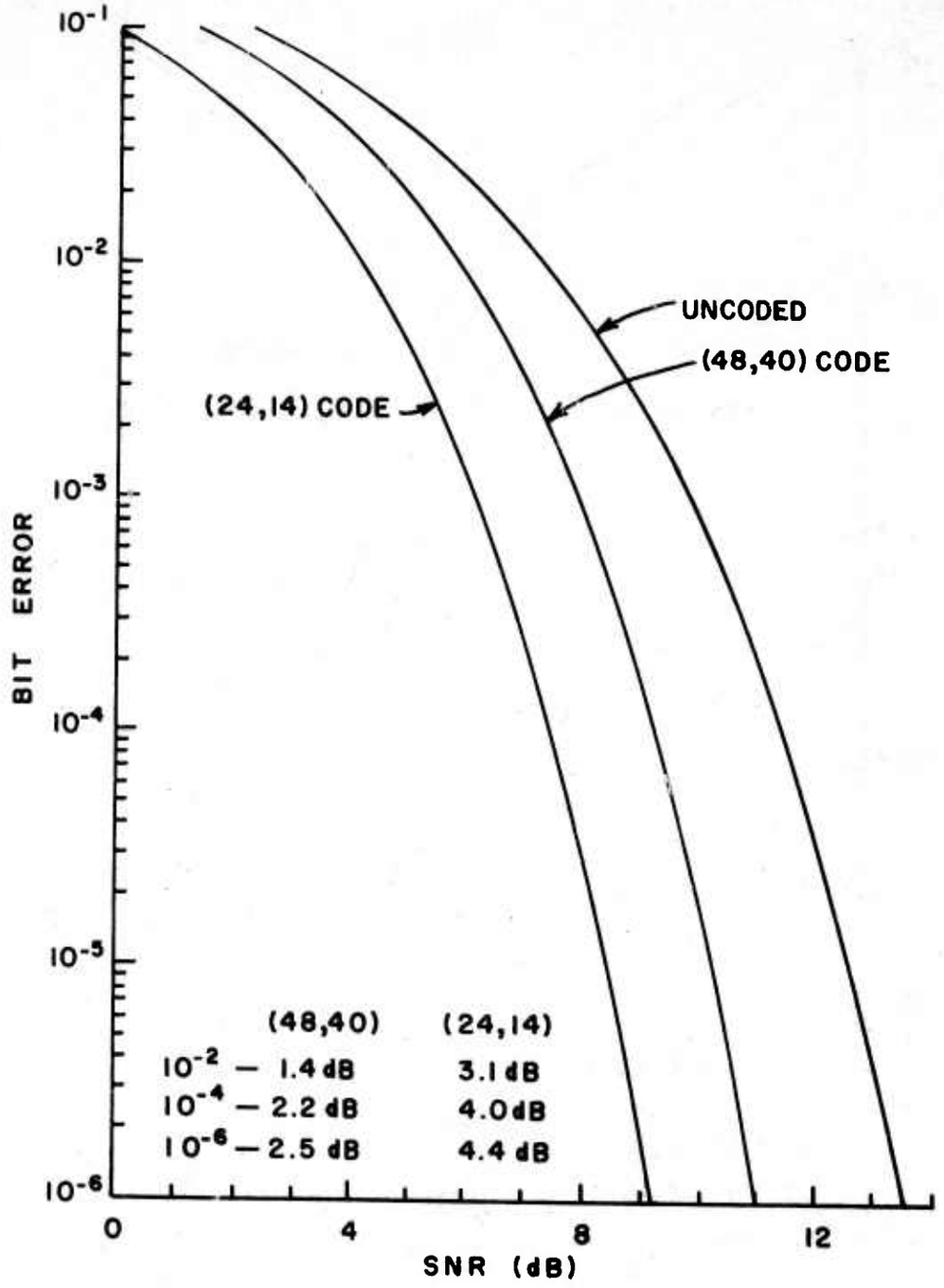


Figure 4. Coded Versus Uncoded Comparison ($E_b^c = E_b$)

SECTION III

CYCLIC CODES

For an (n,k) code, there are 2^k possible information messages. Corresponding to these information messages, 2^k of the total of 2^n n-tuples can be assigned as code words (encoding). There are 2^n-2^k non-zero errors that can be corrected, and since for each error pattern there are 2^k possible errors, the number of correctable non-zero error patterns are $2^{n-k}-1$. This set of 2^k code words is called a block code. A linear block code is a block code whose 2^k code words are a subspace of the vector space of all n-tuples; i.e., the all-zero vector is a code word and the sum of any two code words is also a code word.

Most of the recent research on linear block codes has been concentrated on a subclass known as cyclic codes. When referring to linear block codes which are not cyclic codes, they will be called non-cyclic block codes. Some non-cyclic block codes will be considered in Section VII.

Cyclic codes are attractive for two reasons:

(1) Encoding and syndrome (parity) calculations of a cyclic code can be implemented easily by employing simple shift registers with feedback connections.

(2) Since they have considerable inherent algebraic structure, it is possible to find various simple and efficient decoding methods.

An (n,k) linear block code C is called a cyclic code if it has the following property: If an n-tuple $v = (v_0, v_1, v_2,\ldots,v_{n-1})$ is a code vector of C, the n-tuple $v^{(1)} = (v_{n-1}, v_0, v_1,\ldots,v_{n-2})$ obtained by shifting v cyclically one place to the right is also a code vector of C. Because of this cyclic nature, it is more convenient to represent the vector as a polynomial.

$$v = (v_0, v_1, v_2,\ldots,v_{n-1}) \Leftrightarrow v(X) = v_0 + v_1X = v_2X^2 +\ldots+v_{n-1}X^{n-1} \qquad (14)$$

The terms code vector and code polynomial are used interchangeably. With polynomial representation, it is possible to develop some important properties for a cyclic code which make the simple implementation of encoding and syndrome calculation possible.

The components of the vector v and, likewise, the coefficients of the polynomial v(X), for consideration here, are taken to be 0 or 1. The symbols, 0 and 1, together with modulo 2 addition and normal multiplication, are known as a field of 2 elements (binary field or Galois field of 2 elements), and this field is denoted GF(2). The Galois field of 2^m elements, $GF(2^m)$, is obtained by starting with an irreducible polynomial f(X) of degree m with binary coefficients (not divisible by any polynomial of degree less than m and greater than zero). With 0, 1, α, α^2,..., α^{2m-2} denoting the 2^m elements, $f(\alpha)$ is set equal to 0 in a similar fashion to modulo 2 addition where 2 = 0. Using $f(\alpha) = 0$, all of the α^i can be represented as binary m-tuples. Tables I and II give $GF(2^4)$ for $f(X) = X^4 + X^3 + 1$ and $f(X) = X^4 + X^3 + X^2 + X + 1$, respectively. For example, from Table I, α^{11} is obtained as $\alpha^{11} = \alpha(\alpha^{10}) = \alpha(\alpha + \alpha^3) = \alpha^2 + \alpha^4 = \alpha^2 + 1 + \alpha^3 = 1 + \alpha^2 + \alpha^3$. An irreducible polynomial f(X) of degree m that gives a complete table with 2^m distinct m-tuples, as in Table I, is called a primitive polynomial.

In an (n,k) cyclic code, there exists one and only one code polynomial, g(X), of degree n-k,

$$g(X) = 1 + g_1X + g_2X^2 +...+ g_{n-k-1}X^{n-k-1} + X^{n-k} \tag{15}$$

Also, every code polynomial v(X) is a multiple of g(X), and every polynomial of degree n-1 or less which is a multiple of g(X) must be a code polynomial.

This allows a code polynomial to be expressed as:

$$v(X) = m(X)\, g(X) = (m_0 + m_1X +...+ m_{k-1}X^{k-1})\, g(X) \tag{16}$$

and the encoding of a message m(X) is equivalent to multiplying the message by g(X). An (n,k) cyclic code is completely specified by g(X), and g(X) is appropriately called the generator polynomial of the cyclic code. The degree n-k of g(X) is equal to the number of parity check digits of the code.

The generator polynomial, g(X), of an (n,k) cyclic code is a factor of $X^n + 1$, i.e.,

$$X^n + 1 = g(X)\, h(X) \tag{17}$$

where h(X) is called the parity check polynomial of the code, and if g(X) is

TABLE I. $GF(2^4)$ FOR $f(X) = X^4 + X^3 + 1$

	α^0	α^1	α^2	α^3		α^0	α^1	α^2	α^3
0	0	0	0	0	α^7	1	1	1	0
1	1	0	0	0	α^8	0	1	1	1
α	0	1	0	0	α^9	1	0	1	0
α^2	0	0	1	0	α^{10}	0	1	0	1
α^3	0	0	0	1	α^{11}	1	0	1	1
α^4	1	0	0	1	α^{12}	1	1	0	0
α^5	1	1	0	1	α^{13}	0	1	1	0
α^6	1	1	1	1	α^{14}	0	0	1	1
					α^{15}	1	0	0	0

TABLE II. $GF(2^4)$ FOR $f(X) = X^4 + X^3 + X^2 + X + 1$

	α^0	α^1	α^2	α^3		α^0	α^1	α^2	α^3
0	0	0	0	0	α^7	0	0	1	0
1	1	0	0	0	α^8	0	0	0	1
α	0	1	0	0	α^9	1	1	1	1
α^2	0	0	1	0	α^{10}	1	0	0	0
α^3	0	0	0	1	α^{11}	0	1	0	0
α^4	1	1	1	1	α^{12}	0	0	1	0
α^5	1	0	0	0	α^{13}	0	0	0	1
α^6	0	1	0	0	α^{14}	1	1	1	1
					α^{15}	1	0	0	0

a polynomial of degree n-k and is a factor of $X^n + 1$, then g(X) generates an (n,k) cyclic code. This result enables generating polynomials of cyclic codes to be obtained by factoring $X^n + 1$.

The form of v(X) in Equation (16) is acceptable, but it has the message bits intermixed with the parity bits, i.e.,

$$v(X) = m_0 + (m_0 g_1 + m_1)X + (m_0 g_2 + m_1 g_1 + m_2)X^2$$

$$+ \ldots + (m_{k-1} g_{n-1} + m_{k-2})X^{n-2} + m_{k-1}X^{n-1} \qquad (18)$$

It is desirable to have the message bits segregated from the parity bits in the code polynomial (systematic form). Dividing $X^{n-k}m(X)$ by g(X) yields

$$X^{n-k}\, m(X) = g(X)\, q(X) + p(X) \qquad (19)$$

where p(X) is of degree n-k-1 or less. Thus,

$$v(X) = g(X)\, q(X) = p(X) + X^{n-k}\, m(X)$$

$$= p_0 + p_1 X + \ldots + p_{n-k-1}X^{n-k-1} + m_0 X^{n-k} + m_1 X^{n-k+1} + \ldots$$

$$+ m_{k-1}X^{n-1} \qquad (20)$$

and the message bits have been segregated to the higher order positions. The realization of an encoder [Equation (20)] then requires a circuit to multiply the message by X^{n-k} and a circuit to divide this product by g(X) and retain the remainder.

Since in this system all of the addition is modulo 2 there are no carries from lower order bits to higher order bits. A normal binary adder is implemented by adding the multiplicand (if the least significant digit of the multiplier is 1) to the multiplicand shifted one digit (if the next digit of the multiplier is 1) to etc. until the most significant digit of the multiplier is used. This can be accomplished in a serial manner by serially feeding the multiplicand into a shift register through binary adders at points along the register where ones occur in the multiplier. For example, consider the multiplication of $d_0 + d_1 X + \ldots + d_i X^i$ by $f_0 + f_1 X + \ldots + f_j X^j$

which is shown in Figure 5. The d's are serially fed into the shift register starting with d_i. The circles represent a connection if $f_k = 1$ or no connection if $f_k = 0$. This is not well suited for binary multiplication because of the forward carries, but for this case there are no carries because of the modulo 2 addition and the binary adders can be replaced by exclusive or gates. This multiplication for modulo 2 arithmetic is given in Figure 6. The output of the highest order flip-flop is the coefficient of X^{j+k} when d_k is on the input. For binary division, the adders of Figure 5 would be replaced by subtractors and the d's fed into the first flip-flop through an additional subtractor. The output of the highest order flip-flop (which is a 1 when the shifted part of the dividend is greater than or equal to j) is fed back to the f's for subtraction. After all the d's are serially fed in, the contents of the flip-flops is the remainder and the quotient is the portion that has been shifted out of the flip-flops. For modulo 2, arithmetic subtraction is identical to addition, and this division for dividing d(X) by f(X) is shown in Figure 7. A circuit for simultaneously multiplying by $1 + X + X^5$ and dividing by $1 + X^3 + X^4 + X^5 + X^6$ is given in Figure 8. Table III shows the contents of the 6-bit register of Figure 8 for $d(X) = 1 + X^2 + X^4 + X^6$. It can be observed from Table III that the

TABLE III. $(1 + X + X^5)d(X)$ DIVIDED BY $1 + X^3 + X^4 + X^5 + X^6$

	d(X)	6-bit register						
X^6	1	1	1	0	0	0	1	
X^5	0	1	1	1	1	1	1	
X^4	1	0	0	1	0	0	1	
X^3	0	1	0	0	0	1	1	← quotient
X^2	1	0	0	0	1	1	1	
X	0	1	0	0	1	0	0	
1	1	1	0	0	0	1	1	← remainder

quotient from the division (previous outputs from the register) is $X + X^2 + X^3 + X^4 + X^5$, and the remainder (contents of the register is $1 + X^4 + X^5$.

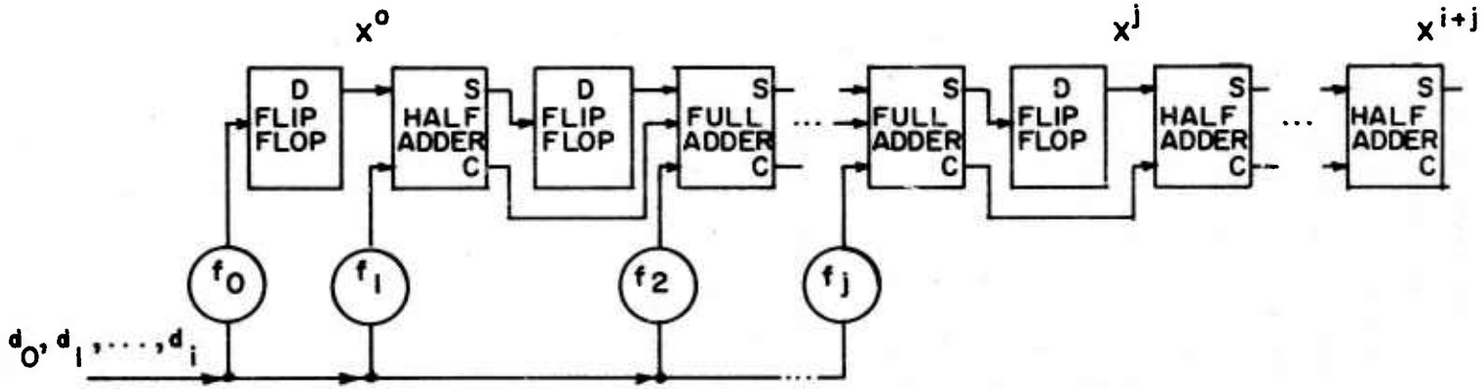


Figure 5. Binary Multiplication of d(X) by f(X)

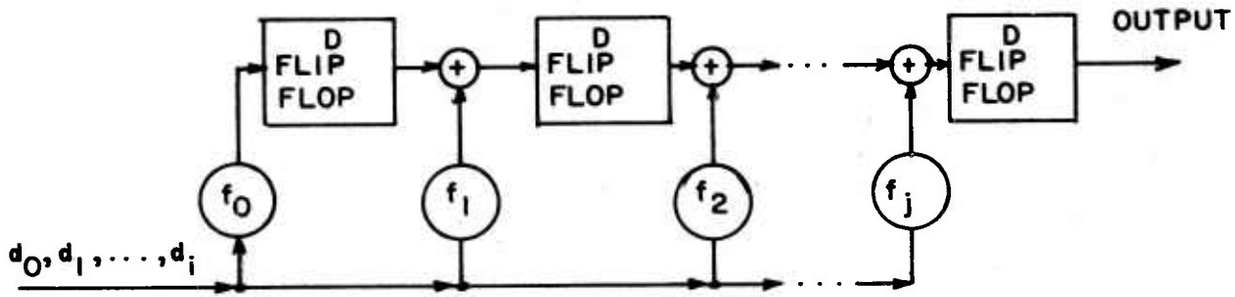


Figure 6. Modulo 2 Multiplication of d(X) by f(X)

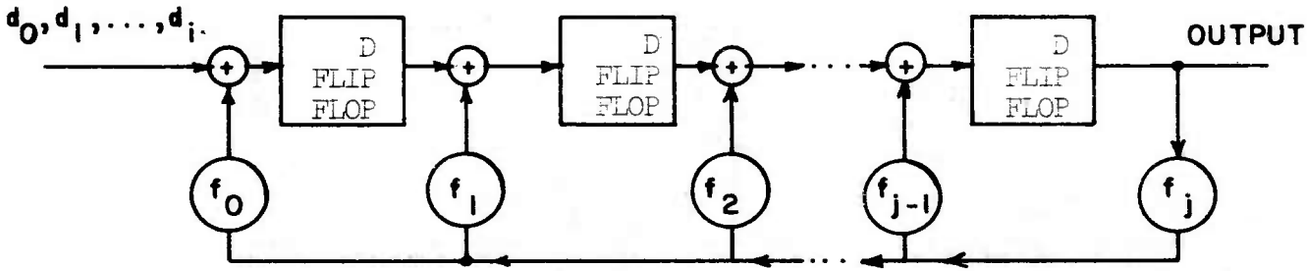


Figure 7. Modulo 2 Division of d(X) by f(X)

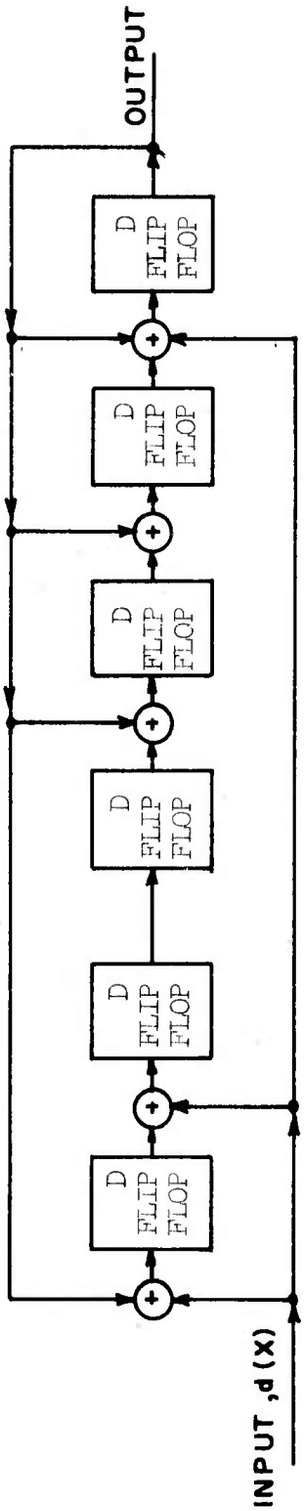


Figure 8. $(1 + X + X^5)d(X)$ Divided by $(1 + X^3 + X^4 + X^5 + X^6)$

The encoder [multiply m(X) by X^{n-k} and divide by g(X)] then takes the form shown in Figure 9. With the Gate on and the switch in the lower position, all k information bits are simultaneously fed to the output and fed back to the parity generator. After the k bits are clocked in, the Gate is turned off and the switch is moved to the upper position. The contents (n-k bits) of the parity register are then serially shifted to the output.

In the decoder, if there are no errors made, the remainder from dividing r(X) (the received signal) by g(X) should be zero, since v(X) is a multiple of g(X). When there are errors, the remainder (syndrome) is nonzero, and an association can be made between the syndrome and the error pattern. The syndrome is obtained from r(X) as

$$r(X) = g(X)q(X) + S(X) \tag{21}$$

Thus, to obtain the syndrome, all that is needed is a circuit to divide by g(X), which makes the syndrome calculation of the decoder identical to the encoder. The decoder takes the form shown in Figure 10. After the syndrome calculation, all that the decoder is left to do is make the association of the syndrome with the correct error pattern. This association is called the error pattern calculation, and it can be realized with a combinatorial logic circuit. The easiest way to obtain the error pattern calculation for single random, burst, and some multiple random error-correcting codes is to use an error-trapping technique.

The fact that if a cyclic decoder can decode the highest order bit in a word correctly for all correctable error patterns, then the entire word can be decoded with the same circuitry gives rise to the simplicity of cyclic decoding. Thus, for an error-trapping decoder, the objective is to trap the error patterns in the syndrome register. For a single random error-correcting code the error-trapping decoder tests for an error in the highest order syndrome bit by testing for all zeros in the other bits, and for a burst length ℓ error-correcting code the error-trapping decoder test for an error pattern in the ℓ highest order syndrome bits by testing for all zeros in the other bits. For a multiple random error-correcting code the error-trapping decoder, in general, has to test for some of the errors

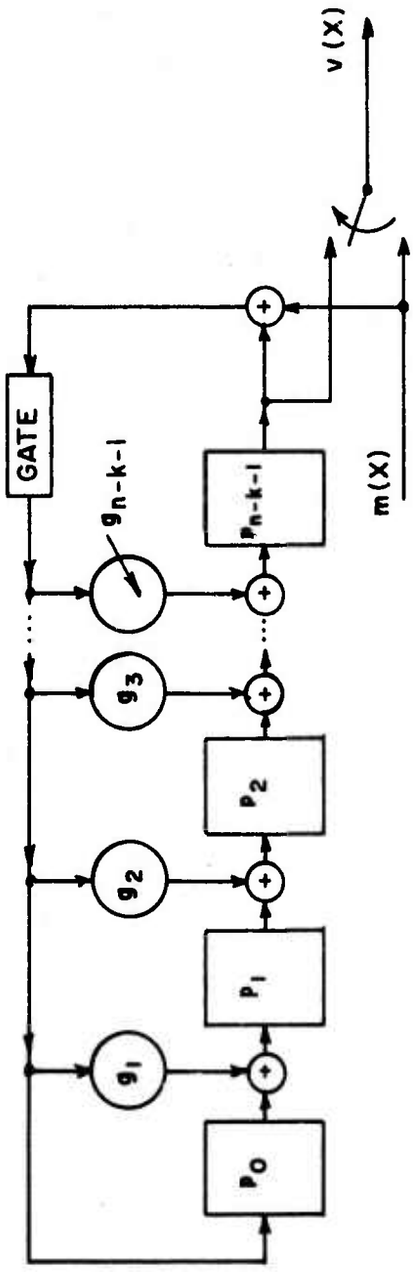


Figure 9. General Cyclic Encoder

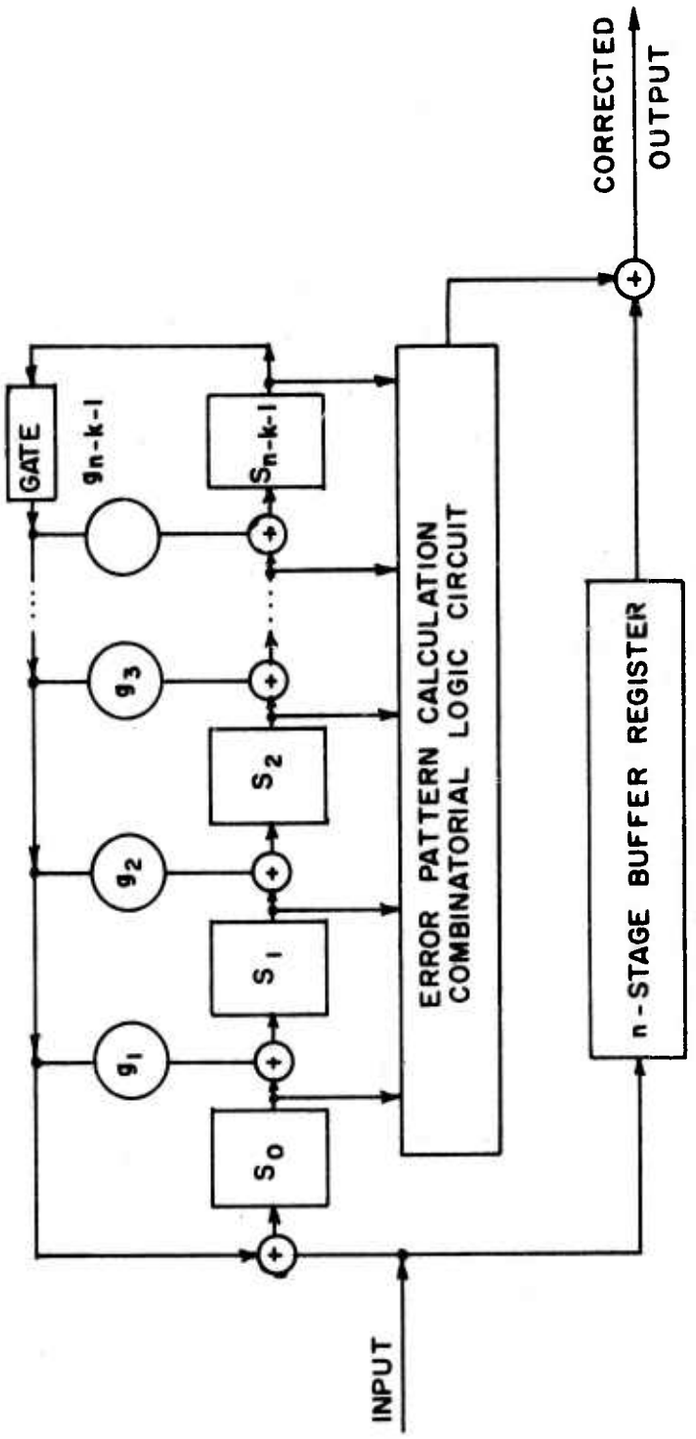


Figure 10. General Cyclic Decoder

trapped in the syndrome register and others trapped at specific locations outside the syndrome register. The complexity of an error-trapping decoder for multiple random errors is a function of the specific code, and a decision on the selection of an error-trapping decoder or some other decoder has to be made on an individual code basis.

Error trapping decoders for random errors are considered in more detail in Section IV, and error-trapping decoders for burst errors are considered in Section V.

SECTION IV

RANDOM ERROR-CORRECTING CODES

If transmission errors occur independently, i.e., if each transmitted symbol is affected independently by noise, they are classified as random errors. For random errors the probability of having i errors is less than the probability of having i-1 errors. Because of this fact, for random error-correcting codes it is desirable to be able to correct all single error patterns, then all double error patterns, etc., in a progressively increasing manner. This leads to the classification of a t error-correcting code as a code that corrects all error patterns of t or fewer errors. A t error-correcting (n,k) code is called a perfect code if it satisfies

$$\sum_{i=0}^{t} \binom{n}{i} = 2^{n-k} \tag{22}$$

and a quasi-perfect code if it satisfies

$$\sum_{i-0}^{t} \binom{n}{i} + N_{t+1} = 2^{n-k} \tag{23}$$

where

$$0 < N_{t+1} < \binom{n}{t+1} \tag{24}$$

For an (n, k) code the total number of correctable error patterns (including the zero error pattern) is 2^{n-k}, and the number of error patterns containing i errors is $\binom{n}{i}$. Thus, for random error-correcting codes it is desirable to obtain perfect or quasi-perfect codes when they exist, and, in general, to obtain a t as large as possible for a given n and k.

A (n, k) cyclic Hamming code is a code whose generator polynomial is a primitive polynomial of degree n-k, and n-k can be any positive integer. A Hamming code is a single error-correcting code and n equals $2^{n-k} - 1$ since the generator polynomial is a primitive polynomial. This length

satisfies Equation (22) with t = 1; thus, the Hamming codes are perfect codes. The only known multiple error-correcting binary perfect code is the Golay (23, 12) triple error correcting code.

Since in this report the interest was in finding codes with the highest error-correcting capability for a fixed number of parity bits (fixed n-k), Table IV lists codes for n-k equal to 6 through 10 along with their minimum distance, d, error-correcting capability, t_c, and error-detecting capability, t_d. t_c plus t_d can be expressed as d-1. The codes of Table IV are derived from Peterson Table (2), but, could be developed directly from factoring $X^n + 1$. The code lengths between the one given in Table IV can be filled in by using the fact that a shortened (n-η, k-η) code with the same error-correcting capabilities can be obtained from a (n,k) code by eliminating the m high order information bits. This (n-η, k-η) code is called a shortened cyclic code. For example, with 8 bits available for parity, it is seen that if the desired length of the code is greater than 17 only a single error-correcting code can be obtained, but if the desired length is 17 or less, a double error-correcting code can be obtained.

The encoding for random error-correcting codes is accomplished with the general cyclic encoder of Figure 9. The decoding is accomplished with the general cyclic decoder of Figure 10 with the error-trapping calculation left to be determined. The error-trapping calculation necessary for the decoding of single errors and the calculation necessary for the decoding of multiple errors will now be obtained.

Error-Trapping Decoding for Single Errors

Error-decoding consists of cyclic shifting of the received vector until the errors are confined to the n-k parity positions (syndrome registers) of the (n,k) cyclic code. If a cyclic decoder can decode the first (highest order) symbol in a word correctly for all correctable error patterns, then the entire word can be decoded with the same circuitry. The form of the decoder is given in Figure 11. The connections to the AND gates are determined by finding the syndrome corresponding to an error in the higest bit position.

TABLE IV. RANDOM ERROR-CORRECTING CODES

n-k	Code (n,k)	d	t_c	t_d	Generator Polynomial $X^{10} \ldots X^5 \ldots X^0$
6	(9,3)	3	1	1	1 0 0 1 0 0 1
	(15,9)	4	1	2	1 0 1 1 1 0 1
	(21,15)	4	1	2	1 1 0 0 1 0 1
	(31,25)	4	1	2	1 1 0 1 1 1 1
	(63,57)	3	1	1	1 0 0 0 0 1 1
7	(15,8)	4	1	2	1 1 1 0 0 1 1 1
	(21,14)	4	1	2	1 1 1 1 1 0 0 1
	(35,28)	4	1	2	1 1 0 1 1 0 0 1
	(63,56)	4	1	2	1 1 0 0 0 1 0 1
8	(15,7)	5	2	2	1 1 1 0 1 0 0 0 1
	(17,9)	5	2	2	1 1 1 0 1 0 1 1 1
	(21,13)	4	1	2	1 0 1 1 1 1 1 0 1
	(35,27)	4	1	2	1 0 1 1 0 1 0 1 1
	(51,43)	3	1	1	1 1 1 1 1 0 0 1 1
	(63,55)	3	1	1	1 1 1 0 0 1 0 0 1
9	(15,6)	6	2	3	1 0 0 1 1 1 0 0 1 1
	(17,8)	6	2	3	1 0 0 1 1 1 1 0 0 1
	(21,12)	5	2	2	1 1 1 0 1 1 0 0 1 1
	(51,42)	4	1	2	1 0 0 0 0 1 0 1 0 1
	(63,54)	4	1	2	1 0 0 1 0 1 1 0 1 1
10	(15,5)	7	3	3	1 0 1 0 0 1 1 0 1 1 1
	(21,11)	6	2	3	1 0 0 1 1 0 1 0 1 0 1
	(31,21)	5	2	2	1 1 1 0 1 1 0 1 0 0 1
	(33,23)	3	1	1	1 1 0 0 0 1 0 0 0 1 1
	(35,25)	4	1	2	1 0 1 0 1 1 1 0 1 0 1
	(45,35)	4	1	2	1 0 0 0 1 0 0 1 0 1 1
	(51,41)	4	1	2	1 0 1 0 0 1 0 0 1 0 1
	(63,53)	4	1	2	1 0 1 1 1 1 1 1 0 0 1

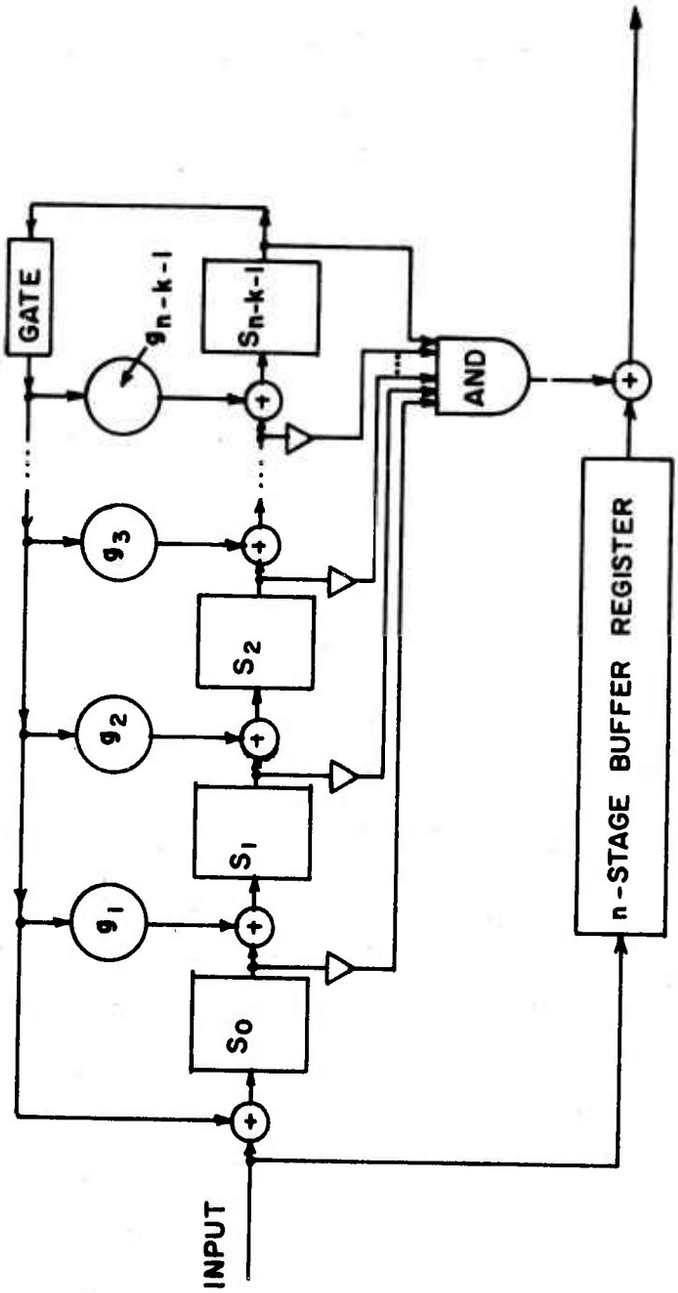


Figure 11. General Single Error-Trapping Decoder

$$e_1(X) = X^{n-1} \tag{25}$$

When this syndrome is detected, the next bit out of the register will be corrected. The syndrome $S_1(X)$ is determined as the remainder of dividing $e(X)$ by $g(X)$ or

$$e_1(X) = g(X)\, q(X) + S_1(X) \tag{26}$$

Since the remainder of dividing X^n by $g(X)$ is 1, $g(X)$ plus a single shift on $S_1(X)$ must equal 1 or

$$g(X) + X\, S_1(X) = 1 \tag{27}$$

Thus, in general form, for

$$g(X) = 1 + g_1X + g_2X^2 + g_3X^3 + \ldots + g_{n-k-1}\, X^{n-k-1} + X^{n-k} \tag{28}$$

the syndrome equals

$$S_1(X) = g_1 + g_2X + g_3X^2 + \ldots + g_{n-k-1}\, X^{n-k-2} + X^{n-k-1} \tag{29}$$

The input to the AND gate test, for the presence of $S_1(X)$ from the $i^{\underline{th}}$ syndrome register, is S_i if g_{i+1} equals 1 or $\overline{S}_i$ if g_{i+1} equals 0 or

$$g_{i+1}\, S_i + \overline{g}_{i+1}\, S_i + 1 \tag{30}$$

From the above analysis, it is obvious that the connections to the AND gate change with different $g(X)$.

It would be desirable to have the same connections to the AND gate for all possible $g(X)$. The connections can, in fact, be made independent of $g(X)$.

For $e_1(X)$ given in Equation (25) look at

$$X^{n-k}e_1(X) = X^{n-k}\,X^{n-1} = X^{n-k-1}\,(X^n+1) + X^{n-k-1} \tag{31}$$

Dividing by $g(X)$ yields

$$X^{n-k}e_1(X) = [X^{n-k-1}h(X)]\,g(X) + X^{n-k-1} \tag{32}$$

and the remainder is

$$S(X) = X^{n-k-1} \tag{33}$$

Now when a 1 in S_{n-k-1} and 0 in all the other S's is detected, the next bit out of the register will be corrected. To obtain the error to be trapped in the highest syndrome position, multiply the input by X^{n-k} which is equivalent to connecting the input to the high end of the syndrome register. The decoder now takes the form shown in Figure 12. The decoding procedure for this single error trapping decoder can be described in the following steps:

1. Gate 1 is turned on, and Gate 2 is turned off. The syndrome S(X) is formed by shifting all n bits of the received signal r(X) into the syndrome register. At the same time the n bits of the word are stored into the buffer register.
2. Gate 1 is turned off, and Gate 2 is turned on. With the input cut off, the word read in during Step 1 is now processed. The n bits stored in the buffer register are shifted while searching for a correctable error pattern. As soon as the highest order bit of the syndrome register is one and all the rest of the bits are zero, the single error is trapped in the highest order syndrome bit. The correction is then made since the output of the AND is high, by adding a one modulo 2

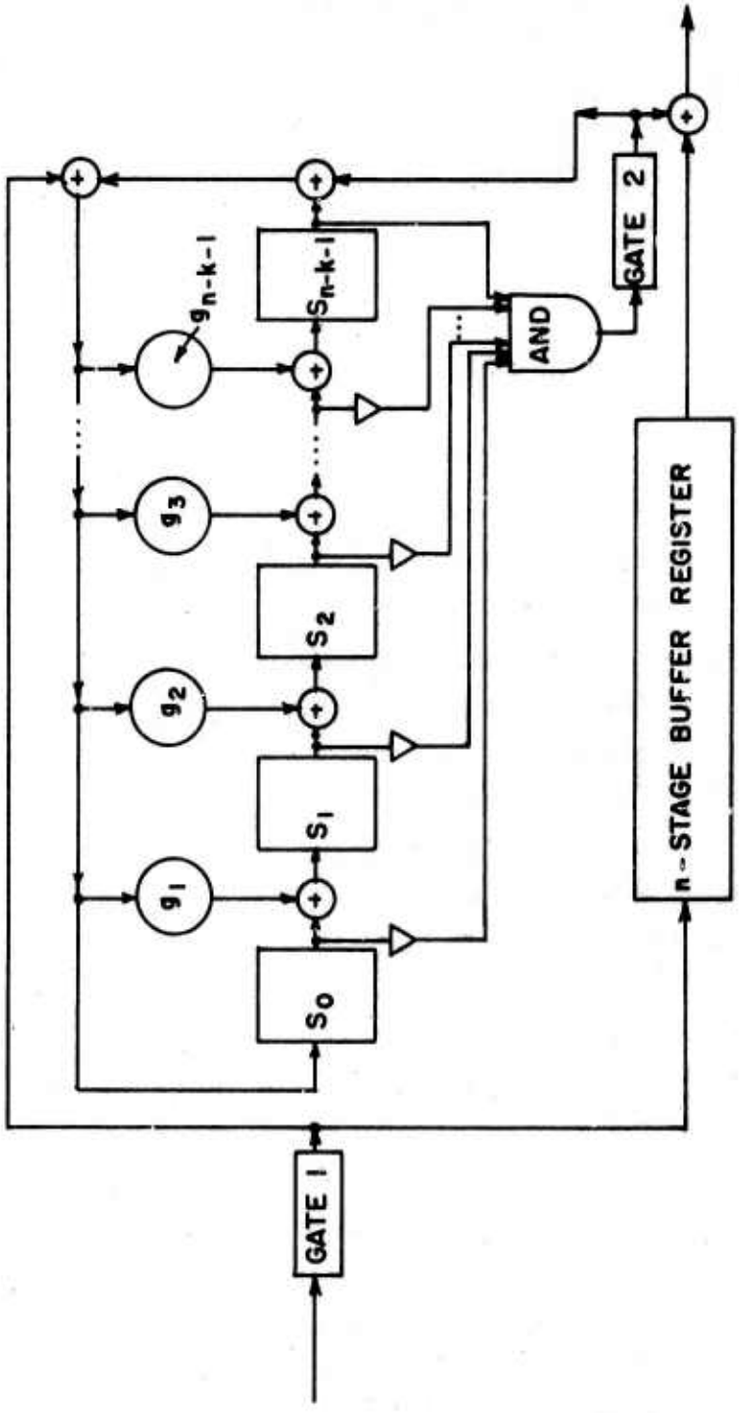


Figure 12. Modified Single Error-Trapping Decoder

to the present output of the buffer register. When the error is corrected, the feedback from the high end of the syndrome register should be eliminated as the syndrome register should contain all zeros. This is accomplished by adding the output of the AND modulo 2 to the output of the highest order syndrome bit.

3. When Step 2 is completed, the decoder is ready for the next word and Step 1 is repeated.

When a shortened cyclic rather than a cyclic code is used, the decoder of Figure 12 and the decoding algorithm for this decoder are still applicable with only the input connection changed for the appropriate shortening. The encoder of Figure 9 is unchanged for a shortened cyclic code. For a cyclic code the input to the syndrome register is X^{n-k} times the input or $X^{n-k}r(X)$ which makes $r(X)$ fed into the X^{n-k} position of the syndrome register. For a shortened cyclic code that is shortened by η bits, $(n-\eta, k-\eta)$ code, from a (n,k) cyclic code, the input to the syndrome register is multiplied by X^{η} (to shift the η deleted bits) or the input is $X^{n-k-\eta}r(X)$. Thus, $r(X)$ if fed into the positions of the syndrome register determined by $C(X)$, the remainder from dividing $X^{n-k+\eta}$ by $g(X)$, i.e.,

$$X^{n-k+\eta} = g(X)q(X) + C(X) \tag{34}$$

For shortened cyclic codes the decoder takes the form shown in Figure 13.

As an example, consider the (12,6) code shortened from the (15,9) single error-correcting code (η=3) of Table IV with generator polynomial

$$g(X) = 1 + X^2 + X^3 + X^4 + X^6 \tag{35}$$

The connection polynomial is determined as

$$C(X) = 1 + X + X^2 + X^3 \tag{36}$$

The decoder for the (12,6) shortened cyclic code is shown in Figure 14. Error-trapping decoding for multiple errors is now considered.

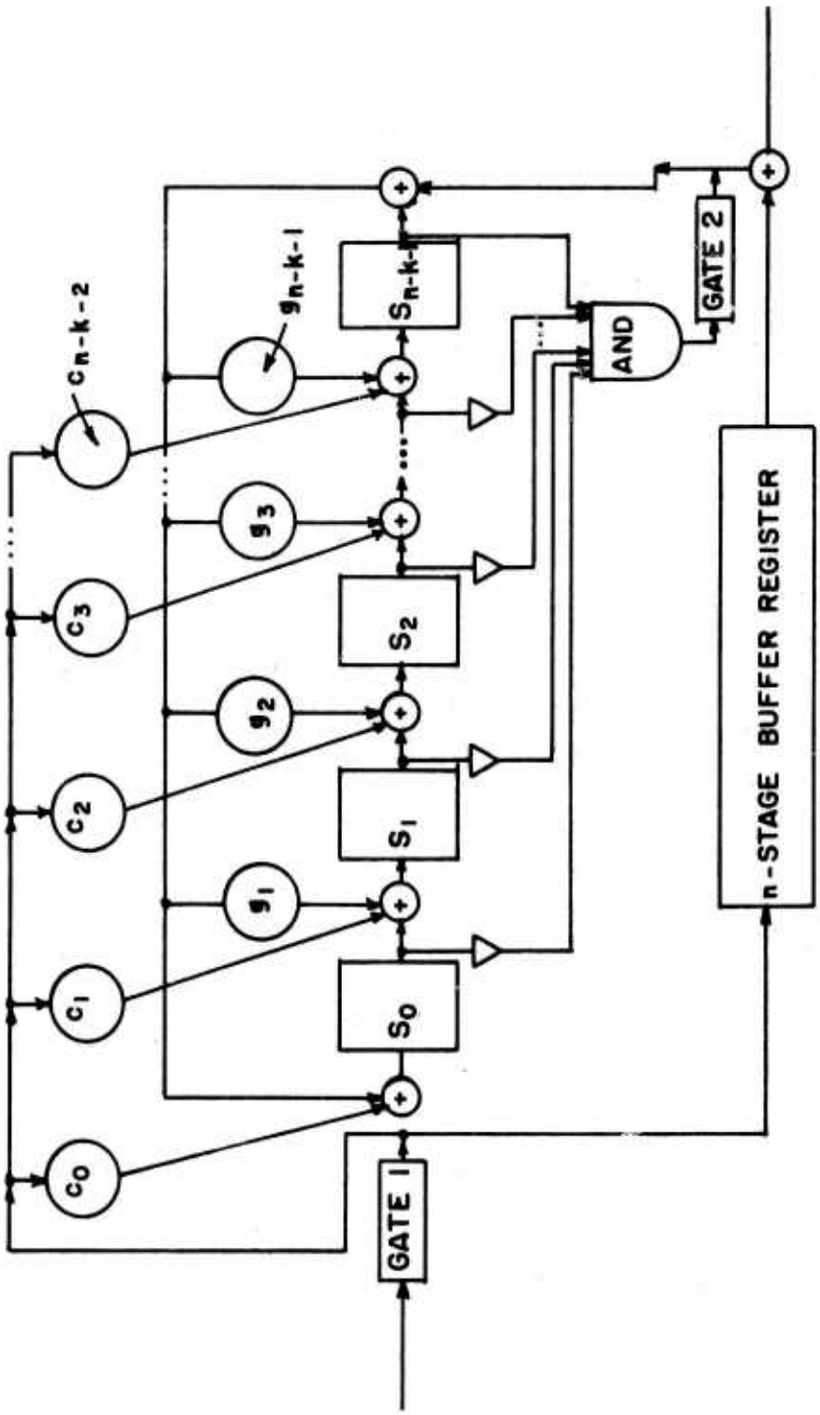


Figure 13. Single Error-Trapping Decoder for Shortened Cyclic Code

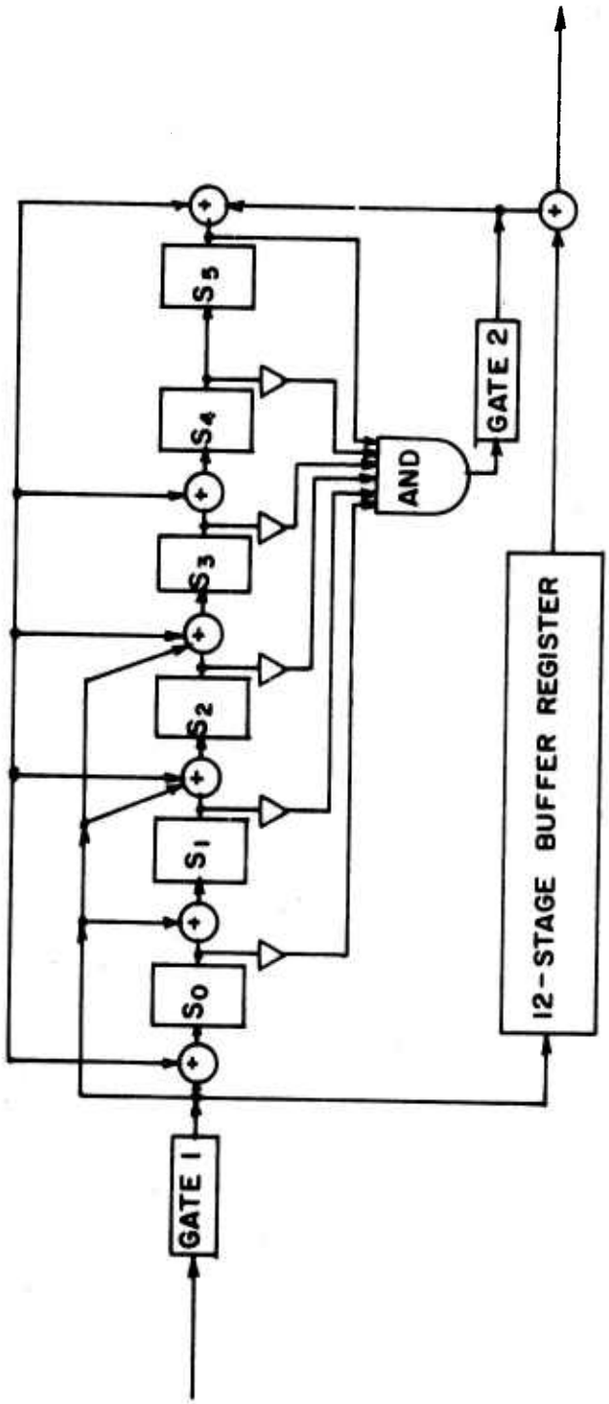


Figure 14. Decoder for (12,6) Shortened Cyclic Code

Error-Trapping Decoding for Multiple Errors

Error-trapping consists of trapping the error patterns in the parity bit positions (in the syndrome bits). For single error-correction this is easily accomplished since a single error can always be trapped in the n-k parity bit positions. With double error-correcting codes there is essentially no further complexity in designing error-trapping decoders as long as all of the possible error can be trapped in the n-k parity positions. This is always possible if

$$n-k \leq k+1 \text{ or } k \leq \frac{n-1}{2} \tag{37}$$

which can be observed by taking the two errors to be in position x^i and X^{i+j}. If j is less than or equal to k, the errors are less than or equal to k digits apart and can be put in less than or equal to k+1, which is less than or equal to n-k positions or trapped in the n-k parity positions. Also for

$$k+1 \leq j \leq n-k \tag{38}$$

the errors are less than or equal to n-k-1 (modulo n) digits apart and can be put in less than or equal to n-k positions or likewise be trapped in the n-k parity positions. Since the distance that these errors are apart are modulo n, these distances can be observed on a circle of n bit positions. For example, consider the (15,7) cyclic code with

$$g(X) = (X^4 + X^3 + X^2 + X + 1)(X^4 + X + 1) = X^8 + X^7 + X^6 + X^4 + 1 \tag{39}$$

This encoder is given in the standard manner as illustrated in Figure 15.

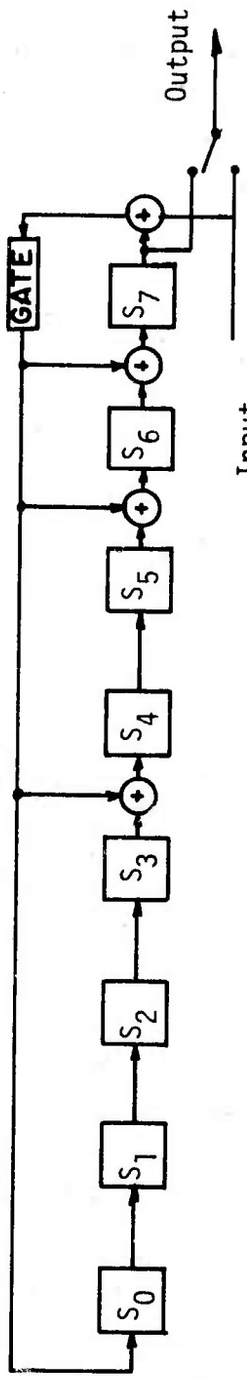


Figure 15. Encoder for (15,7) Cyclic Code

Now for the design of the decoder. Observing our circle representation (Figure 16) of the 15-bit positions where the 0's are the parity positions

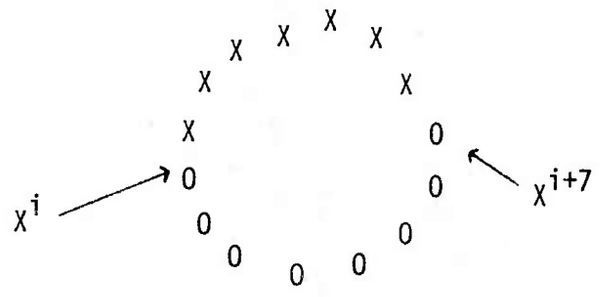


Figure 16. Circle Representation of Bits for (15,7) Cyclic Code

and the X's are the information positions, it can be observed that all double error patterns can be shifted into the 0's (parity positions). Thus, a test for $\sum_{i=0}^{7} S_i$ less than or equal to 2 is needed out of our syndrome register in the decoder. When this test passes, the double error pattern, will be corrected and the decoder takes the form in Figure 17. The decoding procedure for this error-trapping decoder for multiple errors is described in the following steps:

1. G1 (Gate 1) is turned on, G2 is turned on, G3 is turned off, G4 is turned off, and G5 is turned off. The syndrome S(X) is formed by shifting all n bits of the received signal r(X) into the syndrome register. At the same time the n bits of the word are stored into the buffer register.
2. G1 is turned off, G2 remains on, G3 is turned on, G4 remains off, and G5 remains off. With the input cut off the word read in during Step 1 is now processed. For each bit shifted, the syndrome register contents are checked for two or fewer ones. Simultaneously, the bit shifted out of the buffer register is fed back to the low end of the buffer register. When two or fewer ones are found in the syndrome register ($T_0 = 1$), the errors are trapped in the syndrome register and Step 3, for the correction process, is performed.

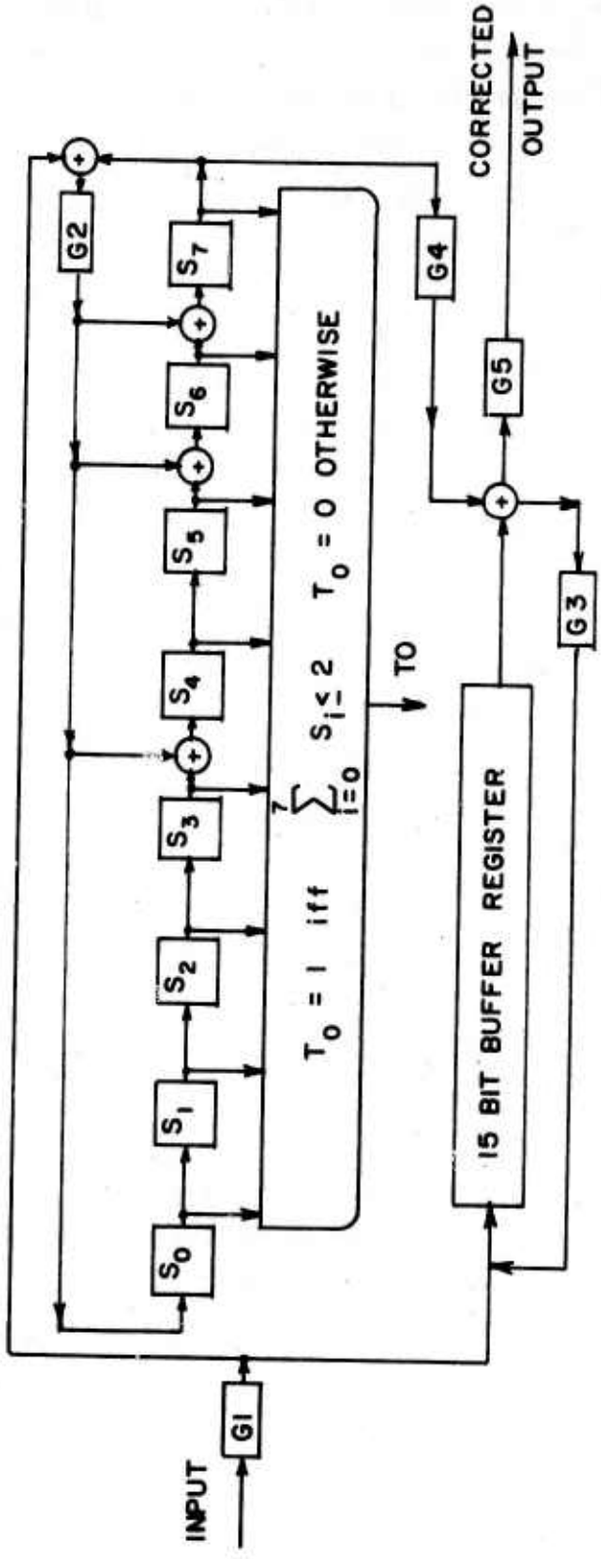


Figure 17. Decoder for (15,7) Double Error-Correcting Code

3. G1 remains off, G2 is turned off, G3 remains on, G4 is turned on, and G5 remains off. The correction is made by feeding the contents of the syndrome register out and adding modulo 2 to the output of the buffer register. The feedback from the high end of the syndrome register is eliminated, as it should be once the errors are located, since G2 is turned off. After there are a total of n shifts in both Steps 2 and 3, the buffer is ready to output the highest order bit and Step 4 is performed.
4. G1 remains off, G2 remains off, G3 is turned off, G4 remains on, and G5 is turned on. The bits are now shifted from the buffer register to the output. It is still possible for some of these bits to be corrected as they are shifted out of the buffer register. If all the errors have been corrected, the output of G4 is zero, no correction made, since the syndrome register will contain all zeros. This step requires n shifts.
5. When Step 4 is completed, the decoder is ready for the next word and Step 1 is repeated.

Table V illustrates how the decoder of Figure 17 processes a given r(X),

$$r(X) = X^4 + X^6 + X^7 \tag{40}$$

For double error-correcting codes for which n-k is less than or equal to k, it is not possible to fit all error patterns in the n-k parity positions. This is also true for higher error-correcting codes. The ideas of error-trapping can still be used by trapping most of the errors in the n-k parity positions and using another test to determine the remaining errors that cannot be trapped in the parity positions. The position outside of the parity positions that will determine the presence of an error is indicated by what is called a covering polynomial $\phi(X)$

$$\phi(X) = X^j \tag{41}$$

TABLE V. PROCESSING OF $r(X)=X^4+X^6+X^7$ WITH (15,7) DECODER

r(X)		s(X)	T_0	Σ_0	Buffer	V*(X)	
X^{14}	0	00000000			$\overline{0}$		
	0	00000000			$0\overline{0}$		
	0	00000000			$00\overline{0}$		
	0	00000000			$000\overline{0}$		
X^{10}	0	00000000			$0000\overline{0}$		
	0	00000000			$00000\overline{0}$		
	0	00000000			$000000\overline{0}$		
	0	00000000			$0000000\overline{0}$		
	1	10001011			$1000000\overline{0}$		
X^5	1	01000101			$1100000\overline{0}$		
	0	10101001			$0110000\overline{0}$		
	1	01010100			$1011000\overline{0}$		
	0	00101010			$0101100\overline{0}$		
	0	00010101			$0010110\overline{0}$		
	0	10000001			$0001011\overline{0}$		
X^0	0	11001011	0	5	$0000101\overline{1}0$		
		11101110	0	6	$\overline{0}000010110000000$		
		01110111	0	6	$0\overline{0}000010110000000$		
		10110000	0	3	$00\overline{0}00001011000000$		
		01011000	0	3	$000\overline{0}0000101100000$		
		00101100	0	3	$0000\overline{0}000010110000$		
		00101100	0	3	$00000\overline{0}00001011000$		
		00001011	0	3	$000000\overline{0}0000101100$		
		10001110	0	4	$0000000\overline{0}000010110$		
		01000111	0	4	$1000000\overline{0}0000101$		
		10101000	0	3	$1100000\overline{0}000010$		
		01010100	0	3	$0110000\overline{0}000001$		
		00101010	0	3	$1011000\overline{0}000000$		
		00010101	0	3	$0101100\overline{0}000000$		
		10000001	1	2	$0010110\overline{0}000000$		
		01000000	1	1	$0001011\overline{0}000000$		
		00100000	1	1	1000101100000	$0\ X^{14}$	Bit 0 Corrected
		00010000	1	1	100010110000	0	
		00001000	1	1	10001011000	0	
		00000100	1	1	1000101100	$0\ X^{10}$	
		00000010	1	1	100010110	0	
		00000001	1	1	10001011	1	Bit 8 Corrected
		00000000	1	0	1000101	1	
		00000000	1	0	100010	1	
		00000000	1	0	10001	$0\ X^5$	
		00000000	1	0	1000	1	
		00000000	1	0	100	0	
		00000000	1	0	10	0	
		00000000	1	0	1	0	
		00000000	1	0		$1\ X^0$	

where j + 1 indicates the number of positions from the highest order parity bit. Letting $e_p(X)$ and $e_I(X)$ be the error patterns in the parity and information bits, respectively, the syndrome for each covering polynomial can be written as

$$S(X) = \rho_i(X) + e_p(X) \tag{42}$$

where $\rho_i(X)$ is the remainder from dividing the $i^{\underline{th}}$ error position outside the parity positions, $X^{n-k}\phi_i(X)$, by g(X). Once this syndrome has been detected, the error pattern in the error bits, $e_p(X)$, is obtained as the modulo 2 sum of S(X) and $\rho_i(X)$, and non-parity errors are fed into the low end of the syndrome register after passing through a (k-j-1) bit delay.

Consider the (21, 12) double error-correcting cyclic code with

$$g(X) = (X^6 + X^4 + X^2 + X + 1)\,(X^3 + X^2 + 1) = X^9 + X^8 + X^7 + X^5 + X^4 + X + 1 \tag{43}$$

Letting d_0 be the separation of the two errors, for d_0 less than or equal to 8, these errors can be trapped in the parity positions, and $\phi_0(X)$ and $\rho_0(X)$ equals 0. For d_0 equals to 9, 10, as shown in Figure 18, it is not possible to trap the errors in the parity positions, and thus a $\phi_1(X)$ not equal to 0 is needed. $\phi_1(X)$ equal to 1 will cover d_0 equal to 9 but not d_0 equal to 10. Now $\phi_1(X)$ equal to X will cover d_0 equal to 9, 10 (in fact, $\phi_1(X)$ equal to $X^1, X^2, \ldots\ldots, X^{10}$ will cover d_0 equal to 9, 10). For $\phi_1(X)$ equal to X, $\rho_1(X)$ equals $1 + X^2 + X^4 + X^6 + X^7$ and the test for an error covered by $\phi_1(X)$ and the other error in the parity bits is to test $S(X) + \rho_1(X)$ for 0 or 1 nonzero terms, i.e., the error pattern is covered by $\phi_1(X)$ and trapped in the parity bits if the sum $\overline{S}_0 + S_1 + \overline{S}_2 + S_3 + \overline{S}_4 + S_5 + \overline{S}_6 + \overline{S}_7 + S_8$ is less than or equal to 1. Upon detecting this error pattern, $\phi_1(X)$ is added to S(X) to obtain the error in the parity positions and a 1 is introduced through a 10-bit delay to S_0 to correct the error covered by $\phi_1(X)$.

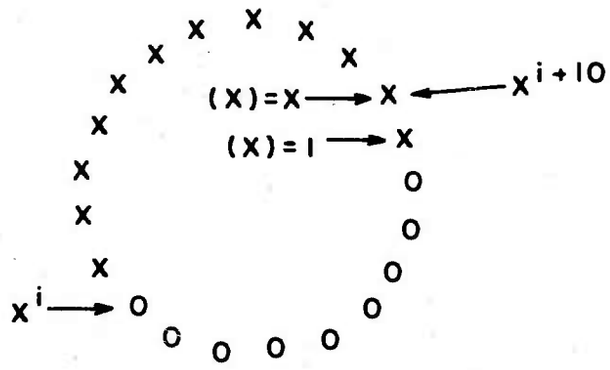


Figure 18. Circle Representation of Bits for (21,12) Cyclic Code

For the code at hand $\phi_1(X)$ can be picked to give the minimum connections for $\rho_1(X)$ or $\phi_1(X)$ equal to X^{10} can be picked for a minimum number of delay bits for correcting the error covered by $\phi_1(X)$. Picking for illustration,

$$\phi_1(X) = X^{10} \tag{44}$$

then

$$\rho_1(X) = 1 + X^2 + X^4 + X^5 + X^8 \tag{45}$$

and the error trapping decoder is obtained as shown in Figure 19. The decoding procedure for this error-trapping decoder for multiple errors is described in the following steps:

1. G1 and G2 are turned on and G3, G4 and G5 are turned off. The syndrome S(X) is formed by shifting all n bits of the received signal r(X) into the syndrome register. At the same time the n bits of the word are stored into the buffer register.
2. G2 and G3 are turned on, and G1, G4 and G5 are turned off. With the input cut off, the word of n bits read in during Step 1 is now processed. For each bit shifted, the bit coming out of the buffer register is fed back to the low end of the buffer register. The syndrome is tested for correctable error patterns as follows:
   (a) If $\sum_{i=0}^{8} S_i \leq 2\ T_0 = 1$ and the errors are trapped in the 9-bit syndrome register. Step 3, for the correction process, is then performed
   (b) If $\sum_{i=0}^{8} (S_i + \rho_i) \leq T_1 = 1$ and one error is trapped in the syndrome register and the other error occurs at two bits before the lowest order syndrome bit. To obtain the error pattern $\rho(X)$ is added to S(X) (T_1 is added to the appropriate registers) and Step 3, for the correction process, is performed.

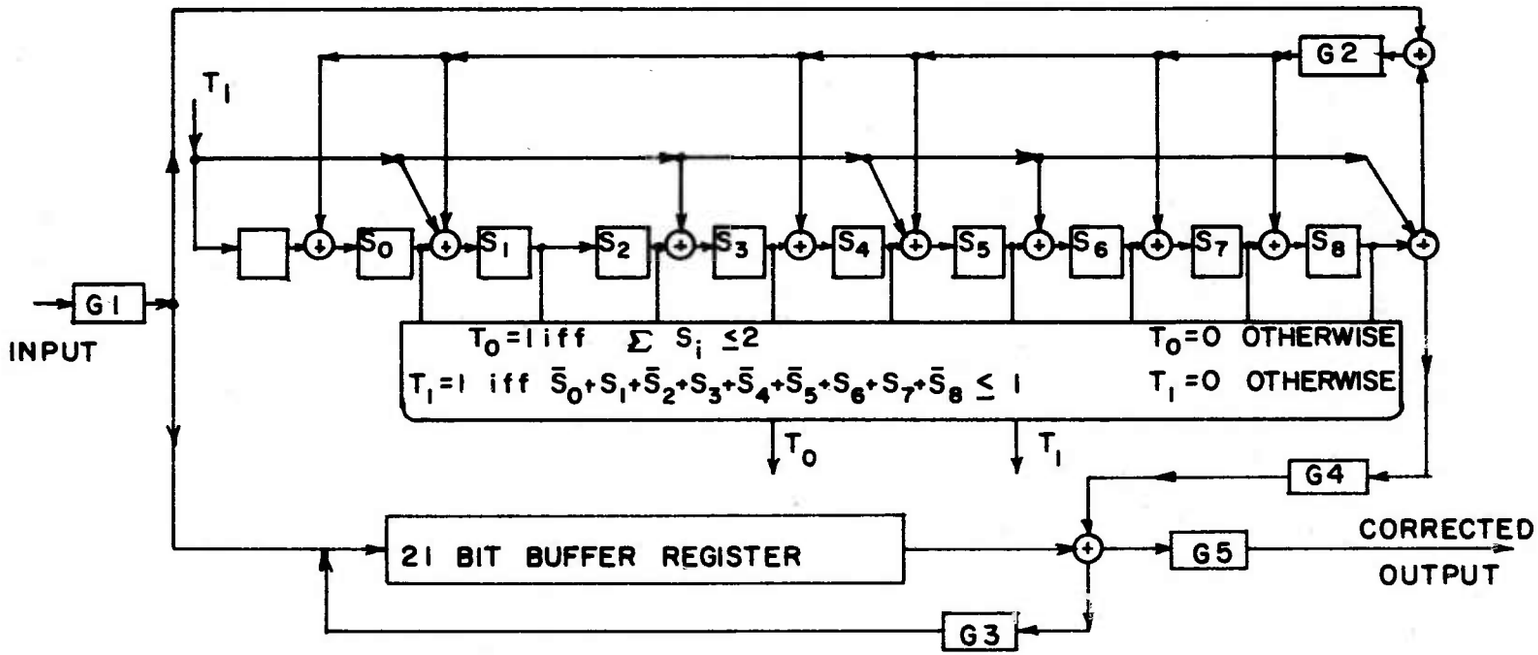


Figure 19. Decoder for (21,12) Double Error-Correcting Code

Step 2 is continued until either a or b is satisfied.

3. G3 and G4 are turned on, and G1, G2 and G5 are turned off. The correction is made by feeding the contents of the syndrome register out and adding modulo 2 to the output of the buffer register. The feedback from the high end of the syndrome register is eliminated since G2 is turned off. After there are a total of n shifts in both Steps 2 and 3, the buffer is ready to output the highest order bit and Step 4 is performed.
4. G4 and G5 are turned on, and G1, G2 and G3 are turned off. The corrected bits are now shifted from the buffer register to the output. It is still possible for some bits to be corrected, but if all the errors have been corrected, the output of G4 is zero (syndrome register contains all zeros) and no correction is made. This step requires n shifts.
5. When Step 4 is completed, the decoder is ready for the next word and Step 1 is repeated.

Table VI illustrates how the decoder of Figure 19 processes a given r(X).

$$r(X) = 1 + X^4 + X^5 + X^7 + X^8 + X^9 + X^{14} \tag{46}$$

Specific random error-correcting codes for the word formats of interest in this report are now considered. The first format of interest has a 48-bit word with 8 parity bits or a (48,40) code is desired. Searching Table IV with n-k equal to 8 and a length greater than or equal to 48, for the largest t_c, it is seen that either the (51,43) or the (63,55) code fits the specification. Choosing the (63,55) code the generator polynomial is given as

$$g(X) = 1 + X^3 + X^6 + X^7 + X^8 \tag{47}$$

and the encoder for the shortened (48,40) code, which is the same as the (63,55) code is shown in Figure 20. This is a single error-correcting code.

TABLE VI. PROCESSING OF $r(X) = 1 + X^4 + X^5 + X^7 + X^8 + X^9 + X^{14}$ WITH (21,12) DECODER

r(X)		s(X)	T_0	Σ_0	T_1	Σ_1	Buffer	V*(X)
X^{20}	0	000000000						$\overline{0}$
	0	000000000						$0\overline{0}$
	0	000000000						$00\overline{0}$
	0	000000000						$000\overline{0}$
	0	000000000						$0000\overline{0}$
X^{15}	0	000000000						$00000\overline{0}$
	1	110011011						$000000\overline{0}$
	0	101010110						$1000000\overline{0}$
	0	010101011						$01000000\overline{0}$
	0	111001110						$001000000\overline{0}$
X^{10}	0	011100111						$0001000000\overline{0}$
	1	001110011						$00001000000\overline{0}$
	1	000111001						$100001000000\overline{0}$
	1	000011100						$1100001000000\overline{0}$
	0	000001110						$11100001000000\overline{0}$
X^5	1	110011100						$01110000100000\overline{0}$
	1	101010101						$10111000010000\overline{0}$
	0	100110001						$11011100001000\overline{0}$
	0	100000011						$01101110000100\overline{0}$
	0	100011010						$00110111000010\overline{0}$
X^0	1	100010110	0	4	0	5		$00011011100001\overline{0}$
		010001011	0	4	0	5		$10001101110000\overline{0}$
		111011110	0	7	0	4		$\overline{0}1000110111000010000$
		011101111	0	7	0	6		$0\overline{0}100011011100001000$
		111101100	0	6	0	5		$00\overline{0}10001101110000100$
		011110110	0	6	0	7		$000\overline{0}1000110111000010$
		001111011	0	6	0	3		$0000\overline{0}100011011100001$
		110100110	0	5	0	8		$10000\overline{0}10001101110000$
		011010011	0	5	0	4		$01000000\overline{0}1000110111000$
		111110010	0	6	0	5		$001000000\overline{0}100011011100$
		011111001	0	6	0	3		$0001000000\overline{0}10001101110$
		111100111	0	7	0	6		$00001000000\overline{0}1000110111$

TABLE VI. (continued)

r(X)	s(X)	T_0	Σ_0	T_1	Σ_1	Buffer	V*(X)	
	101101000	0	4	0	3	10000100000$\overline{0}$100011011		
	010110100	0	4	0	7	11000010000$\overline{0}$10001101		
	001011010	0	4	0	3	11100001000000$\overline{1}$000110		
	000101101	C	4	0	5	01110000100000$\overline{0}$100011		
	110001101	0	5	0	4	10111000010000$\overline{0}$10001		
	101011101	0	6	1	1	11011100001000000$\overline{1}$000		
	1000000010	1	1	0	6	01101110000100000$\overline{0}$100		
	100000001	1	2	0	3	00110111000010000000$\overline{1}$0		
	010000000	1	1	0	6	10011011100001000000$\overline{1}$		Bit 1 Corrected
	001000000	1	1	0	4	11001101110000100000$\overline{0}$		
	000100000	1	1	0	6	1100110111000010000	0 X^{20}	
	000010000	1	1	0	4	110011011100001000	0	
	000001000	1	1	0	4	11001101110000100	0	
	000000100	1	1	0	6	1100110111000010	0	
	000000010	1	1	0	6	110011011100001	0 X^{15}	
	000000001	1	1	0	4	11001101110000	0	
	000000000	1	0	0	5	1100110111000	0	Bit 14 Corrected
	000000000	1	0	0	5	110011011100	0	
	000000000	1	0	0	5	11001101110	0	
	000000000	1	0	0	5	1100110111	0 X^{10}	
	000000000	1	0	0	5	110011011	1	
	000000000	1	0	0	5	11001101	1	
	000000000	1	0	0	5	1100110	1	
	000000000	1	0	0	5	110011	0	
	000000000	1	0	0	5	11001	1 X^{5}	
	000000000	1	0	0	5	1100	1	
	000000000	1	0	0	5	110	0	
	000000000	1	0	0	5	11	0	
	000000000	1	0	0	5	1	1	
	000000000	1	0	0	5		1 X^{0}	

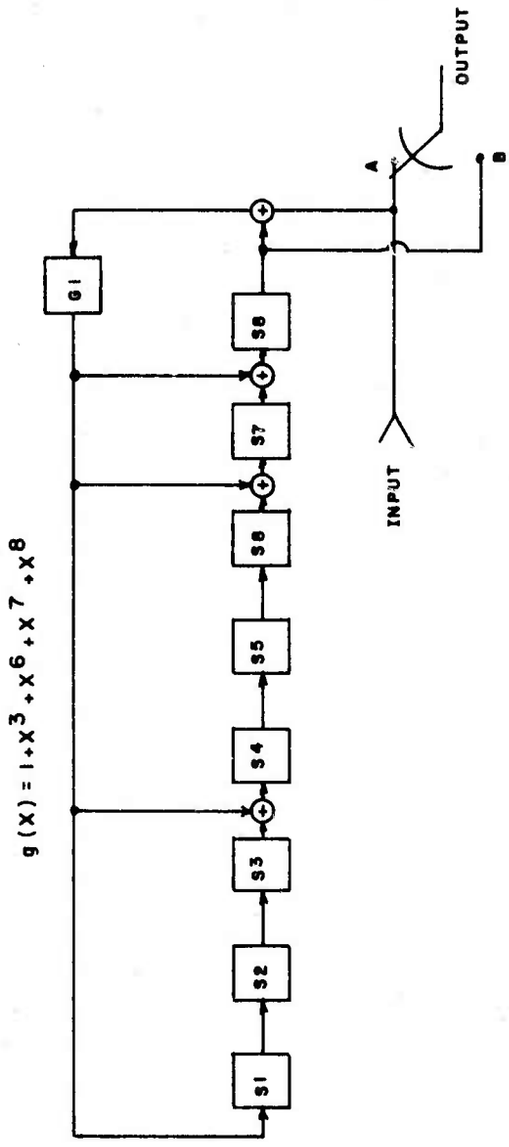


Figure 20. Encoder for (48,40) Shortened Cyclic Single Error-Correcting Code

The (48,40) code has to be shortened 15 bits (η=15) from the (63,55) code and thus the connection polynomial is obtained from

$$X^{n-k+\eta} = X^{8+15} = g(X)\, q(X) + c(X) \tag{48}$$

as

$$c(X) = 1 + X^3 + X^5 \tag{49}$$

The decoder is shown in Figure 21. For the (24,16) code format no additional error-correcting capability can be achieved. This is also true for the (43,38) code format. But for the (24,14) code format the (31,21) double error-correcting code listed in Table IV under 10 parity bits can be shortened. The generator polynomial is given as

$$g(X) = 1 + X^3 + X^5 + X^6 + X^8 + X^9 + X^{10} \tag{50}$$

and the encoder for this shortened (24,14) code is given in Figure 22.

With η equal to 7 here, the connection polynomial is obtained as

$$c(X) = 1 + X^2 + X^3 + X^4 + X^7 + X^9 \tag{51}$$

In addition to the errors trapped in the parity positions, which are covered by $\phi_0(X)$ equal to 0, choosing

$$\phi_1(X) = X^{6+\eta} = X^{13} \tag{52}$$

will cover all the errors that cannot be trapped in the parity positions; $\rho_1(X)$ is then obtained from

$$X^{n-k}\phi_1(X) = X^{23} = g(X)\, q(X) + \rho_1(X) \tag{53}$$

as

$$\rho_1(X) = 1 + X^4 + X^6 + X^7 \tag{54}$$

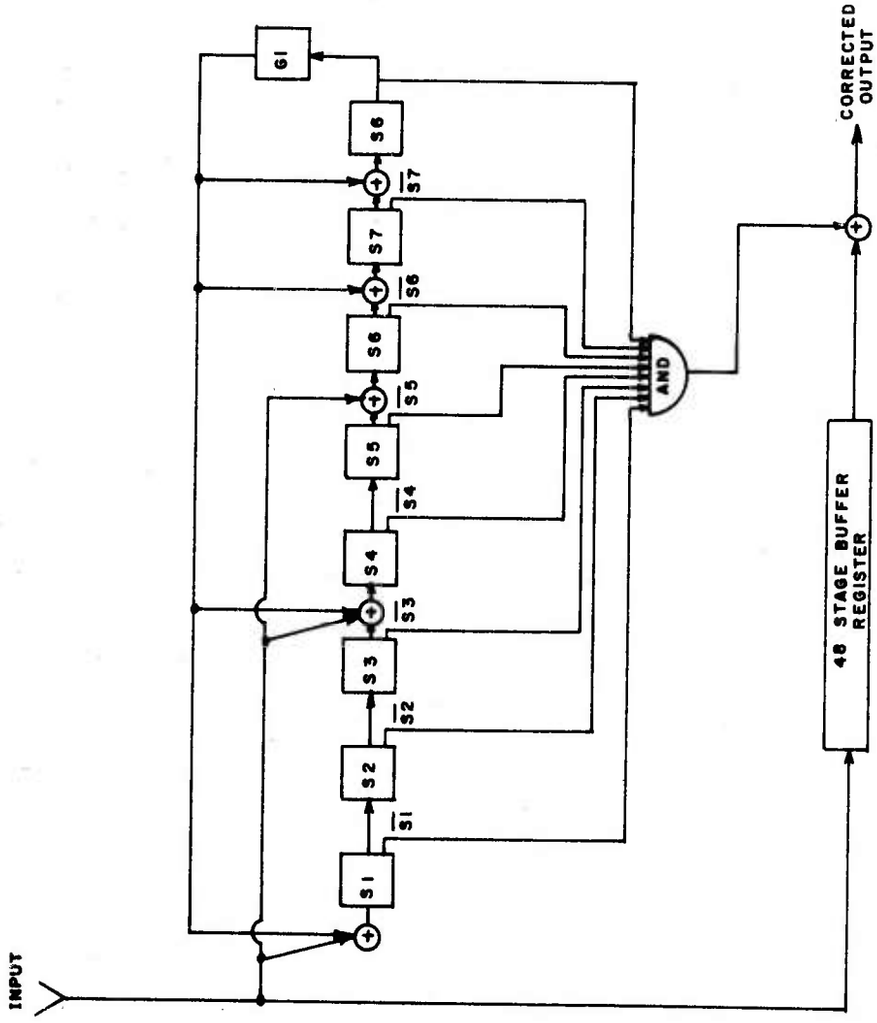


Figure 21. Decoder for (48,40) Shortened Cyclic Single Error-Correcting Code

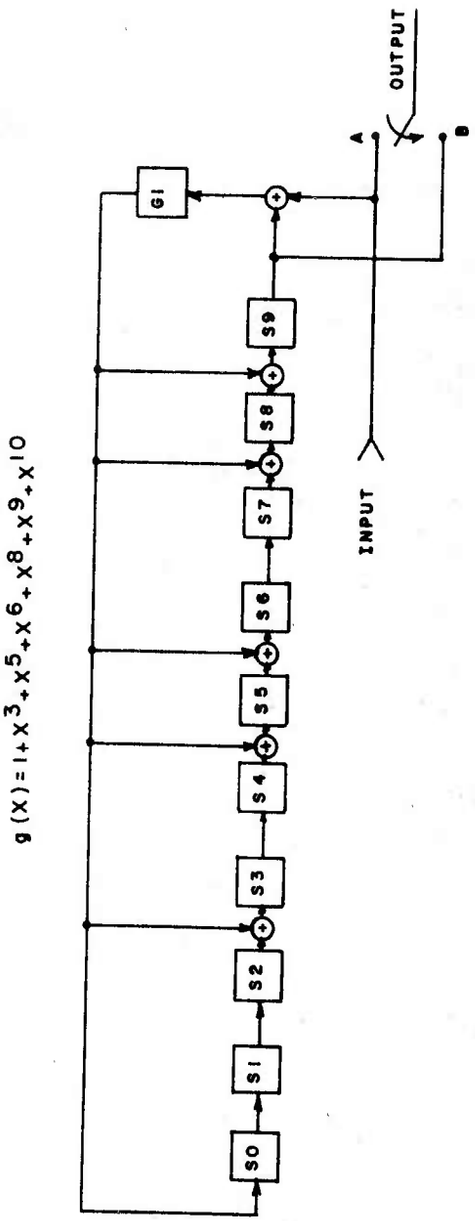


Figure 22. Encoder for (24,14) Double Error-Correcting Code

The decoder is shown in Figure 23 where

$$\Sigma_0 = \sum_{i=0}^{9} S_i \tag{55}$$

and

$$\Sigma_1 = \sum_{i=0}^{9} (S_i + \rho_i) = \tag{56}$$

$$\overline{S}_0 + S_1 + S_2 + S_3 + \overline{S}_4 + S_5 + \overline{S}_6 + \overline{S}_7 + S_8 + S_9$$

Tables VII, VIII, and IX illustrate how this (24,14) decoder processes three different received signals. Table VII illustrates the processing of r(X), where

$$r(X) = X^3 + X^5 + X^8 + X^9 + X^{10} \tag{57}$$

and the test Σ_0 is satisfied. Table VIII illustrates the processing of

$$r(X) = X^3 + X^5 + X^6 + X^8 + X^{10} \tag{58}$$

and the test Σ_1 is satisifed. Table IX illustrates the processing of

$$r(X) = X^3 + X^5 + X^6 + X^8 + X^9 + X^{10} + X^{18} \tag{59}$$

and the test Σ_1 is satisfied. Table IX also illustrates the fact that a 7-bit advance has to be inserted as the step from X^0 to X^{23} is performed, because this code has been shortened by 7 bits.

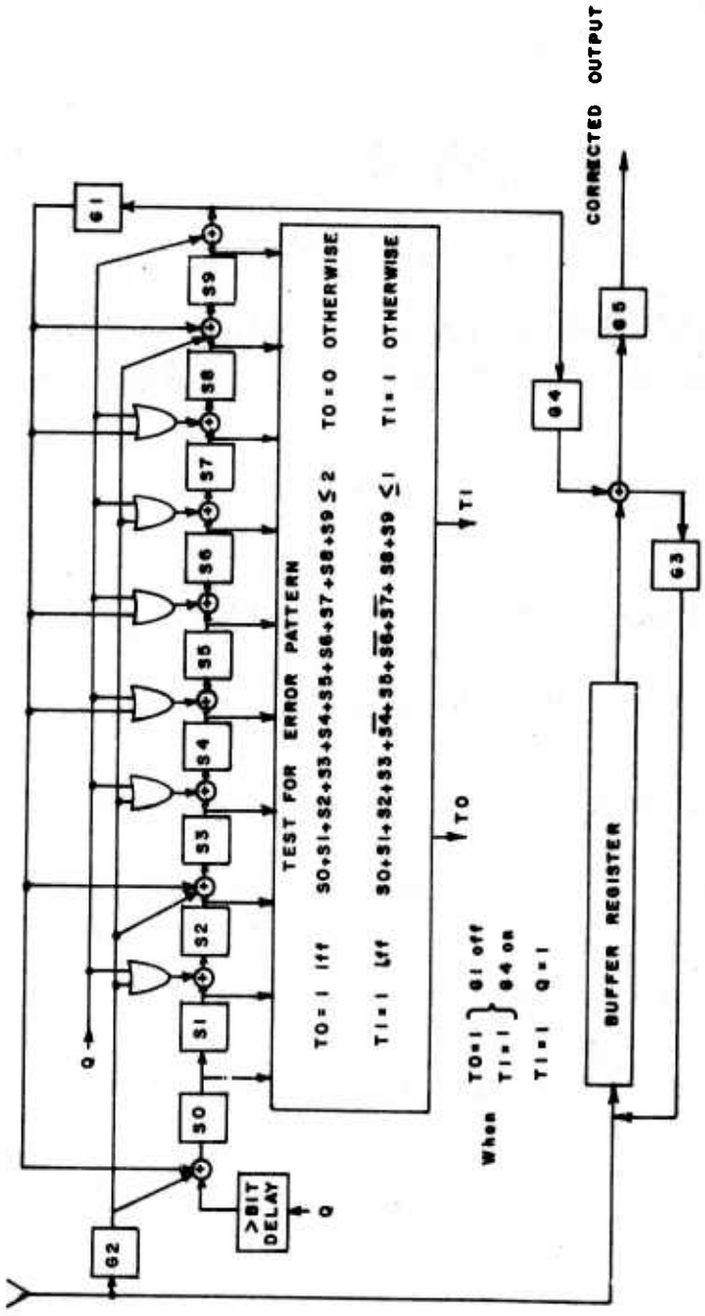


Figure 23. Decoder for (24,14) Double Error-Correcting Code

TABLE VII. PROCESSING OF $r(X) = X^3 + X^5 + X^8 + X^9 + X^{10}$ WITH (24,14) DECODER

r(X)		s(X)	T_0	Σ_0	T_1	Σ_1	Buffer	$V^*(X)$
X^{23}	0	0000000000					$\overline{0}$	
	0	0000000000					$0\overline{0}$	
	0	0000000000					$00\overline{0}$	
X^{20}	0	0000000000					$000\overline{0}$	
	0	0000000000					$0000\overline{0}$	
	0	0000000000					$00000\overline{0}$	
	0	0000000000					$000000\overline{0}$	
	0	0000000000					$0000000\overline{0}$	
X^{15}	0	0000000000					$00000000\overline{0}$	
	0	0000000000					$000000000\overline{0}$	
	0	0000000000					$0000000000\overline{0}$	
	0	0000000000					$00000000000\overline{0}$	
	0	0000000000					$000000000000\overline{0}$	
X^{10}	1	1011100101					$100000000000000\overline{0}$	
	1	0111001100					$1100000000000000\overline{0}$	
	1	1000000011					$1110000000000000\overline{0}$	
	0	1101011010					$0111000000000000\overline{0}$	
	0	0110101101					$0011100000000000\overline{0}$	
X^5	1	0001101000					$1001110000000000\overline{0}$	
	0	0000110100					$0100111000000000\overline{0}$	
	1	1011111111					$1010011100000000\overline{0}$	
	0	1100100100					$0101001110000000\overline{0}$	
	0	0110010010					$0010100111000000\overline{0}$	
X^0	0	0011001001	0	4	0	4	$0001010011100000\overline{0}$	
		1000111111	0	7	0	3	$\overline{0}000101001110000$	
		1101000100	0	3	0	4	$0\overline{0}00101001110000$	
		0110100010	0	3	0	6	$00\overline{0}0010100111000$	
		0011010001	0	3	0	8	$000\overline{0}001010011100$	
		1000110011	0	5	0	5	$0000\overline{0}00101001110$	
		1101000010	0	4	0	6	$00000\overline{0}0010100111$	
		0110100001	0	4	0	6	$000000\overline{0}00101001$	
		1010001011	0	5	0	5	$0000000\overline{0}0010100$	

TABLE VII. (continued)

r(X)	s(X)	T_0	Σ_0	T_1	Σ_1	Buffer	$V^*(X)$	
	1100011110	0	6	0	4	000000000$\overline{0}$0001010011110000		
	0110001111	0	6	0	6	0000000000$\overline{0}$000101001111000		
	1010011100	0	5	0	3	00000000000$\overline{0}$00010100111100		
	0101001110	0	5	0	5	000000000000$\overline{0}$0001010011110		
	0010100111	0	5	0	5	0000000000000$\overline{0}$000101001111		
	1000001000	1	2	0	2	100000000000000$\overline{0}$001010011		Bit 6 Corrected
	0100000100	1	2	0	4	110000000000000$\overline{0}$000101001		
	0010000010	1	2	0	6	111000000000000$\overline{0}$$\overline{0}$00010100		
	0001000001	1	2	0	6	011100000000000$\overline{0}$$\overline{0}$$\overline{0}$001010		
	0000100000	1	1	0	3	101110000000000$\overline{0}$$\overline{0}$$\overline{0}$00101		
	0000010000	1	1	0	5	110111000000000$\overline{0}$$\overline{0}$$\overline{0}$$\overline{0}$0010		
	0000001000	1	1	0	3	011011100000000$\overline{0}$$\overline{0}$$\overline{0}$$\overline{0}$$\overline{0}$001		
	0000000100	1	1	0	3	101101110000000$\overline{0}$$\overline{0}$$\overline{0}$$\overline{0}$$\overline{0}$$\overline{0}$000		
	0000000010	1	1	0	5	010110111000000$\overline{0}$$\overline{0}$$\overline{0}$$\overline{0}$$\overline{0}$$\overline{0}$$\overline{0}$000		
	0000000001	1	1	0	5	001011011100000$\overline{0}$$\overline{0}$$\overline{0}$$\overline{0}$$\overline{0}$$\overline{0}$$\overline{0}$$\overline{0}$00		
	0000000000	1	0	0	4	100101101110000$\overline{0}$$\overline{0}$$\overline{0}$$\overline{0}$$\overline{0}$$\overline{0}$$\overline{0}$$\overline{0}$$\overline{0}$0		Bit 0 Corrected
	0000000000	1	0	0	4	10010110111000000000000	0 X^{23}	
	0000000000	1	0	0	4	1001011011100000000000	0	
	0000000000	1	0	0	4	100101101110000000000	0	
	0000000000	1	0	0	4	10010110111000000000	0 X^{20}	
	0000000000	1	0	0	4	1001011011100000000	0	
	0000000000	1	0	0	4	100101101110000000	0	
	0000000000	1	0	0	4	10010110111000000	0	
	0000000000	1	0	0	4	1001011011100000	0	
	0000000000	1	0	0	4	100101101110000	0 X^{15}	
	0000000000	1	0	0	4	10010110111000	0	
	0000000000	1	0	0	4	1001011011100	0	
	0000000000	1	0	0	4	100101101110	0	
	0000000000	1	0	0	4	10010110111	0	
	0000000000	1	0	0	4	1001011011	1 X^{10}	
	0000000000	1	0	0	4	100101101	1	
	0000000000	1	0	0	4	10010110	1	
	0000000000	1	0	0	4	1001011	0	

TABLE VII. (continued)

r(X)	s(X)	T_0	Σ_0	T_1	Σ_1	Buffer	$V^*(X)$
	0000000000	1	0	0	4	100101	1
	0000000000	1	0	0	4	10010	1 X^5
	0000000000	1	0	0	4	1001	0
	0000000000	1	0	0	4	100	1
	0000000000	1	0	0	4	10	0
	0000000000	1	0	0	4	1	0
	0000000000	1	0	0	4		1 X^0

TABLE VIII. PROCESSING OF $r(X) = X^3 + X^5 + X^6 + X^8 + X^{10}$ WITH (24,14) DECODER

r(X)		s(X)	T_0	Σ_0	T_1	Σ_1	Buffer	$V^*(X)$
X^{23}	0	0000000000					$\overline{0}$	
	0	0000000000					$0\overline{0}$	
	0	0000000000					$00\overline{0}$	
X^{20}	0	0000000000					$000\overline{0}$	
	0	0000000000					$0000\overline{0}$	
	0	0000000000					$00000\overline{0}$	
	0	0000000000					$000000\overline{0}$	
	0	0000000000					$0000000\overline{0}$	
X^{15}	0	0000000000					$00000000\overline{0}$	
	0	0000000000					$000000000\overline{0}$	
	0	0000000000					$0000000000\overline{0}$	
	0	0000000000					$00000000000\overline{0}$	
	0	0000000000					$000000000000\overline{0}$	
X^{10}	1	1011100101					$100000000000000\overline{0}$	
	0	1100101001					$010000000000000\overline{0}$	
	1	0100101010					$101000000000000\overline{0}$	
	0	0010010101					$010100000000000\overline{0}$	
	1	0011110100					$101010000000000\overline{0}$	
X^5	1	1010011111					$110101000000000\overline{0}$	
	0	1100010100					$011010100000000\overline{0}$	
	1	1101101111					$101101010000000\overline{0}$	
	0	1111101100					$010110101000000\overline{0}$	
	0	0111110110					$001011010100000\overline{0}$	
X^0	0	0011111011	0	7	0	7	$000101101010000\overline{0}$	
		1000100110	0	4	0	2	$\overline{0}000101101010000000000$	
		0100010011	0	4	0	8	$0\overline{0}000101101010000000000$	
		1011010010	0	5	0	7	$00\overline{0}000101101010000000000$	
		0101101001	0	5	0	5	$000\overline{0}000101101010000000000$	
		1011101111	0	8	0	4	$0000\overline{0}000101101010000000000$	
		1100101100	0	5	1	1	$00000\overline{0}000101101010000000$	
		7B0010000000	1	1	0	5	$000000\overline{0}000101101010000000$	
		6B0001000000	1	1	0	5	$0000000\overline{0}000101101010000000$	
		5B0000100000	1	1	0	3	$00000000\overline{0}000101101010000$	
		4B0000010000	1	1	0	5	$000000000\overline{0}000101101010000$	

TABLE VIII. (continued)

r(X)	s(X)	T_0	Σ_0	T_1	Σ_1	Buffer	$V^*(X)$	
	3B0000001000	1	1	0	3	000000000000001011010100		
	2B0000000100	1	1	0	3	000000000000000101101010		
	1B0000000010	1	1	0	5	000000000000000010110101		
	1000000001	1	2	0	4	100000000000000001011010		
	0100000000	1	1	0	5	110000000000000000101101		Bit 9 Corrected
	0010000000	1	1	0	5	111000000000000000010110		
	0001000000	1	1	0	5	011100000000000000001011		
	0000100000	1	1	0	3	101110000000000000000101		
	0000010000	1	1	0	5	110111000000000000000010		
	0000001000	1	1	0	3	011011100000000000000001		
	0000000100	1	1	0	3	101101110000000000000000		
	0000000010	1	1	0	5	010110111000000000000000		
	0000000001	1	1	0	5	001011011100000000000000		
	0000000000	1	0	0	4	100101101110000000000000		Bit 0 Corrected
	0000000000	1	0	0	4	10010110111000000000000	0	X^{23}
	0000000000	1	0	0	4	1001011011100000000000	0	
	0000000000	1	0	0	4	100101101110000000000	0	
	0000000000	1	0	0	4	10010110111000000000	0	X^{20}
	0000000000	1	0	0	4	1001011011100000000	0	
	0000000000	1	0	0	4	100101101110000000	0	
	0000000000	1	0	0	4	10010110111000000	0	
	0000000000	1	0	0	4	1001011011100000	0	
	0000000000	1	0	0	4	100101101110000	0	X^{15}
	0000000000	1	0	0	4	10010110111000	0	
	0000000000	1	0	0	4	1001011011100	0	
	0000000000	1	0	0	4	100101101110	0	
	0000000000	1	0	0	4	10010110111	0	
	0000000000	1	0	0	4	1001011011	1	X^{10}
	0000000000	1	0	0	4	100101101	1	
	0000000000	1	0	0	4	10010110	1	
	0000000000	1	0	0	4	1001011	0	
	0000000000	1	0	0	4	100101	1	
	0000000000	1	0	0	4	10010	1	X^5

TABLE VIII. (continued)

r(X)	s(X)	T_0	Σ_0	T_1	Σ_1	Buffer	$V^*(X)$
	0000000000	1	0	0	4	1010	0
	0000000000	1	0	0	4	100	1
	0000000000	1	0	0	4	10	0
	0000000000	1	0	0	4	1	0
	0000000000	1	0	0	4		1 X^0

TABLE IX. PROCESSING OF $r(X) = X^3 + X^5 + X^6 + X^8 + X^9 + X^{10} + X^{18}$ WITH (24,14) DECODER

r(X)		s(X)	T_0	Σ_0	T_1	Σ_1	Buffer	V*(X)
X^{23}	0	0000000000					$\overline{0}$	
	0	0000000000					$0\overline{0}$	
	0	0000000000					$00\overline{0}$	
X^{20}	0	0000000000					$000\overline{0}$	
	0	0000000000					$0000\overline{0}$	
	1	1011100101					$00000\overline{0}$	
	0	1100101001					$10000\overline{0}$	
	0	1111001111					$01000\overline{0}$	
X^{15}	0	1110111100					00100000	
	0	0111011110					000100000	
	0	0011101111					0000100000	
	0	1000101100					00000100000	
	0	0100010110					000000100000	
X^{10}	1	1001101110					0000000100000	
	1	1111010010					10000000100000	
	1	1100001100					11000000100000	
	0	0110000110					11100000100000	
	1	1000100110					01110000100000	
X^5	1	1111110110					10111000100000	
	0	0111111011					11011100100000	
	1	0001000011					01101110100000	
	0	1001111010					10110111100000	
	0	0100111101					01011011110000	
X^0	0	1011000101	0	5	0	5	00101101111000	
		1100111001	0	6	0	4	00010110111100	
		1111000111	0	7	0	7	00001011011110	
		1110111000	0	6	0	4	00000101101111	
		0111011100	0	6	0	6	00000010110111	
		0011101110	0	6	0	4	00000001011011	
		0001110111	0	6	0	6	10000000101101	
		1001100000	0	3	0	3	01000000010110	

TABLE IX. (continued)

r(X)	s(X)	T_0	Σ_0	T_1	Σ_1	Buffer	$V^*(X)$	
	0100110000	0	3	0	5	00100000$\overline{0}$0010110111100000		
	0010011000	0	3	0	5	000100000$\overline{0}$0010110111000		
	0001001100	0	3	0	3	0000100000$\overline{0}$0010110111000		
	0000100110	0	3	0	3	00000100000$\overline{0}$0010110111000		
	0000010011	0	3	0	7	000000100000$\overline{0}$0010110111000		
	1001010010	0	4	0	6	0000000100000$\overline{0}$0010110111		
	0100101001	0	4	0	4	1000000010000$\overline{0}$0001011011		
	1011011111	0	7	0	5	1100000001000$\overline{0}$0000101101		
	1100111100	0	6	0	2	1110000000100000$\overline{0}$0010110		
	0110011110	0	6	0	6	0111000000010000$\overline{0}$0001011		
	0011001111	0	6	0	6	1011100000001000$\overline{0}$0000101		
	1000111100	0	5	1	1	1101110000000100000$\overline{0}$000010		
	7B0000001000	1	1	0	3	0110111000000010000$\overline{0}$00001		
	6B0000000100	1	1	0	3	1011011100000001000$\overline{0}$0000		
	5B0000000010	1	1	0	5	0101101110000000100000$\overline{0}$000		
	4B0000000001	1	1	0	5	0010110111000000010000$\overline{0}$00		
	3B0000000000	1	0	0	4	1001011011100000001000$\overline{0}$0		
	2B0000000000	1	0	0	4	100101101110000000100000		
	1B0000000000	1	0	0	4	10010110111000000010000	0 X^{23}	Bit 0 Corrected
	1000000000	1	1	0	3	1001011011100000001000	0	
	0100000000	1	1	0	5	100101101110000000100	0	
	0010000000	1	1	0	5	10010110111000000010	0 X^{20}	
	0000000000	1	0	0	4	1001011011100000001	0	
	0000000000	1	0	0	4	100101101110000000	0	Bit 18 Corrected
	0000000000	1	0	0	4	10010110111000000	0	
	0000000000	1	0	0	4	1001011011100000	0	
	0000000000	1	0	0	4	100101101110000	0 X^{15}	
	0000000000	1	0	0	4	10010110111000	0	
	0000000000	1	0	0	4	1001011011100	0	
	0000000000	1	0	0	4	100101101110	0	
	0000000000	1	0	0	4	10010110111	0	
	0000000000	1	0	0	4	1001011011	1 X^{10}	

TABLE IX. (continued)

r(X)	s(X)	T_0	Σ_0	T_1	Σ_1	Buffer	$V^*(X)$
	0000000000	1	0	0	4	100101101	1
	0000000000	1	0	0	4	10010110	1
	0000000000	1	0	0	4	1001011	0
	0000000000	1	0	0	4	100101	1
	0000000000	1	0	0	4	10010	1 X^5
	0000000000	1	0	0	4	1001	0
	0000000000	1	0	0	4	100	1
	0000000000	1	0	0	4	10	0
	0000000000	1	0	0	4	1	0
	0000000000	1	0	0	4		1 X^0

SECTION V

BURST ERROR-CORRECTING CODES

Since, in general, all errors do not occur independently and in some cases one error will cause several errors in succession to be made, the correction of several errors in a row, a burst of errors, should be considered. A burst of length ℓ is defined as a group of bits of length ℓ of which at least the first and the $\ell^{\underline{th}}$ bit are in error. A code which is capable of correcting all bursts of errors of length ℓ or less but not all bursts of length $\ell+1$ is classified as an ℓ burst error-correcting code. The burst-correcting ability is the important parameter for burst error-correcting codes. A necessary condition for an (n,k) ℓ burst error-correcting code is that the number of parity check digits be at least 2ℓ, i.e.,

$$n - k \geq 2\ell \tag{60}$$

Codes that satisfy this bound with equality are said to be optimal in that they have the largest burst error-correcting capability. This is not the only parameter that should be considered, however, since it is possible to have an optimal code in the above sense and still have a code with a low rate (ratio of information bits to total bits).

Some best cyclic and shortened cyclic codes for burst error-correction have been found by Kasami[3] by computer searching. These are best in the sense that they have the maximum number of information digits among all shortened cyclic burst ℓ error-correcting codes with a given number of check digits n-k. A list of these codes is presented in Table X. The only disadvantage to these codes is that there are very few in number.

The (n,k) cyclic code generated by[4]

$$g(X) = p(X)\,(1 + X^{2\ell-1}) \tag{61}$$

has burst error-correcting capability ℓ and is classified as a "Fire code". Here

$$n = \mathrm{LCM}(e,\ 2\ell-1) \tag{62}$$

TABLE X. COMPUTER-GENERATED BURST ERROR-CORRECTING CODES

	Code		Generating Polynomial
$n-k-2\ell$	(n,k)	ℓ	$X^{20} \ldots X^{15} \ldots X^{10} \ldots X^{5} \ldots X^{0}$
0	(7,3)	2	1 0 1 1 1
	(15,9)	3	1 0 0 1 1 1 1
	(19,11)	4	1 1 0 0 1 0 1 0 1
	(27,17)	5	1 0 1 1 0 1 1 1 0 0 1
	(34,22)	6	1 1 0 1 0 0 1 1 1 1 0 1 1
	(38,24)	7	1 0 1 0 0 1 1 1 0 1 1 0 1 0 1
	(50,34)	8	1 0 0 1 0 1 0 0 1 0 1 0 1 1 0 0 1
	(56,38)	9	1 1 0 1 0 0 0 1 0 1 1 1 1 1 1 1 0 1 1
	(59,39)	10	1 0 0 0 0 0 0 0 0 0 1 1 0 1 1 1 0 1 0 0 1
1	(15,10)	2	1 0 1 0 1 1
	(27,20)	3	1 0 0 1 0 0 1 1
	(38,29)	4	1 0 0 1 0 1 1 0 0 1
	(48,37)	5	1 0 0 0 0 0 1 0 1 0 0 1
	(67,54)	6	1 0 1 0 1 1 1 1 0 0 1 1 1 1
	(103,88)	7	1 0 0 0 1 1 1 1 0 0 0 1 1 0 0 1
	(96,79)	8	1 0 0 0 0 0 0 0 0 1 0 0 0 0 0 1 0 1
2	(31,25)	2	1 0 0 0 1 1 1
	(63,55)	3	1 0 0 1 0 0 1 1 1
	(85,75)	4	1 0 0 1 0 1 0 1 1 0 1
	(131,119)	5	1 1 0 0 1 1 1 0 0 1 0 1 1
	(169,155)	6	1 0 1 0 1 0 1 1 1 1 0 1 1 0 1
3	(63,56)	2	1 0 1 1 0 1 1 1
	(121,112)	3	1 0 0 1 0 0 0 0 1 1
	(164,153)	4	1 0 1 1 0 1 0 1 0 0 1 1
	(290,277)	5	1 0 0 1 0 0 1 1 1 0 0 1 0 1
4	(511,499)	4	1 0 0 0 1 0 0 1 0 1 0 0 1
5	(1023,1010)	4	1 0 0 1 0 0 1 1 1 1 0 1 0 1

where LCM means "least common multiple" and e is the exponent of p(X) [e is the smallest positive integer such that $X^e + 1$ is divisible by p(X) or equivalently e is the length of the Galois field generated by p(X)] p(X) is an irreducible polynomial of degree m greater than or equal to ℓ. If p(X) is a primitive polynomial, e equals $2^m - 1$. The parity checks associated with the factor $1 + X^{2\ell-1}$ are $2\ell-1$ interlaced and evenly spaced, each of which will be affected by no more than one error in any burst of length $2\ell-1$ or less. Any burst of length b or less will leave at least b-1 successive parity checks unaffected, which determines the symbol at the beginning of the burst. Thus, the factor $1 + X^{2\ell-1}$ is sufficient to determine completely the error pattern for bursts of length no greater than b. The location of the burst is then provided by the factor p(X).

The number of parity bits of the Fire codes are

$$n-k = m \div 2\ell-1 \tag{63}$$

and the inefficiency factor is given as

$$n-k - 2\ell = m - 1 \tag{64}$$

For comparison with the computer-generated burst error-correcting codes, some comparable Fire codes are presented in Table XI.

The (n,k) cyclic code generated by

$$g(X) = p(X^\lambda)(1 + X^{\lambda m}) \tag{65}$$

has burst error capability

$$\ell = (\lambda-1)m + 1 \tag{66}$$

and is classified as a "Burton code". The length is given as

$$n = \lambda \cdot \text{LCM}\ (e,m) = \lambda\sigma m \tag{67}$$

where e is the exponent of p(X), an irreducible polynomial of degree m, and is the interlacing parameter. If p(X) is primitive and e and m do not have any common factors, σ equals e which equals 2^m-1. For the Burton code the number of parity bits is

$$n - k = 2\lambda m \tag{68}$$

TABLE XI. FIRE CODES

Code			Generating Polynomial
$n-k-2\ell$	(n,k)	ℓ	$X^{20} \ldots X^{15} \ldots X^{10} \ldots X^{5} \ldots X^{0}$
2	(21,15)	2	1 0 1 0 0 1 1
	(35,27)	3	1 0 1 1 0 1 0 1 1
3	(15,8)	2	1 0 0 0 1 0 1 1
	(15,6)	3	1 0 0 1 1 1 0 0 1 1
	(105,94)	4	1 0 0 1 1 0 0 1 0 0 1 1
4	(93,85)	2	1 0 0 0 0 1 1 0 1
	(155,145)	3	1 0 0 1 0 0 0 0 1 0 1
	(217,205)	4	1 0 0 1 0 1 0 1 0 0 1 0 1
	(279,265)	5	1 0 0 1 0 1 0 0 0 1 0 0 1 0 1
5	(63,54)	2	1 0 0 1 0 1 1 0 1 1
	(315,304)	3	1 0 0 0 0 0 1 0 0 0 1 1
	(63,50)	4	1 0 0 0 0 1 1 1 0 0 0 0 1 1
	(63,48)	5	1 0 0 0 0 1 1 0 0 1 0 0 0 0 1 1
	(693,676)	6	1 0 0 0 0 1 1 0 0 0 0 1 0 0 0 0 1 1
6	(381,371)	2	1 0 0 1 1 0 0 0 0 0 1
	(635,623)	3	1 0 0 0 1 1 0 1 0 1 0 0 1
	(889,875)	4	1 0 0 0 1 0 0 0 0 0 0 0 1 0 0 1
	(1143,1127)	5	1 0 0 0 1 0 0 1 0 1 0 0 0 1 0 0 1
	(1397,1379)	6	1 0 0 0 1 0 0 1 0 0 0 1 0 0 0 1 0 0 1
	(1651,1631)	7	1 0 0 0 1 0 0 1 0 0 0 0 0 1 0 0 0 1 0 0 1
7	(255,244)	2	1 0 0 1 1 1 1 1 1 0 1 0 1
	(255,242)	3	1 0 0 0 1 0 1 0 1 1 1 1 1 0 1
	(1785,1770)	4	1 0 0 0 1 1 1 1 1 0 0 1 1 1 0 1
	(765,748)	5	1 0 0 0 1 1 1 0 1 1 0 0 0 1 1 1 0 1
	(2805,2786)	6	1 0 0 0 1 1 1 0 1 0 0 1 0 0 0 1 1 1 0 1
8	(1533,1521)	2	1 0 0 1 0 1 0 0 1 1 0 0 1
	(2555,2541)	3	1 0 0 0 0 0 0 0 0 1 1 0 0 0 1
	(511,493)	4	1 0 0 0 0 1 0 1 0 1 0 0 1 0 0 0 1
	(4599,4581)	5	1 0 0 0 0 1 0 0 0 0 0 0 0 0 1 0 0 0 1
	(5621,5601)	6	1 0 0 0 0 1 0 0 0 1 0 1 0 0 0 0 1 0 0 0 1

and the inefficiency factor is

$$n - k - 2\ell = 2(m-1) \tag{69}$$

For comparison with the computer-generated and Fire burst error-correcting codes, some comparable Burton codes are presented in Table XII.

Another possibility for cyclic burst error-correction is the interlacing of the best single error-correcting codes, Hamming codes. The (n,k) burst ℓ error-correcting code has generated polynomial

$$g(X) = g_h(X^{\ell}) \tag{70}$$

where $g_h(X)$ is an $m^{\underline{th}}$ order primitive polynomial. For this interlaced Hamming code

$$n = \ell(2^m-1) \tag{71}$$

$$n - k = \ell m \tag{72}$$

and

$$n - k - 2\ell = \ell(m-2) \tag{73}$$

The interlaced Hamming codes are given in Table XIII.

The objective in this project was to obtain the best burst error-correcting capabilities for a fixed word length and fixed number of parity bits. The word lengths considered are 48 and 24 bits, which is a fairly short code, with 8 and 10 bits of parity. Searching Tables X to XIII for the (48,40) code with the longest burst error-correcting capability, it is observed that the computer-generated (63,55), which can be shortened to the proper length, can correct bursts of length 3. The best Fire code that meets these specifications is the (93,85) burst 2 error-correcting code. Neither the Burton or the interlaced Hamming have any codes that meet these specifications. For a (24,16) code the computer code yields the same (63,55) ℓ equals 3 code, the Fire code the (35,27) ℓ equals 3 code, the interlaced Hamming the (30,22) ℓ equals 2 code, and still no Burton code. For a (48,38) code the computer code yields the (85,75) ℓ equals 4 code, the Fire code the (155,145) ℓ equals 3, the interlaced Hamming the (62,52) ℓ equals 2 code, and no Burton code. Finally, for a (24,14) code the

TABLE XII. BURTON CODES

	Code		Generating Polynomial
n-k-2ℓ	(n,k)	ℓ	x^{20} ... x^{15} ... x^{10} ... x^{5} ... x^{0}
2	(12,4)	3	1 0 1 0 0 0 1 0 1
	(18,6)	5	1 0 0 1 0 0 0 0 0 1 0 0 1
	(24,8)	7	1 0 0 0 1 0 0 0 0 0 0 0 1 0 0 0 1
4	(42,30)	4	1 0 0 0 1 0 0 0 0 0 1 0 1
	(63,45)	7	1 0 0 0 0 0 1 0 0 0 0 0 0 0 1 0 0 1
6	(120,104)	5	1 0 0 0 0 0 1 0 0 0 0 0 0 1 0 1
6	(310,290)	6	1 0 0 0 0 0 1 0 0 0 0 0 0 0 0 1 0 0 0 1

TABLE XIII. INTERLACED HAMMING BURST ERROR-CORRECTING CODES

	Code		Generating Polynomial
n-k-2ℓ	(n,k)	ℓ	x^{20} ... x^{15} ... x^{10} ... x^{5} ... x^{0}
2	(14,8)	2	1 0 0 0 1 0 1
3	(21,12)	3	1 0 0 0 0 0 1 0 0 1
4	(30,22)	2	1 0 0 0 0 0 1 0 1
	(28,16)	4	1 0 0 0 0 0 0 0 1 0 0 0 1
5	(35,20)	5	1 0 0 0 0 0 0 0 0 0 1 0 0 0 0 1
6	(62,52)	2	1 0 0 0 0 0 1 0 0 0 1
	(45,33)	3	1 0 0 0 0 0 0 0 0 1 0 0 1
	(42,24)	6	1 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 1
8	(60,44)	4	1 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 1

computer code yields the (27,17) ℓ equals 5 code, the Fire code the (155,145) ℓ equals 3 code, the interlaced Hamming the (62,52) ℓ equals 2 code, and no Burton code. The computer-generated codes in all cases were as good as, or better than, the other codes, as would be expected. It was hoped that the noncomputer-generated codes would have been as good as the computer-generated ones since the noncomputer-generated codes are easier to obtain. The only case where they had the same burst error-correcting capability was the (24,16) computer-generated code and the corresponding Fire code with ℓ equals 3. It should be noted that the Burton and the interlaced Hamming codes were not even close to the performance of the computer-generated codes for the short code lengths considered here. Also, in some instances the Fire codes were close to the performance of the computer-generated codes.

To illustrate the correctable error patterns of these burst error-correcting codes, the correspondence between the observed syndromes and the correctable errors for six of these codes have been tabulated in Tables XIV to XIX.

Tables XIV and XV show burst 3 error-correcting codes. The computer-generated code is longer and thus uses more of the available correctable error patterns. The computer-generated code uses $63 \cdot 4 = 252$ of the correctable error patterns while the Fire code uses $35 \cdot 4 = 140$ out of the possible 255 correctable error patterns. As illustrated in Table XIV, the syndrome patterns have been broken into the corresponding burst patterns of length one, 1, length two, 11, and length three, 101 and 111. Each entry is a syndrome. The last syndrome in each division of the table is the burst pattern, e.g., syndrome 63 is the burst pattern of length one, and syndrome 126 is the burst pattern of length two. Similarly, in Table XV for the Fire code, syndrome 35 is the burst pattern of length one and syndrome 70 is the burst pattern of length two. These patterns also correspond to correcting that particular burst at the high end of the word. Working backwards in the same division corrects the same burst at less significant digits in the word, and the first entry in the division corrects this burst at the first bit in the word. For example, in Table XV, syndrome 184 corresponds to the error pattern $X^{55} + X^{57}$, and syndrome 129 corresponds to the error pattern $1 + X^2$.

TABLE XIV. (63,55) ℓ=3 COMPUTER GENERATED CODE

IG=11100100

1 11100100	51 10000011	101 00100101	151 10110010
2 01110010	52 10100101	102 11110110	152 01011001
3 00111001	53 10110110	103 01111011	153 11001000
4 11111000	54 01011011	104 11011001	154 01100100
5 01111100	55 11001001	105 10001000	155 00110010
6 00111110	56 10000000	106 01000100	156 00011001
7 00011111	57 01000000	107 00100010	157 11101000
8 11101011	58 00100000	108 00010001	158 01110100
9 10010001	59 00010000	109 11101100	159 00111010
10 10101100	60 00001000	110 01110110	160 00011101
11 01010110	61 00000100	111 00111011	161 11101010
12 00101011	62 00000010	112 11111001	162 01110101
13 11110001	63 00000001	113 10011000	163 11011110
14 10011100	64 11100101	114 01001100	164 01101111
15 01001110	65 10010110	115 00100110	165 11010011
16 00100111	66 01001011	116 00010011	166 10001101
17 11110111	67 11000001	117 11101101	167 10100110
18 10011111	68 10000100	118 10010010	168 01010001
19 10101011	69 01000010	119 01001001	169 11001100
20 10110001	70 00100001	120 11000000	170 01100110
21 10111100	71 11110100	121 01100000	171 00110011
22 01011110	72 01111010	122 00110000	172 11111101
23 00101111	73 00111101	123 00011000	173 10011010
24 11110011	74 11111010	124 00001100	174 01001101
25 10011101	75 01111101	125 00000110	175 11000010
26 10101010	76 11011010	126 00000011	176 01100001
27 01010101	77 01101101	127 11100110	177 11010100
28 11001110	78 11010010	128 01110011	178 01101010
29 01100111	79 01101001	129 11011101	179 00110101
30 11010111	80 11010000	130 10001010	180 11111110
31 10001111	81 01101000	131 01000101	181 01111111
32 10100011	82 00110100	132 11000110	182 11011011
33 10110101	83 00011010	133 01100011	183 10001001
34 10111110	84 00001101	134 11010101	184 10100000
35 01011111	85 11100010	135 10001110	185 01010000
36 11001011	86 01110001	136 01000111	186 00101000
37 10000001	87 11011100	137 11000111	187 00010100
38 10100100	88 01101110	138 10000111	188 00001010
39 01010010	89 00110111	139 10100111	189 00000101
40 00101001	90 11111111	140 10110111	190 11100111
41 11110000	91 10011011	141 10111111	191 10010111
42 01111000	92 10101001	142 10111011	192 10101111
43 00111100	93 10110000	143 10111001	193 10110011
44 00011110	94 01011000	144 10111000	194 10111101
45 00001111	95 00101100	145 01011100	195 10111010
46 11100011	96 00010110	146 00101110	196 01011101
47 10010101	97 00001011	147 00010111	197 11001010
48 10101110	98 11100001	148 11101111	198 01100101
49 01010111	99 10010100	149 10010011	199 11010110
50 11001111	100 01001010	150 10101101	200 01101011

TABLE XIV. (continued)

201 11010001	214 01000001	227 11101110	240 00011011
202 10001100	215 11000100	228 01110111	241 11101001
203 01000110	216 01100010	229 11011111	242 10010000
204 00100011	217 00110001	230 10001011	243 01001000
205 11110101	218 11111100	231 10100001	244 00100100
206 10011110	219 01111110	232 10110100	245 00010010
207 01001111	220 00111111	233 01011010	246 00001001
208 11000011	221 11111011	234 00101101	247 11100000
209 10000101	222 10011001	235 11110010	248 01110000
210 10100110	223 10101000	236 01111001	249 00111000
211 01010011	224 01010100	237 11011000	250 00011100
212 11001101	225 00101010	238 01101100	251 00001110
213 10000010	226 00010101	239 00110110	252 00000111

TABLE XV. (35,27) ℓ =3 FIRE CODE

IG=110101101

1 11010110	51 11110110	101 01010000	151 01110110
2 01101011	52 01111011	102 00101000	152 00111011
3 11100011	53 11101011	103 00010100	153 11001011
4 10100111	54 10100011	104 00001010	154 10110011
5 10000101	55 10000111	105 00000101	155 10001111
6 10010100	56 10010101	106 11010101	156 10010001
7 01001010	57 10011100	107 10111100	157 10011110
8 00100101	58 01001110	108 01011110	158 01001111
9 11000100	59 00100111	109 00101111	159 11110001
10 01100010	60 11000101	110 11000001	160 10101110
11 00110001	61 10110100	111 10110110	161 01010111
12 11001110	62 01011010	112 01011011	162 11111101
13 01100111	63 00101101	113 11111011	163 10101000
14 11100101	64 11000000	114 10101011	164 01010100
15 10100100	65 01100000	115 10000011	165 00101010
16 01010010	66 00110000	116 10010111	166 00010101
17 00101001	67 00011000	117 10011101	167 11011100
18 11000010	68 00001100	118 10011000	168 01101110
19 01100001	69 00000110	119 01001100	169 00110111
20 11100110	70 00000011	120 00100110	170 11001101
21 01110011	71 11010100	121 00010011	171 10110000
22 11101111	72 01101010	122 11011111	172 01011000
23 10100001	73 00110101	123 10111001	173 00101100
24 10000110	74 11001100	124 10001010	174 00010110
25 01000011	75 01100110	125 01000101	175 00001011
26 11110111	76 00110011	126 11110100	176 11010000
27 10101101	77 11001111	127 01111010	177 01101000
28 10000000	78 10110001	128 00111101	178 00110100
29 01000000	79 10001110	129 11001000	179 00011010
30 00100000	80 01000111	130 01100100	180 00001101
31 00010000	81 11110101	131 00110010	181 11010001
32 00001000	82 10101100	132 00011001	182 10111110
33 00000100	83 01010110	133 11011010	183 01011111
34 00000010	84 00101011	134 01101101	184 11111001
35 00000001	85 11000011	135 11100000	185 10101010
36 11010111	86 10110111	136 01110000	186 01010101
37 10111101	87 10001101	137 00111000	187 11111100
38 10001000	88 10010000	138 00011100	188 01111110
39 01000100	89 01001000	139 00001110	189 00111111
40 00100010	90 00100100	140 00000111	190 11001001
41 00010001	91 00010010	141 11010011	191 10110010
42 11011110	92 00001001	142 10111111	192 01011001
43 01101111	93 11010010	143 10001001	193 11111010
44 11100001	94 01101001	144 10010010	194 01111101
45 10100110	95 11100010	145 01001001	195 11101000
46 01010011	96 01110001	146 11110010	196 01110100
47 11111111	97 11101110	147 01111001	197 00111010
48 10101001	98 01110111	148 11101010	198 00011101
49 10000010	99 11101101	149 01110101	199 11011000
50 01000001	100 10100000	150 11101100	200 01101100

TABLE XV. (35,27) ℓ =3 FIRE CODE

201 00110110	205 10001011	209 10011010	213 00111100
202 00011011	206 10010011	210 01001101	214 00011110
203 11011011	207 10011111	211 11110000	215 00001111
204 10111011	208 10011001	212 01111000	

TABLE XVI. (85,75) ℓ=4 COMPUTER GENERATED CODE

IG=10110101001

1 1011010100	51 1011111100	101 0100000110	151 0100100100
2 0101101010	52 0101111110	102 0010000011	152 0010010010
3 0010110101	53 0010111111	103 1010010101	153 0001001001
4 1010001110	54 1010001011	104 1110011110	154 1011110000
5 0101000111	55 1110010001	105 0111001111	155 0101111000
6 1001110111	56 1100011100	106 1000110011	156 0010111100
7 1111101111	57 0110001110	107 1111001101	157 0001011110
8 1100100011	58 0011000111	108 1100110010	158 0000101111
9 1101000101	59 1010110111	109 0110011001	159 1011000011
10 1101110110	60 1110001111	110 1000011000	160 1110110101
11 0110111011	61 1100010011	111 0100001100	161 1100001110
12 1000001001	62 1101011101	112 0010000110	162 0110000111
13 1111010000	63 1101111010	113 0001000011	163 1000010111
14 0111101000	64 0110111101	114 1011110101	164 1111011111
15 0011110100	65 1000001010	115 1110101110	165 1100111011
16 0001111010	66 0100000101	116 0111010111	166 1101001001
17 0000111101	67 1001010110	117 1000111111	167 1101110000
18 1011001010	68 0100101011	118 1111001011	168 0110111000
19 0101100101	69 1001000001	119 1100110001	169 0011011100
20 1001100110	70 1111110100	120 1101001100	170 0001101110
21 0100110011	71 0111111010	121 0110100110	171 0000110111
22 1001001101	72 0011111101	122 0011010011	172 1011001111
23 1111110010	73 1010101010	123 1010111101	173 1110110011
24 0111111001	74 0101010101	124 1110001010	174 1100001101
25 1000101000	75 1001111110	125 0111000101	175 1101010010
26 0100010100	76 0100111111	126 1000110110	176 0110101001
27 0010001010	77 1001001011	127 0100011011	177 1000000000
28 0001000101	78 1111110001	128 1001011001	178 0100000000
29 1011110110	79 1100101100	129 1111111000	179 0010000000
30 0101111011	80 0110010110	130 0111111100	180 0001000000
31 1001101001	81 0011001011	131 0011111110	181 0000100000
32 1111100000	82 1010110001	132 0001111111	182 0000010000
33 0111110000	83 1110001100	133 1011101011	183 0000001000
34 0011111000	84 0111000110	134 1110100001	184 0000000100
35 0001111100	85 0011100011	135 1100000100	185 0000000010
36 0000111110	86 1010100101	136 0110000010	186 0000000001
37 0000011111	87 1110000110	137 0011000001	187 1011010101
38 1011011011	88 0111000011	138 1010110100	188 1110111110
39 1110111001	89 1000110101	139 0101011010	189 0111011111
40 1100001000	90 1111001110	140 0010101101	190 1000111011
41 0110000100	91 0111100111	141 1010000010	191 1111001001
42 0011000010	92 1000100111	142 0101000001	192 1100110000
43 0001100001	93 1111000111	143 1001110100	193 0110011000
44 1011100100	94 1100110111	144 0100111010	194 0011001100
45 0101110010	95 1101001111	145 0010011101	195 0001100110
46 0010111001	96 1101110011	146 1010011010	196 0000110011
47 1010001000	97 1101101101	147 0101001101	197 1011001101
48 0101000100	98 1101100010	148 1001110010	198 1110110010
49 0010100010	99 0110110001	149 0100111001	199 0111011001
50 0001010001	100 1000001100	150 1001001000	200 1000111000

TABLE XVI. (continued)

201 0100011100	251 1110110111	301 0101011011	351 0011100100
202 0010001110	252 1100001111	302 1001111001	352 0001110010
203 0001000111	253 1101010011	303 1111101000	353 0000111001
204 1011110111	254 1101111101	304 0111110100	354 1011001000
205 1110101111	255 1101101010	305 0011111010	355 0101100100
206 1100000011	256 0110110101	306 0001111101	356 0010110010
207 1101010101	257 1000001110	307 1011101010	357 0001011001
208 1101111110	258 0100000111	308 0101110101	358 1011111000
209 0110111111	259 1001010111	309 1001101110	359 0101111100
210 1000001011	260 1111111111	310 0100110111	360 0010111110
211 1111010001	261 1100101011	311 1001001111	361 0001011111
212 1100111100	262 1101000001	312 1111110011	362 1011111011
213 0110011110	263 1101110100	313 1100101101	363 1110101001
214 0011001111	264 0110111010	314 1101000010	364 1100000000
215 1010110011	265 0011011101	315 0110100001	365 0110000000
216 1110001101	266 1010111010	316 1000000100	366 0011000000
217 1100010010	267 0101011101	317 0100000010	367 0001100000
218 0110001001	268 1001111010	318 0010000001	368 0000110000
219 1000010000	269 0100111101	319 1010010100	369 0000011000
220 0100001000	270 1001001010	320 0101001010	370 0000001100
221 0010000100	271 0100100101	321 0010100101	371 0000000110
222 0001000010	272 1001000110	322 1010000110	372 0000000011
223 0000100001	273 0100100011	323 0101000011	373 1011010110
224 1011000100	274 1001000101	324 1001110101	374 0101101011
225 0101100010	275 1111110110	325 1111101110	375 1001100001
226 0010110001	276 0111111011	326 0111110111	376 1111100100
227 1010001100	277 1000101001	327 1000101111	377 0111110010
228 0101000110	278 1111000000	328 1111000011	378 0011111001
229 0010100011	279 0111100000	329 1100110101	379 1010101000
230 1010000101	280 0011110000	330 1101001110	380 0101010100
231 1110010110	281 0001111000	331 0110100111	381 0010101010
232 0111001011	282 0000111100	332 1000000111	382 0001010101
233 1000110001	283 0000011110	333 1111010111	383 1011111110
234 1111001100	284 0000001111	334 1100111111	384 0101111111
235 0111100110	285 1011010011	335 1101001011	385 1001101011
236 0011110011	286 1110111101	336 1101110001	386 1111100001
237 1010101101	287 1100001010	337 1101101100	387 1100100100
238 1110000010	288 0110000101	338 0110110110	388 0110010010
239 0111000001	289 1000010110	339 0011011011	389 0011001001
240 1000110100	290 0100001011	340 1010111001	390 1010110000
241 0100011010	291 1001010001	341 1110001000	391 0101011000
242 0010001101	292 1111111100	342 0111000100	392 0010101100
243 1010010010	293 0111111110	343 0011100010	393 0001010110
244 0101001001	294 0011111111	344 0001110001	394 0000101011
245 1001110000	295 1010101011	345 1011101100	395 1011000001
246 0100111000	296 1110000001	346 0101110110	396 1110110100
247 0010011100	297 1100010100	347 0010111011	397 0111011010
248 0001001110	298 0110001010	348 1010001001	398 0011101101
249 0000100111	299 0011000101	349 1110010000	399 1010100010
250 1011000111	300 1010110110	350 0111001000	400 0101010001

TABLE XVI. (continued)

401 1001111100	451 0101100111	501 1011100011	551 1010000000
402 0100111110	452 1001100111	502 1110100101	552 0101000000
403 0010011111	453 1111100111	503 1100000110	553 0010100000
404 1010011011	454 1100100111	504 0110000011	554 0001010000
405 1110011001	455 1101000111	505 1000010101	555 0000101000
406 1100011000	456 1101110111	506 1111011110	556 0000010100
407 0110001100	457 1101101111	507 0111101111	557 0000001010
408 0011000110	458 1101100011	508 1000100011	558 0000000101
409 0001100011	459 1101100101	509 1111000101	559 1011010111
410 1011100101	460 1101100110	510 1100110110	560 1110111111
411 1110100110	461 0110110011	511 0110011011	561 1100001011
412 0111010011	462 1000001101	512 1000011001	562 1101010001
413 1000111101	463 1111010010	513 1111011000	563 1101111100
414 1111001010	464 0111101001	514 0111101100	564 0110111110
415 0111100101	465 1000100000	515 0011110110	565 0011011111
416 1000100110	466 0100010000	516 0001111011	566 1010111011
417 0100010011	467 0010001000	517 1011101001	567 1110001001
418 1001011101	468 0001000100	518 1110100000	568 1100010000
419 1111111010	469 0000100010	519 0111010000	569 0110001000
420 0111111101	470 0000010001	520 0011101000	570 0011000100
421 1000101010	471 1011011100	521 0001110100	571 0001100010
422 0100010101	472 0101101110	522 0000111010	572 0000110001
423 1001011110	473 0010110111	523 0000011101	573 1011001100
424 0100101111	474 1010001111	524 1011011010	574 0101100110
425 1001000011	475 1110010011	525 0101101101	575 0010110011
426 1111110101	476 1100011101	526 1001100010	576 1010001101
427 1100101110	477 1101011010	527 0100110001	577 1110010010
428 0110010111	478 0110101101	528 1001001100	578 0111001001
429 1000011111	479 1000000010	529 0100100110	579 1000110000
430 1111011011	480 0100000001	530 0010010011	580 0100011000
431 1100111001	481 1001010100	531 1010011101	581 0010001100
432 1101001000	482 0100101010	532 1110011010	582 0001000110
433 0110100100	483 0010010101	533 0111001101	583 0000100011
434 0011010010	484 1010011110	534 1000110010	584 1011000101
435 0001101001	485 0101001111	535 0100011001	585 1110110110
436 1011100000	486 1001110011	536 1001011000	586 0111011011
437 0101110000	487 1111101101	537 0100101100	587 1000111001
438 0010111000	488 1100100010	538 0010010110	588 1111001000
439 0001011100	489 0110010001	539 0001001011	589 0111100100
440 0000101110	490 1000011100	540 1011110001	590 0011110010
441 0000010111	491 0100001110	541 1110101100	591 0001111001
442 1011011111	492 0010000111	542 0111010110	592 1011101000
443 1110111011	493 1010010111	543 0011101011	593 0101110100
444 1100001001	494 1110011111	544 1010100001	594 0010111010
445 1101010000	495 1100011011	545 1110000100	595 0001011101
446 0110101000	496 1101011001	546 0111000010	596 1011111010
447 0011010100	497 1101111000	547 0011100001	597 0101111101
448 0001101010	498 0110111100	548 1010100100	598 1001101010
449 0000110101	499 0011011110	549 0101010010	599 0100110101
450 1011001110	500 0001101111	550 0010101001	600 1001001110

TABLE XVI. (continued)

601 0100100111	651 0000000111	701 1011011001	751 1100010110
602 1001000111	652 1011010000	702 1110111000	752 0110001011
603 1111110111	653 0101101000	703 0111011100	753 1000010001
604 1100101111	654 0010110100	704 0011101110	754 1111011100
605 1101000011	655 0001011010	705 0001110111	755 0111101110
606 1101110101	656 0000101101	706 1011101111	756 0011110111
607 1101101110	657 1011000010	707 1110100011	757 1010101111
608 0110110111	658 0101100001	708 1100000101	758 1110000011
609 1000001111	659 1001100100	709 1101010110	759 1100010101
610 1111010011	660 0100110010	710 0110101011	760 1101011110
611 1100111101	661 0010011001	711 1000000001	761 0110101111
612 1101001010	662 1010011000	712 1111010100	762 1000000011
613 0110100101	663 0101001100	713 0111101010	763 1111010101
614 1000000110	664 0010100110	714 0011110101	764 1100111110
615 0100000011	665 0001010011	715 1010101110	765 0110011111
616 1001010101	666 1011111101	716 0101010111	766 1000011011
617 1111111110	667 1110101010	717 1001111111	767 1111011001
618 0111111111	668 0111010101	718 1111101011	768 1100111000
619 1000101011	669 1000111110	719 1100100001	769 0110011100
620 1111000001	670 0100011111	720 1101000100	770 0011001110
621 1100110100	671 1001011011	721 0110100010	771 0001100111
622 0110011010	672 1111111001	722 0011010001	772 1011100111
623 0011001101	673 1100101000	723 1010111100	773 1110100111
624 1010110010	674 0110010100	724 0101011110	774 1100000111
625 0101011001	675 0011001010	725 0010101111	775 1101010111
626 1001111000	676 0001100101	726 1010000011	776 1101111111
627 0100111100	677 1011100110	727 1110010101	777 1101101011
628 0010011110	678 0101110011	728 1100011110	778 1101100001
629 0001001111	679 1001101101	729 0110001111	779 1101100100
630 1011110011	680 1111100010	730 1000010011	780 0110110010
631 1110101101	681 0111110001	731 1111011101	781 0011011001
632 1100000010	682 1000101100	732 1100111010	782 1010111000
633 0110000001	683 0100010110	733 0110011101	783 0101011100
634 1000010100	684 0010001011	734 1000011010	784 0010101110
635 0100001010	685 1010010001	735 0100001101	785 0001010111
636 0010000101	686 1110011100	736 1001010010	786 1011111111
637 1010010110	687 0111001110	737 0100101001	787 1110101011
638 0101001011	688 0011100111	738 1001000000	788 1100000001
639 1001110001	689 1010100111	739 0100100000	789 1101010100
640 1111101100	690 1110000111	740 0010010000	790 0110101010
641 0111110110	691 1100010111	741 0001001000	791 0011010101
642 0011111011	692 1101011111	742 0000100100	792 1010111110
643 1010101001	693 1101111011	743 0000010010	793 0101011111
644 1110000000	694 1101101001	744 0000001001	794 1001111011
645 0111000000	695 1101100000	745 1011010001	795 1111101001
646 0011100000	696 0110110000	746 1110111100	796 1100100000
647 0001110000	697 0011011000	747 0111011110	797 0110010000
648 0000111000	698 0001101100	748 0011101111	798 0011001000
649 0000011100	699 0000110110	749 1010100011	799 0001100100
650 0000001110	700 0000011011	750 1110000101	800 0000110010

TABLE XVI. (continued)

801 0000011001	851 1001100000	901 1000010010	951 0001101011
802 1011011000	852 0100110000	902 0100001001	952 1011100001
803 0101101100	853 0010011000	903 1001010000	953 1110100100
804 0010110110	854 0001001100	904 0100101000	954 0111010010
805 0001011011	855 0000100110	905 0010010100	955 0011101001
806 1011111001	856 0000010011	906 0001001010	956 1010100000
807 1110101000	857 1011011101	907 0000100101	957 0101010000
808 0111010100	858 1110111010	908 1011000110	958 0010101000
809 0011101010	859 0111011101	909 0101100011	959 0001010100
810 0001110101	860 1000111010	910 1001100101	960 0000101010
811 1011101110	861 0100011101	911 1111100110	961 0000010101
812 0101110111	862 1001011010	912 0111110011	
813 1001101111	863 0100101101	913 1000101101	
814 1111100011	864 1001000010	914 1111000010	
815 1100100101	865 0100100001	915 0111100001	
816 1101000110	866 1001000100	916 1000100100	
817 0110100011	867 0100100010	917 0100010010	
818 1000000101	868 0010010001	918 0010001001	
819 1111010110	869 1010011100	919 1010010000	
820 0111101011	870 0101001110	920 0101001000	
821 1000100001	871 0010100111	921 0010100100	
822 1111000100	872 1010000111	922 0001010010	
823 0111100010	873 1110010111	923 0000101001	
824 0011110001	874 1100011111	924 1011000000	
825 1010101100	875 1101011011	925 0101100000	
826 0101010110	876 1101111001	926 0010110000	
827 0010101011	877 1101101000	927 0001011000	
828 1010000001	878 0110110100	928 0000101100	
829 1110010100	879 0011011010	929 0000010110	
830 0111001010	880 0001101101	930 0000001011	
831 0011100101	881 1011100010	931 1011011110	
832 1010100110	882 0101110001	932 0101101111	
833 0101010011	883 1001101100	933 1001100011	
834 1001111101	884 0100110110	934 1111100101	
835 1111101010	885 0010011011	935 1100100110	
836 0111110101	886 1010011001	936 0110010011	
837 1000101110	887 1110011000	937 1000011101	
838 0100010111	888 0111001100	938 1111011010	
839 1001011111	889 0011100110	939 0111101101	
840 1111111011	890 0001110011	940 1000100010	
841 1100101001	891 1011101101	941 0100010001	
842 1101000000	892 1110100010	942 1001011100	
843 0110100000	893 0111010001	943 0100101110	
844 0011010000	894 1000111100	944 0010010111	
845 0001101000	895 0100011110	945 1010011111	
846 0000110100	896 0010001111	946 1110011011	
847 0000011010	897 1010010011	947 1100011001	
848 0000001101	898 1110011101	948 1101011000	
849 1011010010	899 1100011010	949 0110101100	
850 0101101001	900 0110001101	950 0011010110	

TABLE XVII. (27,17) ℓ =5 COMPUTER GENERATED CODE

IG=10011101101
1 1001110110
2 0100111011
3 1011101011
4 1100000011
5 1111110111
6 1110001101
7 1110110000
8 0111011000
9 0011101100
10 0001110110
11 0000111011
12 1001101011
13 1101000011
14 1111010111
15 1110011101
16 1110111000
17 0111011100
18 0011101110
19 0001110111
20 1001001101
21 1101010000
22 0110101000
23 0011010100
24 0001101010
25 0000110101
26 1001101100
27 0100110110
28 0010011011
29 1000111011
30 1101101011
31 1111000011
32 1110010111
33 1110111101
34 1110101000
35 0111010100
36 0011101010
37 0001110101
38 1001001100
39 0100100110
40 0010010011
41 1000111111
42 1101101001
43 1111000010
44 0111100001
45 1010000110
46 0101000011
47 1011010111
48 1100011101
49 1111111000
50 0111111100
51 0011111110
52 0001111111
53 1001001001
54 1101010010
55 0110101001
56 1010100010
57 0101010001
58 1011011110
59 0101101111
60 1011000001
61 1100010110
62 0110001011
63 1010110011
64 1100101111
65 1111100001
66 1110000110
67 0111000011
68 1010010111
69 1100111101
70 1111101000
71 0111110100
72 0011111010
73 0001111101
74 1001001000
75 0100100100
76 0010010010
77 0001001001
78 1001010010
79 0100101001
80 1011100010
81 0101110001
82 1011001110
83 0101100111
84 1011000101
85 1100010100
86 0110001010
87 0011000101
88 1000010100
89 0100001010
90 0010000101
91 1000110100
92 0100011010
93 0010001101
94 1000110000
95 0100011000
96 0010001100
97 0001000110
98 0000100011
99 1001100111
100 1101000101
101 1111010100
102 0111101010
103 0011110101
104 1000001100
105 0100000110
106 0010000011
107 1000110111
108 1101101101
109 1111000000
110 0111100000
111 0011110000
112 0001111000
113 0000111100
114 0000011110
115 0000001111
116 1001110001
117 1101001110
118 0110100111
119 1010100101
120 1100100100
121 0110010010
122 0011001001
123 1000010010
124 0100001001
125 1011110010
126 0101111001
127 1011001010
128 0101100101
129 1011000100
130 0101100010
131 0010110001
132 1000101110
133 0100010111
134 1011111101
135 1100001000
136 0110000100
137 0011000010
138 0001100001
139 1001000110
140 0100100011
141 1011100111
142 1100000101
143 1111110100
144 0111111010
145 0011111101
146 1000001000
147 0100000100
148 0010000010
149 0001000001
150 1001010110
151 0100101011
152 1011100011
153 1100000111
154 1111110101
155 1110001100
156 0111000110
157 0011100011
158 1000000111
159 1101110101
160 1111001100
161 0111100110
162 0011110011
163 1000001111
164 1101110001
165 1111001110
166 0111100111
167 1010000101
168 1100110100
169 0110011010
170 0011001101
171 1000010000
172 0100001000
173 0010000100
174 0001000010
175 0000100001
176 1001100110
177 0100110011
178 1011101111
179 1100000001
180 1111110110
181 0111111011
182 1010001011
183 1100110011
184 1111101111
185 1110000001
186 1110110110
187 0111011011
188 1010011011
189 1100111011
190 1111101011
191 1110000011
192 1110110111
193 1110101101
194 1110100000
195 0111010000
196 0011101000
197 0001110100
198 0000111010
199 0000011101
200 1001111000

TABLE XVII. (27,17) ℓ =5 COMPUTER GENERATED

201 0100111100	251 0101100110	301 0011101011	351 0010011010
202 0010011110	252 0010110011	302 1000000011	352 0001001101
203 0001001111	253 1000101111	303 1101110111	353 1001010000
204 1001010001	254 1101100001	304 1111001101	354 0100101000
205 1101011110	255 1111000110	305 1110010000	355 0010010100
206 0110101111	256 0111100011	306 0111001000	356 0001001010
207 1010100001	257 1010000111	307 0011100100	357 0000100101
208 1100100110	258 1100110101	308 0001110010	358 1001100100
209 0110010011	259 1111101100	309 0000111001	359 0100110010
210 1010111111	260 0111110110	310 1001101010	360 0010011001
211 1100101001	261 0011111011	311 0100110101	361 1000111010
212 1111100010	262 1000001011	312 1011101100	362 0100011101
213 0111110001	263 1101110011	313 0101110110	363 1011111000
214 1010001110	264 1111001111	314 0010111011	364 0101111100
215 0101000111	265 1110010001	315 1000101011	365 0010111110
216 1011010101	266 1110111110	316 1101100011	366 0001011111
217 1100011100	267 0111011111	317 1111000111	367 1001011001
218 0110001110	268 1010011001	318 1110010101	368 1101011010
219 0011000111	269 1100111010	319 1110111100	369 0110101101
220 1000010101	270 0110011101	320 0111011110	370 1010100000
221 1101111100	271 1010111000	321 0011101111	371 0101010000
222 0110111110	272 0101011100	322 1000000001	372 0010101000
223 0011011111	273 0010101110	323 1101110110	373 0001010100
224 1000011001	274 0001010111	324 0110111011	374 0000101010
225 1101111010	275 1001011101	325 1010101011	375 0000010101
226 0110111101	276 1101011000	326 1100100011	376 1001111100
227 1010101000	277 0110101100	327 1111100111	377 0100111110
228 0101010100	278 0011010110	328 1110000101	378 0010011111
229 0010101010	279 0001101011	329 1110110100	379 1000111001
230 0001010101	280 1001000011	330 0111011010	380 1101101010
231 1001011100	281 1101010111	331 0011101101	381 0110110101
232 0100101110	282 1111011101	332 1000000000	382 1010101100
233 0010010111	283 1110011000	333 0100000000	383 0101010110
234 1000111101	284 0111001100	334 0010000000	384 0010101011
235 1101101000	285 0011100110	335 0001000000	385 1000100011
236 0110110100	286 0001110011	336 0000100000	386 1101100111
237 0011011010	287 1001001111	337 0000010000	387 1111000101
238 0001101101	288 1101010001	338 0000001000	388 1110010100
239 1001000000	289 1111011110	339 0000000100	389 0111001010
240 0100100000	290 0111101111	340 0000000010	390 0011100101
241 0010010000	291 1010000001	341 0000000001	391 1000000100
242 0001001000	292 1100110110	342 1001110111	392 0100000010
243 0000100100	293 0110011011	343 1101001101	393 0010000001
244 0000010010	294 1010111011	344 1111010000	394 1000110110
245 0000001001	295 1100101011	345 0111101000	395 0100011011
246 1001110010	296 1111100011	346 0011110100	396 1011111011
247 0100111001	297 1110000111	347 0001111010	397 1100001011
248 1011101010	298 1110110101	348 0000111101	398 1111110011
249 0101110101	299 1110101100	349 1001101000	399 1110001111
250 1011001100	300 0111010110	350 0100110100	400 1110110001

TABLE XVII. (CONTINUED)

401 1110101110	451 1000100000	501 0010111001	551 1100101100
402 0111010111	452 0100010000	502 1000101010	552 0110010110
403 1010011101	453 0010001000	503 0100010101	553 0011001011
404 1100111000	454 0001000100	504 1011111100	554 1000010011
405 0110011100	455 0000100010	505 0101111110	555 1101111111
406 0011001110	456 0000010001	506 0010111111	556 1111001001
407 0001100111	457 1001111110	507 1000101001	557 1110010010
408 1001000101	458 0100111111	508 1101100010	558 0111001001
409 1101010100	459 1011101001	509 0110110001	559 1010010010
410 0110101010	460 1100000010	510 1010101110	560 0101001001
411 0011010101	461 0110000001	511 0101010111	561 1011010010
412 1000011100	462 1010110110	512 1011011101	562 0101101001
413 0100001110	463 0101011011	513 1100011000	563 1011000010
414 0010000111	464 1011011011	514 0110001100	564 0101100001
415 1000110101	465 1100011011	515 0011000110	565 1011000110
416 1101101100	466 1111111011	516 0001100011	566 0101100011
417 0110110110	467 1110001011	517 1001000111	567 1011000111
418 0011011011	468 1110110011	518 1101010101	568 1100010101
419 1000011011	469 1110101111	519 1111011100	569 1111111100
420 1101111011	470 1110100001	520 0111101110	570 0111111110
421 1111001011	471 1110100110	521 0011110111	571 0011111111
422 1110010011	472 0111010011	522 1000001101	572 1000001001
423 1110111111	473 1010011111	523 1101110000	573 1101110010
424 1110101001	474 1100111001	524 0110111000	574 0110111001
425 1110100010	475 1111101010	525 0011011100	575 1010101010
426 0111010001	476 0111110101	526 0001101110	576 0101010101
427 1010011110	477 1010001100	527 0000110111	577 1011011100
428 0101001111	478 0101000110	528 1001101101	578 0101101110
429 1011010001	479 0010100011	529 1101000000	579 0010110111
430 1100011110	480 1000100111	530 0110100000	580 1000101101
431 0110001111	481 1101100101	531 0011010000	581 1101100000
432 1010110001	482 1111000100	532 0001101000	582 0110110000
433 1100101110	483 0111100010	533 0000110100	583 0011011000
434 0110010111	484 0011110001	534 0000011010	584 0001101100
435 1010111101	485 1000001110	535 0000001101	585 0000110110
436 1100101000	486 0100000111	536 1001110000	586 0000011011
437 0110010100	487 1011110101	537 0100111000	587 1001111011
438 0011001010	488 1100001100	538 0010011100	588 1101001011
439 0001100101	489 0110000110	539 0001001110	589 1111010011
440 1001000100	490 0011000011	540 0000100111	590 1110011111
441 0100100010	491 1000010111	541 1001100101	591 1110111001
442 0010010001	492 1101111101	542 1101000100	592 1110101010
443 1000111110	493 1111001000	543 0110100010	593 0111010101
444 0100011111	494 0111100100	544 0011010001	594 1010011100
445 1011111001	495 0011110010	545 1000011110	595 0101001110
446 1100001010	496 0001111001	546 0100001111	596 0010100111
447 0110000101	497 1001001010	547 1011110001	597 1000100101
448 1010110100	498 0100100101	548 1100001110	598 1101100100
449 0101011010	499 1011100100	549 0110000111	599 0110110010
450 0010101101	500 0101110010	550 1010110101	600 0011011001

TABLE XVII. (CONTINUED)

601 1000011010	651 1001010011	701 0110101011	751 1011111110
602 0100001101	652 1101011111	702 1010100011	752 0101111111
603 1011110000	653 1111011001	703 1100100111	753 1011001001
604 0101111000	654 1110011010	704 1111100101	754 1100010010
605 0010111100	655 0111001101	705 1110000100	755 0110001001
606 0001011110	656 1010010000	706 0111000010	756 1010110010
607 0000101111	657 0101001000	707 0011100001	757 0101011001
608 1001100001	658 0010100100	708 1000000110	758 1011011010
609 1101000110	659 0001010010	709 0100000011	759 0101101101
610 0110100011	660 0000101001	710 1011110111	760 1011000000
611 1010100111	661 1001100010	711 1100001101	761 0101100000
612 1100100101	662 0100110001	712 1111110000	762 0010110000
613 1111100100	663 1011101110	713 0111111000	763 0001011000
614 0111110010	664 0101110111	714 0011111100	764 0000101100
615 0011111001	665 1011001101	715 0001111110	765 0000010110
616 1000001010	666 1100010000	716 0000111111	766 0000001011
617 0100000101	667 0110001000	717 1001101001	767 1001110011
618 1011110100	668 0011000100	718 1101000010	768 1101001111
619 0101111010	669 0001100010	719 0110100001	769 1111010001
620 0010111101	670 0000110001	720 1010100110	770 1110011110
621 1000101000	671 1001101110	721 0101010011	771 0111001111
622 0100010100	672 0100110111	722 1011011111	772 1010010001
623 0010001010	673 1011101101	723 1100011001	773 1100111110
624 0001000101	674 1100000000	724 1111111010	774 0110011111
625 1001010100	675 0110000000	725 0111111101	775 1010111001
626 0100101010	676 0011000000	726 1010001000	776 1100101010
627 0010010101	677 0001100000	727 0101000100	777 0110010101
628 1000111100	678 0000110000	728 0010100010	778 1010111100
629 0100011110	679 0000011000	729 0001010001	779 0101011110
630 0010001111	680 0000001100	730 1001011110	780 0010101111
631 1000110001	681 0000000110	731 0100101111	781 1000100001
632 1101101110	682 0000000011	732 1011100001	782 1101100110
633 0110110111	683 1001110100	733 1100000110	783 0110110011
634 1010101101	684 0100111010	734 0110000011	784 1010101111
635 1100100000	685 0010011101	735 1010110111	785 1100100001
636 0110010000	686 1000111000	736 1100101101	786 1111100110
637 0011001000	687 0100011100	737 1111100000	787 0111110011
638 0001100100	688 0010001110	738 0111110000	788 1010001111
639 0000110010	689 0001000111	739 0011111000	789 1100110001
640 0000011001	690 1001010101	740 0001111100	790 1111101110
641 1001111010	691 1101011100	741 0000111110	791 0111110111
642 0100111101	692 0110101110	742 0000011111	792 1010001101
643 1011101000	693 0011010111	743 1001111001	793 1100110000
644 0101110100	694 1000011101	744 1101001010	794 0110011000
645 0010111010	695 1101111000	745 0110100101	795 0011001100
656 0001011101	696 0110111100	746 1010100100	796 0001100110
647 1001011000	697 0011011110	747 0101010010	797 0000110011
648 0100101100	698 0001101111	748 0010101001	798 1001101111
649 0010010110	699 1001000001	749 1000100010	799 1101000001
650 0001001011	700 1101010110	750 0100010001	800 1111010110

TABLE XVII. (CONTINUED)

801 0111101011	851 1100011111	901 1111011011	951 1011100101
802 1010000011	852 1111111001	902 1110011011	952 1100000100
803 1100110111	853 1110001010	903 1110111011	953 0110000010
804 1111101101	854 0111000101	904 1110101011	954 0011000001
805 1110000000	855 1010010100	905 1110100011	955 1000010110
806 0111000000	856 0101001010	906 1110100111	956 0100001011
807 0011100000	857 0010100101	907 1110100101	957 1011110011
808 0001110000	858 1000100100	908 1110100100	958 1100001111
809 0000111000	859 0100010010	909 0111010010	959 1111110001
810 0000011100	860 0010001001	910 0011101001	960 1110001110
811 0000001110	861 1000110010	911 1000000010	961 0111000111
812 0000000111	862 0100011001	912 0100000001	962 1010010101
813 1001110101	863 1011111010	913 1011110110	963 1100111100
814 1101001100	864 0101111101	914 0101111011	964 0110011110
815 0110100110	865 1011001000	915 1011001011	965 0011001111
816 0011010011	866 0101100100	916 1100010011	966 1000010001
817 1000011111	867 0010110010	917 1111111111	967 1101111110
818 1101111001	868 0001011001	918 1110001001	968 0110111111
819 1111001010	869 1001011010	919 1110110010	969 1010101001
820 0111100101	870 0100101101	920 0111011001	970 1100100010
821 1010000100	871 1011100000	921 1010011010	971 0110010001
822 0101000010	872 0101110000	922 0101001101	972 1010111110
823 0010100001	873 0010111000	923 1011010000	973 0101011111
824 1000100110	874 0001011100	924 0101101000	974 1011011001
825 0100010011	875 0000101110	925 0010110100	975 1100011010
826 1011111111	876 0000010111	926 0001011010	976 0110001101
827 1100001001	877 1001111101	927 0000101101	977 1010110000
828 1111110010	878 1101001000	928 1001100000	978 0101011000
829 0111111001	879 0110100100	929 0100110000	979 0010101100
830 1010001010	880 0011010010	930 0010011000	980 0001010110
831 0101000101	881 0001101001	931 0001001100	981 0000101011
832 1011010100	882 1001000010	932 0000100110	982 1001100011
833 0101101010	883 0100100001	933 0000010011	983 1101000111
834 0010110101	884 1011100110	934 1001111111	984 1111010101
835 1000101100	885 0101110011	935 1101001001	985 1110011100
836 0100010110	886 1011001111	936 1111010010	986 0111001110
837 0010001011	887 1100010001	937 0111101001	987 0011100111
838 1000110011	888 1111111110	938 1010000010	988 1000000101
839 1101101111	889 0111111111	939 0101000001	989 1101110100
840 1111000001	890 1010001001	940 1011010110	990 0110111010
841 1110010110	891 1100110010	941 0101101011	991 0011011101
842 0111001011	892 0110011001	942 1011000011	992 1000011000
843 1010010011	893 1010111010	943 1100010111	993 0100001100
844 1100111111	894 0101011101	944 1111111101	994 0010000110
845 1111101001	895 1011011000	945 1110001000	995 0001000011
846 1110000010	896 0101101100	946 0111000100	996 1001010111
847 0111000001	897 0010110110	947 0011100010	997 1101011101
848 1010010110	898 0001011011	948 0001110001	998 1111011000
849 0101001011	899 1001011011	949 1001001110	999 0111101100
850 1011010011	900 1101011011	950 0100100111	1000 0011110110

TABLE XVII. (CONCLUDED)

1001 0001111011	1007 0111011101	1013 1101011001	1019 0001010000
1002 1001001011	1008 1010011000	1014 1111011010	1020 0000101000
1003 1101010011	1009 0101001100	1015 0111101101	1021 0000010100
1004 1111011111	1010 0010100110	1016 1010000000	1022 0000001010
1005 1110011001	1011 0001010011	1017 0101000000	1023 0000000101
1006 1110111010	1012 1001011111	1018 0010100000	

TABLE XVIII. (42,30) ℓ=4 BURTON CODE

IG=1010000010001

1 101000001000	51 101111110111	101 101010100000	151 100110000101
2 010100000100	52 111111110011	102 010101010000	152 111011001010
3 001010000010	53 110111110001	103 001010101000	153 011101100101
4 000101000001	54 110011110000	104 000101010100	154 100110111010
5 101010101000	55 011001111000	105 000010101010	155 010011011101
6 010101010100	56 001100111100	106 000001010101	156 100001100110
7 001010101010	57 000110011110	107 101000100010	157 010000110011
8 000101010101	58 000011001111	108 010100010001	158 100000010001
9 101010100010	59 101001101111	109 100010000000	159 111000000000
10 010101010001	60 111100111111	110 010001000000	160 011100000000
11 100010100000	61 110110010111	111 001000100000	161 001110000000
12 010001010000	62 110011000011	112 000100010000	162 000111000000
13 001000101000	63 110001101001	113 000010001000	163 000011100000
14 000100010100	64 110000111100	114 000001000100	164 000001110000
15 000010001010	65 011000011110	115 000000100010	165 000000111000
16 000001000101	66 001100001111	116 000000010001	166 000000011100
17 101000101010	67 101110001111	117 101000000000	167 000000001110
18 010100010101	68 111111001111	118 010100000000	168 000000000111
19 100010000010	69 110111101111	119 001010000000	169 101000001100
20 010001000001	70 110011111111	120 000101000000	170 010100000110
21 100000101000	71 110001110111	121 000010100000	171 001010000011
22 010000010100	72 110000110011	122 000001010000	172 101101001001
23 001000001010	73 110000010001	123 000000101000	173 111110101100
24 000100000101	74 110000000000	124 000000010100	174 011111010110
25 101010001010	75 011000000000	125 000000001010	175 001111101011
26 010101000101	76 001100000000	126 000000000101	176 101111111101
27 100010101010	77 000110000000	127 101000001011	177 111111110110
28 010001010101	78 000011000000	128 111100001101	178 011111111011
29 100000100010	79 000001100000	129 110110001110	179 100111110101
30 010000010001	80 000000110000	130 011011000111	180 111011110010
31 100000000000	81 000000011000	131 100101101011	181 011101111001
32 010000000000	82 000000001100	132 111010111101	182 100110110100
33 001000000000	83 000000000110	133 110101010110	183 010011011010
34 000100000000	84 000000000011	134 011010101011	184 001001101101
35 000010000000	85 101000001010	135 100101011101	185 101100111110
36 000001000000	86 010100000101	136 111010100110	186 010110011111
37 000000100000	87 100010001010	137 011101010011	187 100011000111
38 000000010000	88 010001000101	138 100110100001	188 111001101011
39 000000001000	89 100000101010	139 111011011000	189 110100111101
40 000000000100	90 010000010101	140 011101101100	190 110010010110
41 000000000010	91 100000000010	141 001110110110	191 011001001011
42 000000000001	92 010000000001	142 000111011011	192 100100101101
43 101000001001	93 100000001000	143 101011100101	193 111010011110
44 111100001100	94 010000000100	144 111101111010	194 011101001111
45 011110000110	95 001000000010	145 011110111101	195 100110101111
46 001111000011	96 000100000001	146 100111010110	196 111011011111
47 101111101001	97 101010001000	147 010011101011	197 110101100111
48 111111111100	98 010101000100	148 100001111101	198 110010111011
49 011111111110	99 001010100010	149 111000110110	199 110001010101
50 001111111111	100 000101010001	150 011100011011	200 110000100010

TABLE XVIII. (continued)

201 011000010001	251 000000010110	301 110000010111	351 100100111110
202 100100000000	252 000000001011	302 110000000011	352 010010011111
203 010010000000	253 101000001110	303 110000001001	353 100001000111
204 001001000000	254 010100000111	304 110000001100	354 111000101011
205 000100100000	255 100010001011	305 011000000110	355 110100011101
206 000010010000	256 111001001101	306 001100000011	356 110010000110
207 000001001000	257 110100101110	307 101110001001	357 011001000011
208 000000100100	258 011010010111	308 111111001100	358 100100101001
209 000000010010	259 100101000011	309 011111100110	359 111010011100
210 000000001001	260 111010101001	310 001111110011	360 011101001110
211 101000001101	261 110101011100	311 101111110001	361 001110100111
212 111100001110	262 011010101110	312 111111110000	362 101111011011
213 011110000111	263 001101010111	313 011111111000	363 111111100101
214 100111001011	264 101110100011	314 001111111100	364 110111111010
215 111011101101	265 111111011001	315 000111111110	365 011011111101
216 110101111110	266 110111100100	316 000011111111	366 100101110110
217 011010111111	267 011011110010	317 101001110111	367 010010111011
218 100101010111	268 001101111001	318 111100110011	368 100001010101
219 111010100011	269 101110110100	319 110110010001	369 111000100010
220 110101011001	270 010111011010	320 110011000000	370 011100010001
221 110010100100	271 001011101101	321 011001100000	371 100110000000
222 011001010010	272 101101111110	322 001100110000	372 010011000000
223 001100101001	273 010110111111	323 000110011000	373 001001100000
224 101110011100	274 100011010111	324 000011001100	374 000100110000
225 010111001110	275 111001100011	325 000001100110	375 000010011000
226 001011100111	276 110100111001	326 000000110011	376 000001001100
227 101101111011	277 110010010100	327 101000010001	377 000000100110
228 111110110101	278 011001001010	328 111100000000	378 000000010011
229 110111010010	279 001100100101	329 011110000000	379 101000000010
230 011011101001	280 101110011010	330 001111000000	380 010100000001
231 100101111100	281 010111001101	331 000111100000	381 100010001000
232 010010111110	282 100011101110	332 000011110000	382 010001000100
233 001001011111	283 010001110111	333 000001111000	383 001000100010
234 101100100111	284 100000110011	334 000000111100	384 000100010001
235 111110011011	285 111000010001	335 000000011110	385 101010000000
236 110111000101	286 111100000000	336 000000001111	386 010101000000
237 110011101010	287 011010000000	337 101000000001	387 001010100000
238 011001110101	288 001101000000	338 111100001000	388 000101010000
239 100100110010	289 000110100000	339 011110000100	389 000010101000
240 010010011001	290 000011010000	340 001111000010	390 000001010100
241 100001000100	291 000001101000	341 000111100001	391 000000101010
242 010000100010	292 000000110100	342 101011111000	392 000000010101
243 001000010001	293 000000011010	343 010101111100	393 101000000011
244 101100000000	294 000000001101	344 001010111110	394 111100001001
245 010110000000	295 101000001111	345 000101011111	395 110110001100
246 001011000000	296 111100001111	346 101010100111	396 011011000110
247 000101100000	297 110110001111	347 111101011011	397 001101100011
248 000010110000	298 110011001111	348 110110100101	398 101110111001
249 000001011000	299 110001101111	349 110011011010	399 111111010100
250 000000101100	300 110000111111	350 011001101101	400 011111101010

TABLE XVIII. (continued)

401 001111110101	451 100100010110	501 011110110011	551 011000110011
402 101111110010	452 010010001011	502 100111010001	552 100100010001
403 010111111001	453 100001001101	503 111011100000	553 111010000000
404 100011110100	454 111000101110	504 011101110000	554 011101000000
405 010001111010	455 011100010111	505 001110111000	555 001110100000
406 001000111101	456 100110000011	506 000111011100	556 000111010000
407 101100010110	457 111011001001	507 000011101110	557 000011101000
408 010110001011	458 110101101100	508 000001110111	558 000011110100
409 100011001101	459 011010110110	509 101000110011	559 000000111010
410 111001101110	460 001101011011	510 111100010001	560 000000011101
411 011100110111	461 101110100101	511 110110000000	561 101000000111
412 100110010011	462 111111011010	512 011011000000	562 111100001011
413 111011000001	463 011111101101	513 001101100000	563 110110001101
414 110101101000	464 100111111110	514 000110110000	564 110011001110
415 011010110100	465 010011111111	515 000011011000	565 011001100111
416 001101011010	466 100001110111	516 000001101100	566 100100111011
417 000110101101	467 111000110011	517 000000110110	567 111010010101
418 101011011110	468 110100010001	518 000000011011	568 110101000010
419 010101101111	469 110010000000	519 101000000110	569 011010100001
420 100010111111	470 011001000000	520 010100000011	570 100101011000
421 111001010111	471 001100100000	521 100010001001	571 010010101100
422 110100100011	472 000110010000	522 111001001100	572 001001010110
423 110010011001	473 000011001000	523 011100100110	573 000100101011
424 110001000100	474 000001100100	524 001110010011	574 101010011101
425 011000100010	475 000000110010	525 101111000001	575 111101000110
426 001100010001	476 000000011001	526 111111101000	576 011110100011
427 101110000000	477 101000000101	527 011111110100	577 100111011001
428 010111000000	478 111100001010	528 001111111010	578 111011100100
429 001011100000	479 011110000101	529 000111111101	579 011101110010
430 000101110000	480 100111001010	530 101011110110	580 001110111001
431 000010111000	481 010011100101	531 010101111011	581 101111010100
432 000001011100	482 100001111010	532 100010110101	582 010111101010
433 000000101110	483 010000111101	533 111001010010	583 001011110101
434 000000010111	484 100000010110	534 011100101001	584 101101110010
435 101000000100	485 010000001011	535 100110011100	585 010110111001
436 010100000010	486 100000001101	536 010011001110	586 100011010100
437 001010000001	487 111000001110	537 001001100111	587 010001101010
438 101101001000	488 011100000111	538 101100111011	588 001000110101
439 010110100100	489 100110001011	539 111110010101	589 101100010010
440 001011010010	490 111011001101	540 110111000010	590 010110001001
441 000101101001	491 110101101110	541 011011100001	591 100011001100
442 101010111100	492 011010110111	542 100101111000	592 010001100110
443 010101011110	493 100101010011	543 010010111100	593 001000110011
444 001010101111	494 111010100001	544 001001011110	594 101100010001
445 101101011111	495 110101011000	545 000100101111	595 111110000000
446 111110100111	496 011010101100	546 101010011111	596 011111000000
447 110111011011	497 001101010110	547 111101000111	597 001111100000
448 110011100101	498 000110101011	548 110110101011	598 000111110000
449 110001111010	499 101011011101	549 110011011101	599 000011111000
450 011000111101	500 111101100110	550 110001100110	600 000001111100

TABLE XVIII. (continued)

601 000000111110	651 001011111010	701 011111110011	751 101100100011
602 000000011111	652 000101111101	702 100111110001	752 111110011001
603 101000011000	653 101010110110	703 111011110000	753 110111000100
604 010100001100	654 010101011011	704 011101111000	754 011011100010
605 001010000110	655 100010100101	705 001110111100	755 001101110001
606 000101000011	656 111001011010	706 000111011110	756 101110110000
607 101010101001	657 011100101101	707 000011101111	757 010111011000
608 111101011100	658 100110011110	708 101001111111	758 001011101100
609 011110101110	659 010011001111	709 111100110111	759 000101110110
610 001111010111	660 100001101111	710 110110010011	760 000010111011
611 101111100011	661 111000111111	711 110011000001	761 101001010101
612 111111111001	662 110100010111	712 110001101000	762 111100100010
613 110111110100	663 110010000011	713 011000110100	763 011110010001
614 011011111010	664 110001001001	714 001100011010	764 100111000000
615 001101111101	665 110000101100	715 000110001101	765 010011100000
616 101110110110	666 011000010110	716 101011001110	766 001001110000
617 010111011011	667 001100001011	717 010101100111	767 000100111000
618 100011100101	668 101110001101	718 100010111011	768 000010011100
619 111001111010	669 111111001110	719 111001010101	769 000001001110
620 011100111101	670 011111100111	720 110100100010	770 000000100111
621 100110010110	671 100111111011	721 011010010001	771 101000011100
622 010011001011	672 111011110101	722 100101000000	772 010100001110
623 100001101101	673 110101110010	723 010010100000	773 001010000111
624 111000111110	674 011010111001	724 001001010000	774 101101001011
625 011100011111	675 100101010100	725 000100101000	775 111110101101
626 100110000111	676 010010101010	726 000010010100	776 110111011110
627 111011001011	677 001001010101	727 000001001010	777 011011101111
628 110101101101	678 101100100010	728 000000100101	778 100101111111
629 110010111110	679 010110010001	729 101000011011	779 111010110111
630 011001011111	680 100011000000	730 111100000101	780 110101010011
631 100100100111	681 010001100000	731 110110001010	781 110010100001
632 111010011011	682 001000110000	732 011011000101	782 110001011000
633 110101000101	683 000100011000	733 100101101010	783 011000101100
634 110010101010	684 000010001100	734 010010110101	784 001100010110
635 011001010101	685 000001000110	735 100001010010	785 000110001011
636 100100100010	686 000000100011	736 010000101001	786 101011001101
637 010010010001	687 101000011010	737 100000011100	787 111101101110
638 100001000000	688 010100001101	738 010000001110	788 011110110111
639 010000100000	689 100010001110	739 001000000111	789 100111010011
640 001000010000	690 010001000111	740 101100001011	790 111011100001
641 000100001000	691 100000101011	741 111110001101	791 110101111000
642 000010000100	692 111000011101	742 110111001110	792 011010111100
643 000001000010	693 110100000110	743 011011100111	793 001101011110
644 000000100001	694 011010000011	744 100101111011	794 000110101111
645 101000011001	695 100101001001	745 111010110101	795 101011011111
646 111100000100	696 111010101100	746 110101010010	796 111101100111
647 011110000010	697 011101010110	747 011010101001	797 110110111011
648 001111000001	698 001110101011	748 100101011100	798 110011010101
649 101111101000	699 101111011101	749 010010101110	799 110001100010
650 010111110100	700 111111100110	750 001001010111	800 011000110001

TABLE XVIII. (continued)

801 100100010000	851 000101011000	901 110001101110	951 100111011111
802 010010001000	852 000010101100	902 011000110111	952 111011100111
803 001001000100	853 000001010110	903 100100010011	953 110101111011
804 000100100010	854 000000101011	904 111010000001	954 110010110101
805 000010010001	855 101000011110	905 110101001000	955 110001010010
806 101001000000	856 010100001111	906 011010100100	956 011000101001
807 010100100000	857 100010001111	907 001101010010	957 100100011100
808 001010010000	858 111001001111	908 000110101001	958 010010001110
809 000101001000	859 110100101111	909 101011011100	959 001001000111
810 000010100100	860 110010011111	910 010101101110	960 101100101011
811 000001010010	861 110001000111	911 001010110111	961 111110011101
812 000000101001	862 110000101011	912 101101010011	962 110111000110
813 101000011101	863 110000011101	913 111110100001	963 011011100011
814 111100000110	864 110000000110	914 110111011000	964 100101111001
815 011110000011	865 011000000011	915 011011101100	965 111010110100
816 100111001001	866 100100001001	916 001101110110	966 011101011010
817 111011101100	867 111010001100	917 000110111011	967 001110101101
818 011101110110	868 011101000110	918 101011010101	968 101111011110
819 001110111011	869 001110100011	919 111101100010	969 010111101111
820 101111010101	870 101111011001	920 011110110001	970 100011111111
821 111111100010	871 111111100100	921 100111010000	971 111001110111
822 011111110001	872 011111110010	922 010011101000	972 110100110011
823 100111110000	873 001111111001	923 001001110100	973 110010010001
824 010011111000	874 101111110100	924 000100111010	974 110001000000
825 001001111100	875 010111111010	925 000010011101	975 011000100000
826 000100111110	876 001011111101	926 101001000110	976 001100010000
827 000010011111	877 101101110110	927 010100100011	977 000110001000
828 101001000111	878 010110111011	928 100010011001	978 000011000100
829 111100101011	879 100011010101	929 111001000100	979 000001100010
830 110110011101	880 111001100010	930 011100100010	980 000000110001
831 110011000110	881 011100110001	931 001110010001	981 101000010010
832 011001100011	882 100110010000	932 101111000000	982 010100001001
833 100100111001	883 010011001000	933 010111100000	983 100010001100
834 111010010100	884 001001100100	934 001011110000	984 010001000110
835 011101001010	885 000100110010	935 000101111000	985 001000100011
836 001110100101	886 000010011001	936 000010111100	986 101100011001
837 101111011010	887 101001000100	937 000001011110	987 111110000100
838 010111101101	888 010100100010	938 000000101111	988 011111000010
839 100011111110	889 001010010001	939 101000010000	989 001111100001
840 010001111111	890 101101000000	940 010100001000	990 101111111000
841 100000110111	891 010110100000	941 001010000100	991 010111111100
842 111000010011	892 001011010000	942 000101000010	992 001011111110
843 110100000001	893 000101101000	943 000010100001	993 000101111111
844 110010001000	894 000010110100	944 101001011000	994 101010110111
845 011001000100	895 000001011010	945 010100101100	995 111101010011
846 001100100010	896 000000101101	946 001010010110	996 110110100001
847 000110010001	897 101000011111	947 000101001011	997 110011011000
848 101011000000	898 111100000111	948 101010101101	998 011001101100
849 010101100000	899 110110001011	949 111101011110	999 001100110110
850 001010110000	900 110011001101	950 011110101111	1000 000110011011

TABLE XVIII. (continued)

1001 101011000101	1051 111101010010	1101 011100100000	1151 100010001101
1002 111101101010	1052 011110101001	1102 001110010000	1152 111001001110
1003 011110110101	1053 100111011100	1103 000111001000	1153 011100100111
1004 100111010010	1054 010011101110	1104 000011100100	1154 100110011011
1005 010011101001	1055 001001110111	1105 000001110010	1155 111011000101
1006 100001111100	1056 101100110011	1106 000000111001	1156 110101101010
1007 010000111110	1057 111110010001	1107 101000010101	1157 011010110101
1008 001000011111	1058 110111000000	1108 111100000010	1158 100101010010
1009 101100000111	1059 011011100000	1109 011110000001	1159 010010101001
1010 111110001011	1060 001101110000	1110 100111001000	1160 100001011100
1011 110111001101	1061 000110111000	1111 010011100100	1161 010000101110
1012 110011101110	1062 000011011100	1112 001001110010	1162 001000010111
1013 011001110111	1063 000001101110	1113 000100111001	1163 101100000011
1014 100100110011	1064 000000110111	1114 101010010100	1164 111110001001
1015 111010010001	1065 101000010100	1115 010101001010	1165 110111001100
1016 110101000000	1066 010100001010	1116 001010100101	1166 011011100110
1017 011010100000	1067 001010000101	1117 101101011010	1167 001101110011
1018 001101010000	1068 101101001010	1118 010110101101	1168 101110110001
1019 000110101000	1069 010110100101	1119 100011011110	1169 111111010000
1020 000011010100	1070 100011011010	1120 010001101111	1170 011111101000
1021 000001101010	1071 010001101101	1121 100000111111	1171 001111110100
1022 000000110101	1072 100000111110	1122 111000010111	1172 000111111010
1023 101000010011	1073 010000011111	1123 110100000011	1173 000011111101
1024 111100000001	1074 100000000111	1124 110010001001	1174 101001110110
1025 110110001000	1075 111000001011	1125 110001001100	1175 010100111011
1026 011011000100	1076 110100001101	1126 011000100110	1176 100010010101
1027 001101100010	1077 110010001110	1127 001100010011	1177 111001000010
1028 000110110001	1078 011001000111	1128 101110000001	1178 011100100001
1029 101011010000	1079 100100101011	1129 111111001000	1179 100110011000
1030 010101101000	1080 111010011101	1130 011111100100	1180 010011001100
1031 001010110100	1081 110101000110	1131 001111110010	1181 001001100110
1032 000101011010	1082 011010100011	1132 000111111001	1182 000100110011
1033 000010101101	1083 100101011001	1133 101011110100	1183 101010010001
1034 101001011110	1084 111010100100	1134 010101111010	1184 111101000000
1035 010100101111	1085 011101010010	1135 001010111101	1185 011110100000
1036 100010011111	1086 001110101001	1136 101101010110	1186 001111010000
1037 111001000111	1087 101111011100	1137 010110101011	1187 000111101000
1038 110100101011	1088 010111101110	1138 100011011101	1188 000011110100
1039 110010011101	1089 001011110111	1139 111001100110	1189 000001111010
1040 110001000110	1090 101101110011	1140 011100110011	1190 000000111101
1041 011000100011	1091 111110110001	1141 100110010001	1191 101000010111
1042 100100011001	1092 110111010000	1142 111011000000	1192 111100000011
1043 111010000100	1093 011011101000	1143 011101100000	1193 110110001001
1044 011101000010	1094 001101110100	1144 001110110000	1194 110011001100
1045 001110100001	1095 000110111010	1145 000111011000	1195 011001100110
1046 101111011000	1096 000011011101	1146 000011101100	1196 001100110011
1047 010111101100	1097 101001100110	1147 000001110110	1197 101110010001
1048 001011110110	1098 010100110011	1148 000000111011	1198 111111000000
1049 000101111011	1099 100010010001	1149 101000010110	1199 011111100000
1050 101010110101	1100 111001000000	1150 010100001011	1200 001111110000

TABLE XVIII. (continued)

1201 000111111000 1202 000011111100 1203 000001111110 1204 000000111111

TABLE XIX. (28,16) ℓ=4 INTERLACED HAMMING CODE

IG 1000100000001

1	1000100000000	51	000001100000	101	101100000011	151	011001110111
2	010001000000	52	000000110000	102	110100000001	152	101110111011
3	001000100000	53	000000011000	103	111000000000	153	110101011101
4	000100010000	54	000000001100	104	011100000000	154	111000101110
5	000010001000	55	000000000110	105	001110000000	155	011100010111
6	000001000100	56	000000000011	106	000111000000	156	101100001011
7	000000100010	57	100010000010	107	000011100000	157	110100000101
8	000000010001	58	010001000001	108	000001110000	158	111000000010
9	100010001000	59	101010100000	109	000000111000	159	011100000001
10	010001000100	60	010101010000	110	000000011100	160	101100000000
11	001000100010	61	001010101000	111	000000001110	161	010110000000
12	000100010001	62	000101010100	112	000000000111	162	001011000000
13	100000001000	63	000010101010	113	100010000100	163	000101100000
14	010000000100	64	000001010101	114	010001000010	164	000010110000
15	001000000010	65	100010101010	115	001000100001	165	000001011000
16	000100000001	66	010001010101	116	100110010000	166	000000101100
17	100000000000	67	101010101010	117	010011001000	167	000000010110
18	010000000000	68	010101010101	118	001001100100	168	000000001011
19	001000000000	69	101000101010	119	000100110010	169	100010000110
20	000100000000	70	010100010101	120	000010011001	170	010001000011
21	000010000000	71	101000001010	121	100011001100	171	101010100001
22	000001000000	72	010100000101	122	010001100110	172	110111010000
23	000000100000	73	101000000010	123	001000110011	173	011011101000
24	000000010000	74	010100000001	124	100110011001	174	001101110100
25	000000001000	75	101000000000	125	110001001100	175	000110111010
26	000000000100	76	010100000000	126	011000100110	176	000011011101
27	000000000010	77	001010000000	127	001100010011	177	100011101110
28	000000000001	78	000101000000	128	100100001001	178	010001110111
29	100010000001	79	000020200000	129	110000000100	179	101010111011
30	110011000000	80	000001010000	130	011000000010	180	110111011101
31	011001100000	81	000000101000	131	001100000001	181	111001101110
32	001100110000	82	000000010100	132	100100000000	182	011100110111
33	000110011000	83	000000001010	133	010010000000	183	101100011011
34	000011001100	84	000000000101	134	001001000000	184	110100001101
35	000001100110	85	100010000011	135	000100100000	185	111000000110
36	000000110011	86	110011000001	136	000010010000	186	011100000011
37	100010011001	87	111011100000	137	000001001000	187	101100000001
38	110011001100	88	011101110000	138	000000100100	188	110100000000
39	011001100110	89	001110111000	139	000000010010	189	011010000000
40	001100110011	90	000111011100	140	000000001001	190	001101000000
41	100100011001	91	000011101110	141	100010000101	191	000110100000
42	110000001100	92	000001110111	142	110011000010	192	000011010000
43	011000000110	93	100010111011	143	011001100001	193	000001101000
44	001100000011	94	110011011101	144	101110110000	194	000000110100
45	100100000001	95	111011101110	145	010111011000	195	000000011010
46	110000000000	96	011101110111	146	001011101100	196	000000001101
47	011000000000	97	101100111011	147	000101110110	197	100010000111
48	001100000000	98	110100011101	148	000010111011	198	110011000011
49	000110000000	99	111000001110	149	100011011101	199	111011100001
50	000011000000	100	011100000111	150	110011101110	200	111111110000

TABLE XIX. (continued)

201 011111111000	251 000000100110	301 101110000000	351 010100110101
202 001111111100	252 000000010011	302 010111000000	352 101000011010
203 000111111110	253 100010001010	303 001011100000	353 010100001101
204 000011111111	254 010001000101	304 000101110000	354 101000000110
205 100011111111	255 101010100010	305 000010111000	355 010100000011
206 110011111111	256 010101010001	306 000001011100	356 101000000001
207 111011111111	257 101000101000	307 000000101110	357 110110000000
208 111111111111	258 010100010100	308 000000010111	358 011011000000
209 111101111111	259 001010001010	309 100010001100	359 001101100000
210 111100111111	260 000101000101	310 010001000110	360 000110110000
211 111100011111	261 100000100010	311 001000100011	361 000011011000
212 111100001111	262 010000010001	312 100110010001	362 000001101100
213 111100000111	263 101010001000	313 110001001000	363 000000110110
214 111100000011	264 010101000100	314 011000100100	364 000000011011
215 111100000001	265 001010100010	315 001100010010	365 100010001110
216 111100000000	266 000101010001	316 000110001001	366 010001000111
217 011110000000	267 100000101000	317 100001000100	367 101010100011
218 001111000000	268 010000010100	318 010000100010	368 110111010001
219 000111100000	269 001000001010	319 001000010001	369 111001101000
220 000011110000	270 000100000101	320 100110001000	370 011100110100
221 000001111000	271 100000000010	321 010011000100	371 001110011010
222 000000111100	272 010000000001	322 001001100010	372 000111001101
223 000000011110	273 101010000000	323 000100110001	373 100001100110
224 000000001111	274 010101000000	324 100000011000	374 010000110011
225 100010001001	275 001010100000	325 010000001100	375 101010011001
226 110011000100	276 000101010000	326 001000000110	376 110111001100
227 011001100010	277 000010101000	327 000100000011	377 011011100110
228 001100110001	278 000001010100	328 100000000001	378 001101110011
229 100100011000	279 000000101010	329 110010000000	379 100100111001
230 010010001100	280 000000010101	330 011001000000	380 110000011100
231 001001000110	281 100010001011	331 001100100000	381 011000001110
232 000100100011	282 110011000101	332 000110010000	382 001100000111
233 100000010001	283 111011100010	333 000011001000	383 100100000011
234 110010001000	284 011101110001	334 000001100100	384 110000000001
235 011001000100	285 101100111000	335 000000110010	385 111010000000
236 001100100010	286 010110011100	336 000000011001	386 011101000000
237 000110010001	287 001011001110	337 100010001101	387 001110100000
238 100001001000	288 000101100111	338 110011000110	388 000111010000
239 010000100100	289 100000110011	339 011001100011	389 000011101000
240 001000010010	290 110010011001	340 101110110001	390 000001110100
241 000100001001	291 111011001100	341 110101011000	391 000000111010
242 100000000100	292 011101100110	342 011010101100	392 000000011101
243 010000000010	293 001110110011	343 001101010110	393 100010001111
244 001000000001	294 100101011001	344 000110101011	394 110011000111
245 100110000000	295 110000101100	345 100001010101	395 111111100011
246 010011000000	296 011000010110	346 110010101010	396 111111110001
247 001001100000	297 001100001011	347 011001010101	397 111101111000
248 000100110000	298 100100000101	348 101110101010	398 011110111100
249 000010011000	299 110000000010	349 010111010101	399 001111011110
250 000001001100	300 011000000001	350 101001101010	400 000111101111

TABLE XIX. (continued)

401 100001110111	451 011001100100	501 000100101000	551 011100000101
402 110010111011	452 001100110010	502 000010010100	552 101100000010
403 111011011101	453 000110011001	503 000001001010	553 010110000001
404 111111101110	454 100001001100	504 000000100101	554 101001000000
405 011111110111	455 010000100110	505 100010010011	555 010100100000
406 101101111011	456 001000010011	506 110011001001	556 001010010000
407 110100111101	457 100110001001	507 111011100100	557 000101001000
408 111000011110	458 110001000100	508 011101110010	558 000010100100
409 011100001111	459 011000100010	509 001110111001	559 000001010010
410 101100000111	460 001100010001	510 100101011100	560 000000101001
411 110100000011	461 100100001000	511 010010101110	561 100010010101
412 111000000001	462 010010000100	512 001001010111	562 110011001010
413 111110000000	463 001001000010	513 100110101011	563 011001100101
414 011111000000	464 000100100001	514 110001010101	564 101110110010
415 001111100000	465 100000010000	515 111010101010	565 010111011001
416 000111110000	466 010000001000	516 011101010101	566 101001101100
417 000011111000	467 001000000100	517 101100101010	567 010100110110
418 000001111100	468 000100000010	518 010110010101	568 001010011011
419 000000111110	469 000010000001	519 101001001010	569 100111001101
420 000000011111	470 100011000000	520 010100100101	570 110001100110
421 100010010000	471 010001100000	521 101000010010	571 011000110011
422 010001001000	472 001000110000	522 010100001001	572 101110011001
423 001000100100	473 000100011000	523 101000000100	573 110101001100
424 000100010010	474 000010001100	524 010100000010	574 011010100110
425 000010001001	475 000001000110	525 001010000001	575 001101010011
426 100011000100	476 000000100011	526 100111000000	576 100100101001
427 010001100010	477 100010010010	527 010011100000	577 110000010100
428 001000110001	478 010001001001	528 001001110000	578 011000001010
429 100110011000	479 101010100100	529 000100111000	579 001100000101
430 010011001100	480 010101010010	530 000010011100	580 100100000010
431 001001100110	481 001010101001	531 000001001110	581 010010000001
432 000100110011	482 100111010100	532 000000100111	582 101011000000
433 100000011001	483 010011101010	533 100010010100	583 010101100000
434 110010001100	484 001001110101	534 010001001010	584 001010110000
435 011001000110	485 100110111010	535 001000100101	585 000101011000
436 001100100011	486 010011011101	536 100110010010	586 000010101100
437 100100010001	487 101011101110	537 010011001001	587 000001010110
438 110000001000	488 010101110111	538 101011100100	588 000000101011
439 011000000100	489 101000111011	539 010101110010	589 100010010110
440 001100000010	490 110110011101	540 001010111001	590 010001001011
441 000110000001	491 111001001110	541 100111011100	591 101010100101
442 100001000000	492 011100100111	542 010011101110	592 110111010010
443 010000100000	493 101100010011	543 001001110111	593 011011101001
444 001000010000	494 110100001001	544 100110111011	594 101111110100
445 000100001000	495 111000000100	545 110001011101	595 010111111010
446 000010000100	496 011100000010	546 111010101110	596 001011111101
447 000001000010	497 001110000001	547 011101010111	597 100111111110
448 000000100001	498 100101000000	548 101100101011	598 010011111111
449 100010010001	499 010010100000	549 110100010101	599 101011111111
450 110011001000	500 001001010000	550 111000001010	600 110111111111

TABLE XIX. (continued)

601 111001111111	651 011001000010	701 100010011011	751 011100100000
602 111110111111	652 001100100001	702 110011001101	752 001110010000
603 111101011111	653 100100010000	703 111011100110	753 000111001000
604 111100101111	654 010010001000	704 011101110011	754 000011100100
605 111100010111	655 001001000100	705 101100111001	755 000001110010
606 111100001011	656 000100100010	706 110100011100	756 000000111001
607 111100000101	657 000010010001	707 011010001110	757 100010011101
608 111100000010	658 100011001000	708 001101000111	758 110011001110
609 011110000001	659 010001100100	709 100100100011	759 011001100111
610 101101000000	660 001000110010	710 110000010001	760 101110110011
611 010110100000	661 000100011001	711 111010001000	761 110101011001
612 001011010000	662 100000001100	712 011101000100	762 111000101100
613 000101101000	663 010000000110	713 001110100010	763 011100010110
614 000010110100	664 001000000011	714 000111010001	764 001110001011
615 000001011010	665 100110000001	715 100001101000	765 100101000101
616 000000101101	666 110001000000	716 010000110100	766 110000100010
617 100010010111	667 011000100000	717 001000011010	767 011000010001
618 110011001011	668 001100010000	718 000100001101	768 101110001000
619 111011100101	669 000110001000	719 100000000110	769 010111000100
620 111111110010	670 000011000100	720 010000000011	770 001011100010
621 011111111001	671 000001100010	721 101010000001	771 000101110001
622 101101111100	672 000000110001	722 110111000000	772 100000111000
623 010110111110	673 100010011010	723 011011100000	773 010000011100
624 001011011111	674 010001001101	724 001101110000	774 001000001110
625 100111101111	675 101010100110	725 000110111000	775 000100000111
626 110001110111	676 010101010011	726 000011011100	776 100000000011
627 111010111011	677 101000101001	727 000001101110	777 110010000001
628 111111011101	678 110110010100	728 000000110111	778 111011000000
629 111101101110	679 011011001010	729 100010011100	779 011101100000
630 011110110111	680 001101100101	730 010001001110	780 001110110000
631 101101011011	681 100100110010	731 001000100111	781 000111011000
632 110100101101	682 010010011001	732 100110010011	782 000011101100
633 111000010110	683 101011001100	733 110001001001	783 000001110110
634 011100001011	684 010101100110	734 111010100100	784 000000111011
635 101100000101	685 001010110011	735 011101010010	785 100010011110
636 110100000010	686 100111011001	736 001110101001	786 010001001111
637 011010000001	687 110001101100	737 100101010100	787 101010100111
638 101111000000	688 011000110110	738 010010101010	788 110111010011
639 010111100000	689 001100011011	739 001001010101	789 111001101001
640 001011110000	690 100100001101	740 100110101010	790 111110110100
641 000101111000	691 110000000110	741 010011010101	791 011111011010
642 000010111100	692 011000000011	742 101011101010	792 001111101101
643 000001011110	693 101110000001	743 010101110101	793 100101110110
644 000000101111	694 110101000000	744 101000111010	794 010010111011
645 100010011000	695 011010100000	745 010100011101	795 101011011101
646 010001001100	696 001101010000	746 101000001110	796 110111101110
647 001000100110	697 000110101000	747 010100000111	797 011011110111
648 000100010011	698 000011010100	748 101000000011	798 101111111011
649 100000001001	699 000001101010	749 110110000001	799 110101111101
650 110010000100	700 000000110101	750 111001000000	800 111000111110

TABLE XIX. (continued)

801	011100011111
802	101100001111
803	110100000111
804	111000000011
805	111110000001
806	111101000000
807	011110100000
808	001111010000
809	000111101000
810	000011110100
811	000001111010
812	000000111101
813	100010011111
814	110011001111
815	111011100111
816	111111110011
817	111101111001
818	111100111100
819	011110011110
820	001111001111
821	100101100111
822	110000110011
823	111010011001
824	111111001100
825	011111100110
826	001111110011
827	100101111001
828	110000111100
829	011000011110
830	001100001111
831	100100000111
832	110000000011
833	111010000001
834	111111000000
835	011111100000
836	001111110000
837	000111111000
838	000011111100
839	000001111110
840	000000111111

Tables XVI and XVII illustrate two computer-generated codes with 10 parity bits. It can be seen that the ℓ equals 4 code uses more ($85 \cdot 8 = 680$) of the possible correctable-error patterns (1023) than the ℓ equals 5 code ($27 \cdot 16 = 432$). If the length of the code word is 27 or less, the ℓ equals 5 should be used since it has better burst error-correcting capability and at most 432 correctable-error patterns is needed. Tables XVIII and XIX were included to illustrate the Burton and interlaced Hamming codes. It can be observed that the Burton and interlaced Hamming codes for these short word lengths use very few, $42 \cdot 8 = 336$ and $28 \cdot 8 = 224$, respectively, of the possible correctable-error patterns, 4095.

An ℓ burst error-correcting cyclic code can be most easily decoded by the error-trapping decoder for single errors with a slight modification. If it is assumed that the errors are confined to the ℓ high order bits

$$e(X) = X^{n-\ell} + \ldots + X^{n-2} + X^{n-1} \tag{74}$$

Observing that

$$X^{n-k}e(X) = (X^{n-k-\ell} + \ldots + X^{n-k-2} + X^{n-k-1})(X^n + 1) + X^{n-k-\ell} + \ldots + X^{n-k-2} + X^{n-k-1} \tag{75}$$

and dividing by $g(X)$ yields

$$X^{n-k}e(X) = p(X)g(X) + X^{n-k-\ell} + \ldots + X^{n-k-2} + X^{n-k-1} \tag{76}$$

and the remainder is

$$S(X) = X^{n-k-\ell} + \ldots + X^{n-k-2} + X^{n-k-1} \tag{77}$$

This indicates that the high order syndrome bits correspond to the error bits and the remaining syndrome bits are zero. When zeros are detected in the first $n-k-\ell$ syndrome bits, the ℓ high order syndrome bits is the correctable burst of length ℓ. These zeros can be detected using a NOR gate. To associate the errors with the high order syndrome bits, the input is multiplied by X^{n-k} which is equivalent to connecting the input to the high end of the syndrome register. The general form of the decoder is given in Figure 24.

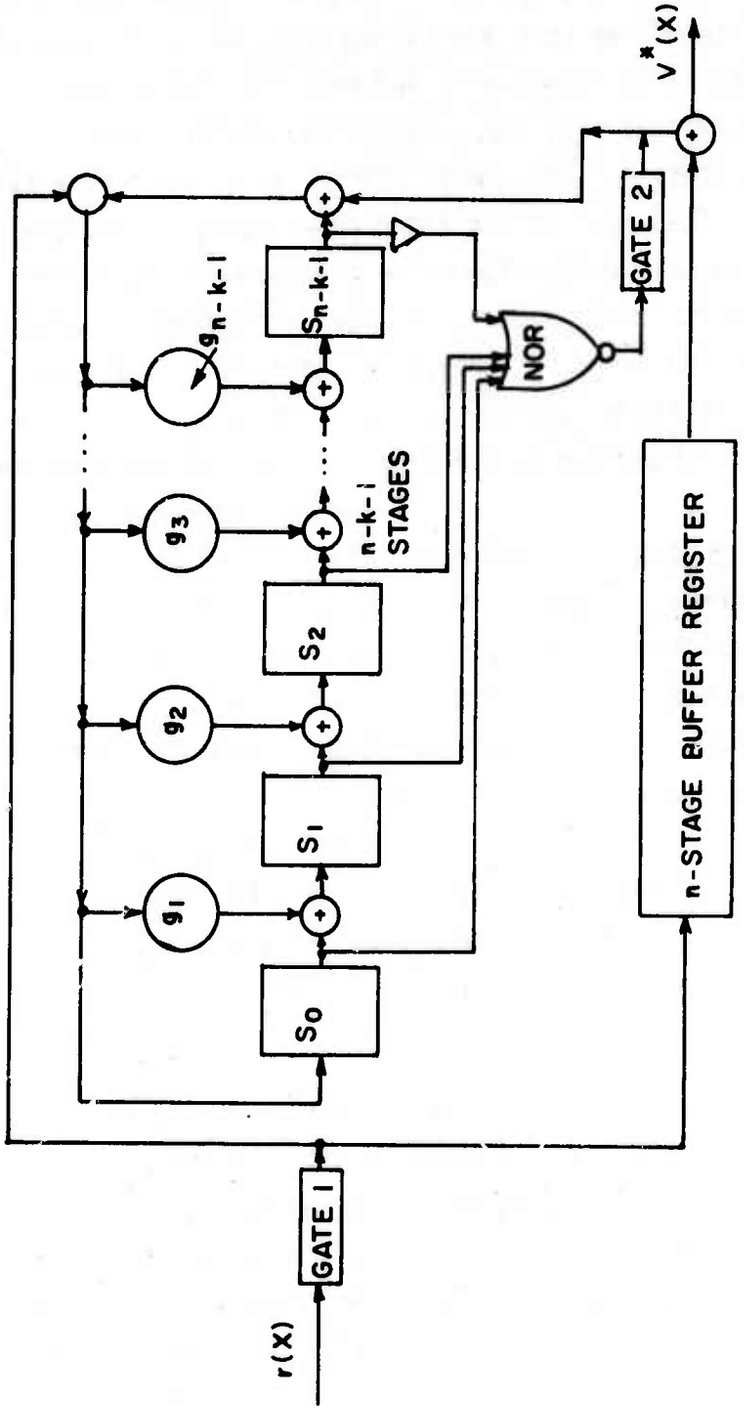


Figure 24. Burst Error-Trapping Decoder

The decoding procedure for this burst error-trapping decoder can be described in the following steps:

1. Gate 1 is turned on, and Gate 2 is turned off. The syndrome S(X) is formed by shifting all n bits of the received signal r(X) into the syndrome register. At the same time the n bits of the word are stored into the buffer register.
2. Gate 1 is turned off, and Gate 2 is turned on. With the input cut off, the word read in during Step 1 is now processed. The n bits stored in the buffer register are shifted out and are ready to be corrected as the syndrome register is shifted while searching for a correctable burst error pattern. As soon as the n-k-ℓ leftmost stages of the syndrome register contain only zeros, the ℓ rightmost stages contain the burst pattern. The corrections are then made when a one is present in the rightmost stage (output of NOR goes to 1 and adds modulo 2 to the present output of the buffer register). When an error is corrected, the feedback from the high end of the syndrome register is also eliminated as there should be no feedback once the burst is found.
3. When Step 2 is completed, the decoder is ready for the next word and Step 1 is repeated.

For the 48-bit block length with 8 parity bits available, the computer-generated (63,55) ℓ equal to 3 code was chosen as the best burst error-correcting code. The generator polynomial is given by

$$g(X) = 1 + X + X^2 + X^5 + X^8 \tag{78}$$

and the encoder for the (63,55) code and the shortened version the (48,40) code which fits the six 8-bit subword format is given in Figure 25. If the code used was a cyclic code and not a shortened cyclic code, the input to the decoder would be to the high end of the syndrome register. The input connections to the decoder for a shortened cyclic code is given as the remainder from dividing X^{n-k+n} by g(X). This can be obtained for the

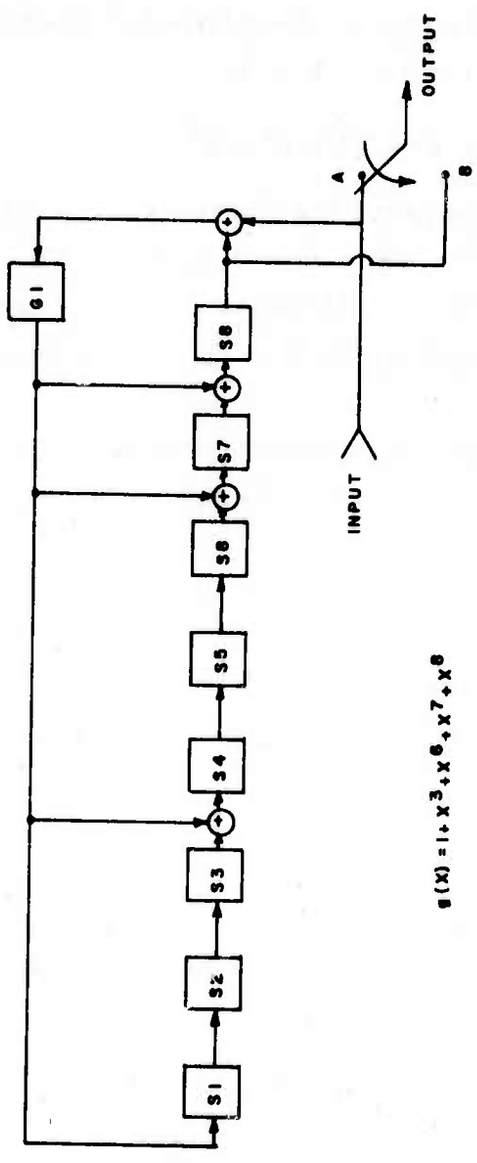


Figure 25. Encoder for (48,40) Shortened Cyclic Burst 3 Error-Correcting Code

(48,40) code shortened from the (63,55) code from Table XIV. The first entry in the table corresponds to no shortening or η equals 0, and each succeeding entry corresponds to another bit shortened. For our case at hand, η equals 15 and the input connection polynomial is obtained from entry 16 in Table XIV as 0 0 1 0 0 1 1 1 or as

$$C(X) = X^2 + X^5 + X^6 + X^7 \tag{79}$$

With this input connection polynomial the decoder for the (48,40) shortened cyclic burst 3 error-correcting code is given in Figure 26. For the 24-bit block length with 8 parity bits, the largest ℓ is 3, the same as the 48-bit block length. This (24,16) code can be obtained by shortening the (63,55) code 29 bits.

For the (48,38) code format the computer-generated (85,75) ℓ equal to 4 code can be shortened by 37 bits. The generator polynomial is given from Table X as

$$g(X) = 1 + X^2 + X^3 + X^5 + X^7 + X^{10} \tag{80}$$

and the input connection polynomial is obtained from Table XVI, with η equal to 138 (37 + 101 since this code is already shortened from length 186)

$$C(X) = X + X^3 + X^5 + X^6 + X^8 \tag{81}$$

The format that was mechanized, the (24,14) ℓ equal to 5 code is derived from the (27,17) code of Table X. The generator polynomial is given as

$$g(X) = 1 + X^3 + X^4 + X^5 + X^7 + X^8 + X^{10} \tag{82}$$

and the encoder is shown in Figure 27. The shortening factor for this (24,14) code is 317 (3 + 314 since this code is already shortened from length 341). The connection polynomial, obtained as the remainder of dividing X^{10+317} by g(X), is given in Table XVII as

$$C(X) = 1 + X + X^2 + X^5 + X^7 + X^9 \tag{83}$$

The decoder is given in Figure 28.

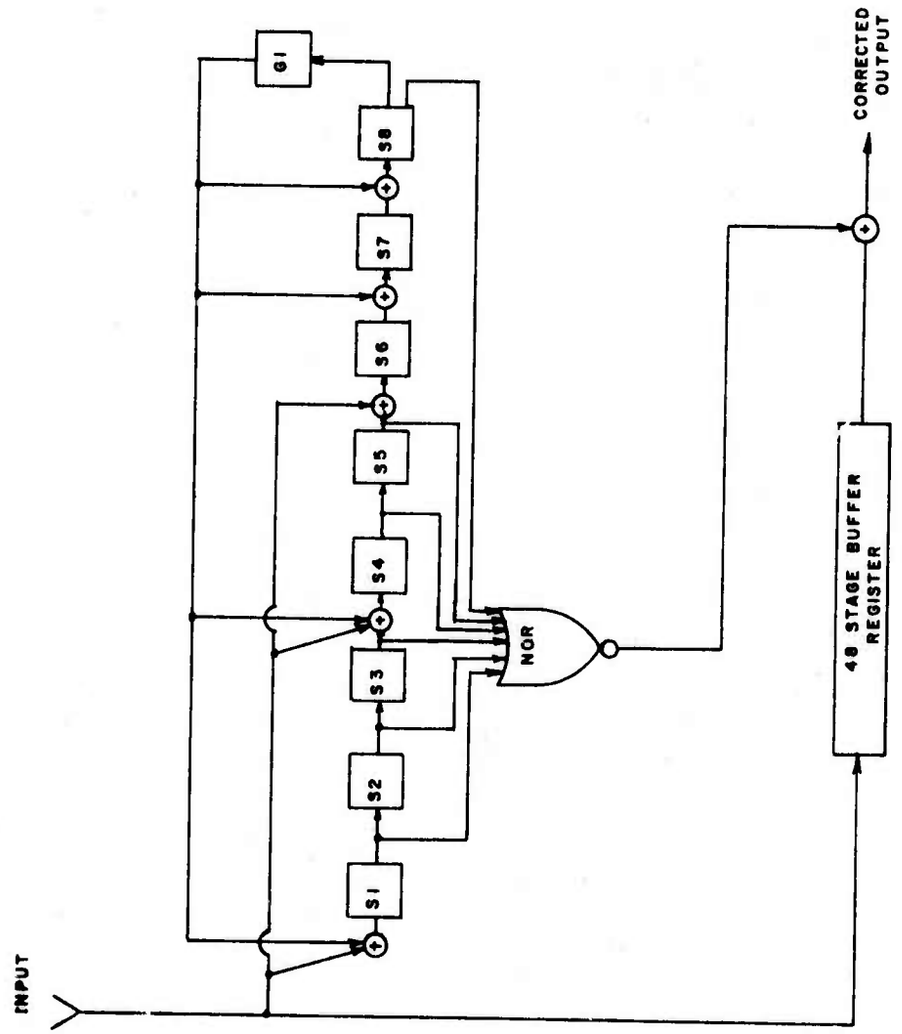


Figure 26. Decoder for (48,40) Shortened Cyclic Burst 3 Error-Correcting Code

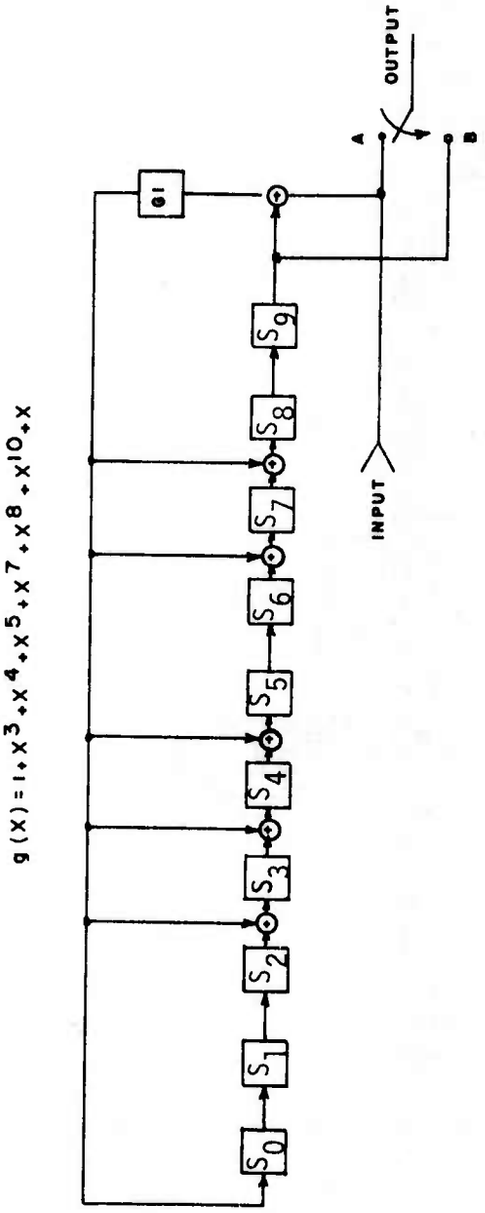


Figure 27. Encoder for (24,14) Burst 5 Error-Correcting Code

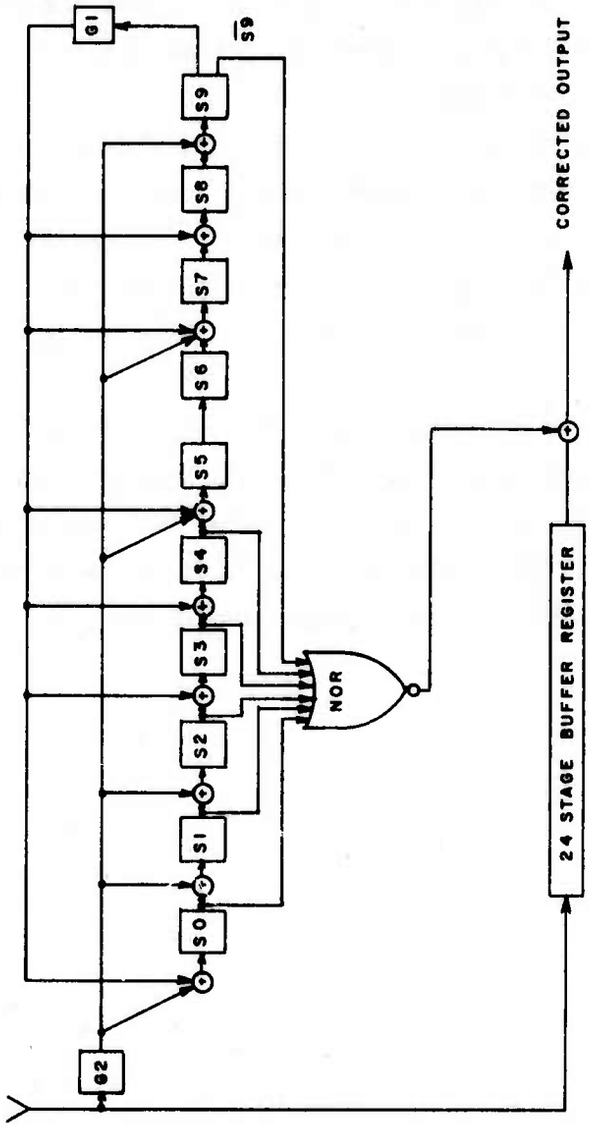


Figure 28. Decoder for (24,14) Burst 5 Error-Correcting Code

Tables XX through XXV illustrate how this (24,14) burst decoder processes six different received signals and corrects the burst errors of length 5 or less. Tables XX and XXI illustrate the correction of burst length 5 errors of 1 1 1 1 1 and 1 1 0 1 1 , respectively. The bursts are trapped when the five least significant bits of the syndrome register are 0. Tables XXII and XXIII illustrate the correction of bursts 1 1 1 1 and 1 0 1 1, respectively, of length 4. Table XXIV shows the correction of a burst of length 1, and Table XXV shows the processing of a received word that does not contain any errors.

Processing the received words with a single decoder of the form shown in Figure 28 requires a two-word time (48-bit time) to decode a single word (24 bits), as is illustrated in Tables XX through XXV. This decoder then cannot operate in real time. The received words can be decoded in real time though, if two of the decoders are used simultaneously as is shown in Figure 29.

The switches change position after each received word (24 bits). The received word is fed into one of the decoders, and the syndrome calculated as the previous word is fed out of the decoder and simultaneously corrected. This configuration will allow the decoding to be accomplished with only an initial delay of one word for the calculation of the first syndrome.

TABLE XX. PROCESSING OF $r(X)=1+X^2+X^3+X^4+X^6+X^9+X^{10}+X^{11}+X^{12}+X^{13}+X^{14}+X^{15}+X^{16}+X^{17}+X^{18}$ WITH (24,14) BURST DECODER

r(X)		s(X)	V*(X)		
X^{23}	0	0000000000			
	0	0000000000			
	0	0000000000			
X^{20}	0	0000000000			
	0	0000000000			
	1	1110010101			
	1	0000101001			
	1	0111110111			
X^{15}	1	0100011000			
	1	1100011001			
	1	0001101111			
	1	0111010100			
	1	1101111111			
X^{10}	1	0001011100			
	1	1110111011			
	0	1110101011			
	0	1110100011			
	1	0000110010			
X^5	0	0000011001			
	1	0111101111			
	1	0100010100			
	1	1100011111			
	0	1111111001			
X^0	1	0000011111			BURST TRAPPED
		0000001111	1	X^{23}	Bit 23 Corrected
		0000000111	1		Bit 22 Corrected
		0000000011	1		Bit 21 Corrected
		0000000001	1	X^{20}	Bit 20 Corrected
		0000000000	1		Bit 19 Corrected
		0000000000	1		
		0000000000	1		
		0000000000	1		
		0000000000	1	X^{15}	
		0000000000	1		
		0000000000	1		
		0000000000	1		
		0000000000	1		
		0000000000	1	X^{10}	
		0000000000	1		
		0000000000	0		
		0000000000	0		
		0000000000	1		
		0000000000	0	X^5	
		0000000000	1		
		0000000000	1		
		0000000000	1		
		0000000000	0		
		0000000000	1	X^0	

TABLE XXI. PROCESSING OF $r(X)=X^7+X^{23}$

WITH (24,14) BURST DECODER

r(X)		s(X)	V*(X)		
X^{23}	1	1110010101			
	0	1110111100			
	0	0111011110			
X^{20}	0	0011101111			
	0	1000000001			
	0	1101110110			
	0	0110111011			
	0	1010101011			
X^{15}	0	1100100011			
	0	1111100111			
	0	1110000101			
	0	1110110100			
	0	0111011010			
X^{10}	0	0011101101			
	0	1000000000			
	0	0100000000			
	1	1100010101			
	0	1111111100			
X^5	0	0111111110			
	0	0011111111			
	0	1000001001			
	0	1101110010			
	0	0110111001			
X^0	0	1010101010			
		0101010101	1	X^{23}	
		1011011100	0		
		0101101110	0		
		0010110111	0	X^{20}	
		1000101101	0		
		1101100000	0		
		0110110000	0		
		0011011000	0		
		0001101100	0	X^{15}	
		0000110110	0		
		0000011011	0		BURST TRAPPED
		0000001101	1		Bit 12 Corrected
		0000000110	1		Bit 11 Corrected
		0000000011	0	X^{10}	
		0000000001	1		Bit 9 Corrected
		0000000000	1		Bit 8 Corrected
		0000000000	1		
		0000000000	0		
		0000000000	0	X^5	
		0000000000	0		
		0000000000	0		
		0000000000	0		
		0000000000	0		
		0000000000	0	X^0	

TABLE XXII. PROCESSING OF $r(X)=1+X+X^2+X^3+X^4+X^5+X^7+X^8+X^9+X^{10}+X^{11}+X^{12}+X^{13}+X^{14}$ WITH (24,14) BURST DECODER

r(X)		s(X)	V*(X)		
X^{23}	0	0000000000			
	0	0000000000			
	0	0000000000			
X^{20}	0	0000000000			
	0	0000000000			
	0	0000000000			
	0	0000000000			
X^{15}	0	0000000000			
	1	1110010101			
	1	0000101001			
	1	0111110111			
	1	0100011000			
X^{10}	1	1100011001			
	1	0001101111			
	1	0111010100			
	1	1101111111			
	0	1111001001			
X^5	1	0000000111			
	1	0111100000			
	1	1101100101			
	1	0001010001			
	1	0111001011			
X^0	1	0100000110			
		0010000011	0	X^{23}	
		1000110111	0		
		1101101101	0		
		1111000000	0	X^{20}	
		0111100000	0		
		0011110000	0		
		0001111000	0		
		0000111100	0		
		0000011110	0	X^{15}	BURST TRAPPED
		0000001111	1		
		0000000111	0		Bit 13 Corrected
		0000000011	0		Bit 12 Corrected
		0000000001	0		Bit 11 Corrected
		0000000000	0	X^{10}	Bit 10 Corrected
		0000000000	1		
		0000000000	1		
		0000000000	1		
		0000000000	0		
		0000000000	1	X^5	
		0000000000	1		
		0000000000	1		
		0000000000	1		
		0000000000	1		
		0000000000	1	X^0	

TABLE XXIII. PROCESSING OF $r(X)=1+X^2+X^3+X^4+X^5+X^6+X^7+X^8+X^9+X^{10}+X^{11}+X^{12}+X^{13}+X^{14}+X^{15}+X^{16}+X^{17}+X^{18}+X^{19}+X^{20}+X^{21}+X^{22}+X^{23}$ WITH (24,14) BURST DECODER

r(X)		s(X)	V*(X)		
X^{23}	1	1110010101			
	1	0000101001			
	1	0111110111			
X^{20}	1	0100011000			
	1	1100011001			
	1	0001101111			
	1	0111010100			
	1	1101111111			
X^{15}	1	0001011100			
	1	1110111011			
	1	0000111110			
	1	1110001010			
	1	1001010000			
X^{10}	1	1010111101			
	1	0010111101			
	1	0110111101			
	1	0100111101			
	1	0101111101			
X^5	1	0101011101			
	1	0101001101			
	1	0101000101			
	1	0101000001			
	0	1011010110			
X^0	1	1011111110			
		0101111111	1	X^{23}	
		1011001001	1		
		1100010010	1		
		0110001001	1	X^{20}	
		1010110010	1		
		0101011001	1		
		1011011010	1		
		0101101101	1		
		1011000000	1	X^{15}	
		0101100000	1		
		0010110000	1		
		0001011000	1		
		0000101100	1		
		0000010110	1	X^{10}	BURST TRAPPED
		0000001011	1		
		0000000101	1		Bit 8 Corrected
		0000000010	0		Bit 7 Corrected
		0000000001	1		
		0000000000	0	X^5	Bit 5 Corrected
		0000000000	1		
		0000000000	1		
		0000000000	1		
		0000000000	0		
		0000000000	1	X^0	

TABLE XXIV. PROCESSING OF $r(X)=X^3+X^4+X^5+X^7+X^8+X^{10}$ WITH (24,14) BURST DECODER

r(X)		s(X)		V*(X)	
X^{23}	0	0000000000			
	0	0000000000			
	0	0000000000			
X^{20}	0	0000000000			
	0	0000000000			
	0	0000000000			
	0	0000000000			
	0	0000000000			
X^{15}	0	0000000000			
	0	0000000000			
	0	0000000000			
	0	0000000000			
	0	0000000000			
X^{10}	0	0000000000			
	1	1110010101			
	0	1110111100			
	1	1001001011			
	1	0011000110			
	0	0001100011			
X^5	1	0111010010			
	1	1101111100			
	1	1000101011			
	0	1101100011			
	0	1111000111			
X^0	0	1110010101			
		1110111100	0	X^{23}	
		0111011110	0		
		0011101111	0		
		1000000001	0	X^{20}	
		1101110110	0		
		0110111011	0		
		1010101011	0		
		1100100011	0		
		1111100111	0	X^{15}	
		1110000101	0		
		1110110100	0		
		0111011010	0		
		0011101101	0		
		1000000000	1	X^{10}	
		0100000000	0		
		0010000000	1		
		0001000000	1		
		0000100000	0		
		0000010000	1	X^5	BURST TRAPPED
		0000001000	1		
		0000000100	1		
		0000000010	0		
		0000000001	0		
		0000000000	1	X^0	Bit 0 Corrected

TABLE XXV. PROCESSING OF $r(X)=1+X^3+X^4+X^5+X^7+X^8+X^{10}$ WITH (24,14) BURST DECODER

r(X)		s(X)	V*(X)	
X^{23}	0	0000000000		
	0	0000000000		
	0	0000000000		
X^{20}	0	0000000000		
	0	0000000000		
	0	0000000000		
	0	0000000000		
	0	0000000000		
X^{15}	0	0000000000		
	0	0000000000		
	0	0000000000		
	0	0000000000		
	0	0000000000		
X^{10}	1	1110010101		
	0	1110111100		
	1	1001001011		
	1	0011000110		
	0	0001100011		
X^5	1	0111010010		
	1	1101111100		
	1	1000101011		
	0	1101100011		
	0	1111000111		
X^0	1	0000000000		BURST TRAPPED
		0000000000	0	X^{23}
		0000000000	0	
		0000000000	0	
		0000000000	0	X^{20}
		0000000000	0	
		0000000000	0	
		0000000000	0	
		0000000000	0	
		0000000000	0	X^{15}
		0000000000	0	
		0000000000	0	
		0000000000	0	
		0000000000	0	
		0000000000	1	X^{10}
		0000000000	0	
		0000000000	1	
		0000000000	1	
		0000000000	0	
		0000000000	1	X^5
		0000000000	1	
		0000000000	1	
		0000000000	0	
		0000000000	0	
		0000000000	1	X^0

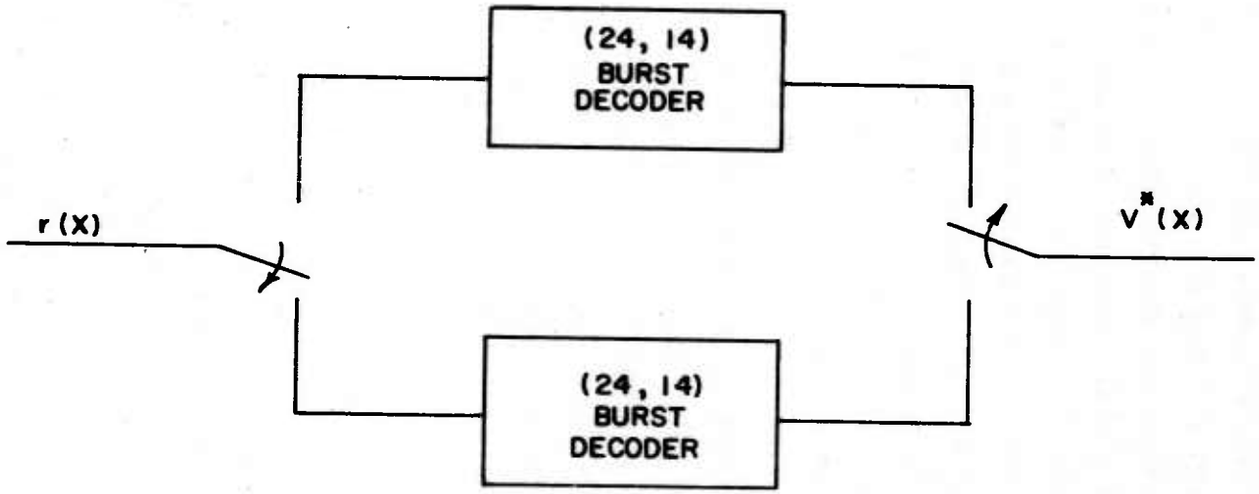


Figure 29. Real Time Decoding with a One-Word Delay

SECTION VI

(24,14) BURST CODE CONSTRUCTION

The construction of the encoder and decoder for the (24,14) ℓ equal to 5 cyclic burst error-correcting code is considered in this section. TTL (transistor-transistor-logic) integrated logic circuits were chosen for the system on the basis of proven reliability, ease of design, and availability of a large number of circuit functions from several manufacturers.

For the encoder shown in Figure 27 a 10-bit shift register for the parity check register, six 2-input EXCLUSIVE OR gates, a gate, G1, that is to be on during the 14 clock pulses that the information bits are read in and off after the $14^{\underline{th}}$ clock pulse, and a switch that is in position A for the first 14 clock pulses and in position B after the $14^{\underline{th}}$ clock pulse was needed. D flip-flops were chosen for the parity check register and, in particular, the N7474, which is a dual edge-triggered flip-flop, was chosen, and the N7486 quad 2-input integrated circuit was used for the EXCLUSIVE OR gates. The switch and the control for G1 consisted of a 4-bit counter, N7493, which counts the first 14 clock pulses, a monostable multivibrator which is driven by the output of the count 14, and a JK flip-flop, N7470 (edge-triggered and gated inputs), which is clocked by the monostable to set its output, Q, to 1 and $\overline{Q}$ to 0. The output of the JK flip-flop controls the switch and G1. The AND gates used in the switch and G1 circuitry are N7411s, which are triple 3-input AND gates. The encoder was mechanized on 2 printed circuit boards, and the circuit diagrams for encoder boards 1 and 2 are shown in Figures 30 and 31, respectively. The printed circuit boards constructed from these circuit diagrams are shown in Figures 32 and 33. These are two-sided boards to minimize external wiring. O is the output of the encoder, V^+ is +5v, $\Delta 1$ is the message input, $\Delta 2$ is the feedback from G1, C is the clock input, Ξ is ground, Γ is the clear, and E7 is the output of parity register 1, which is on board 1 and has to be connected to board 2. G1 is not shown on the printed circuit board as it was added later.

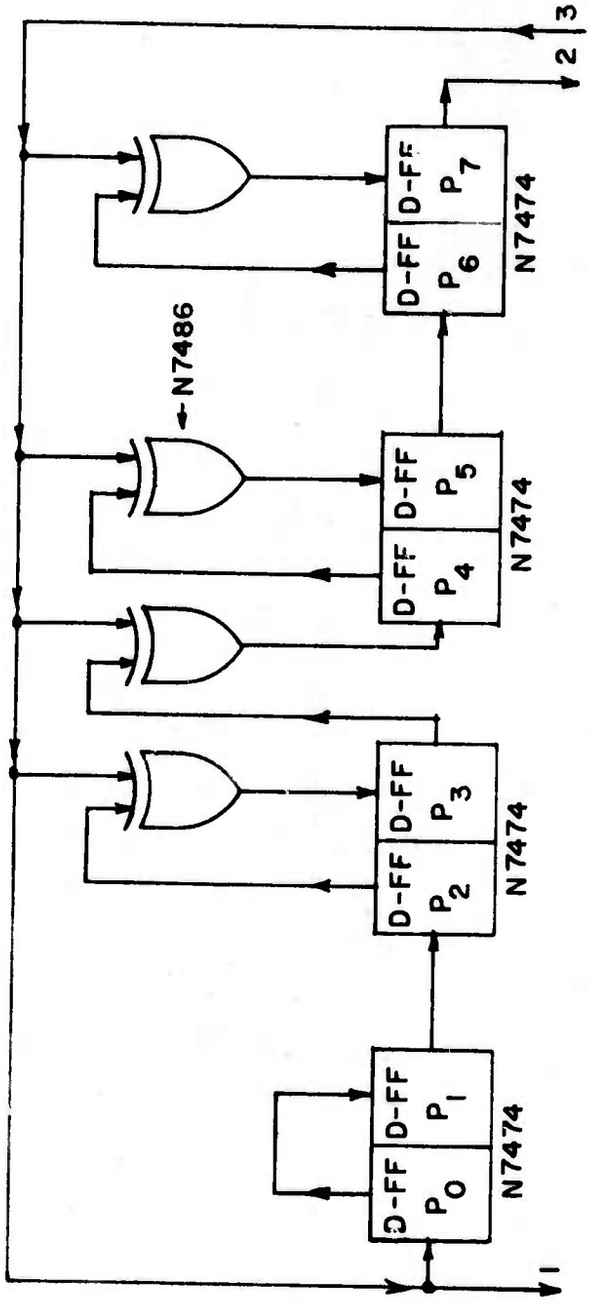


Figure 30. Encoder Circuit Diagram 1

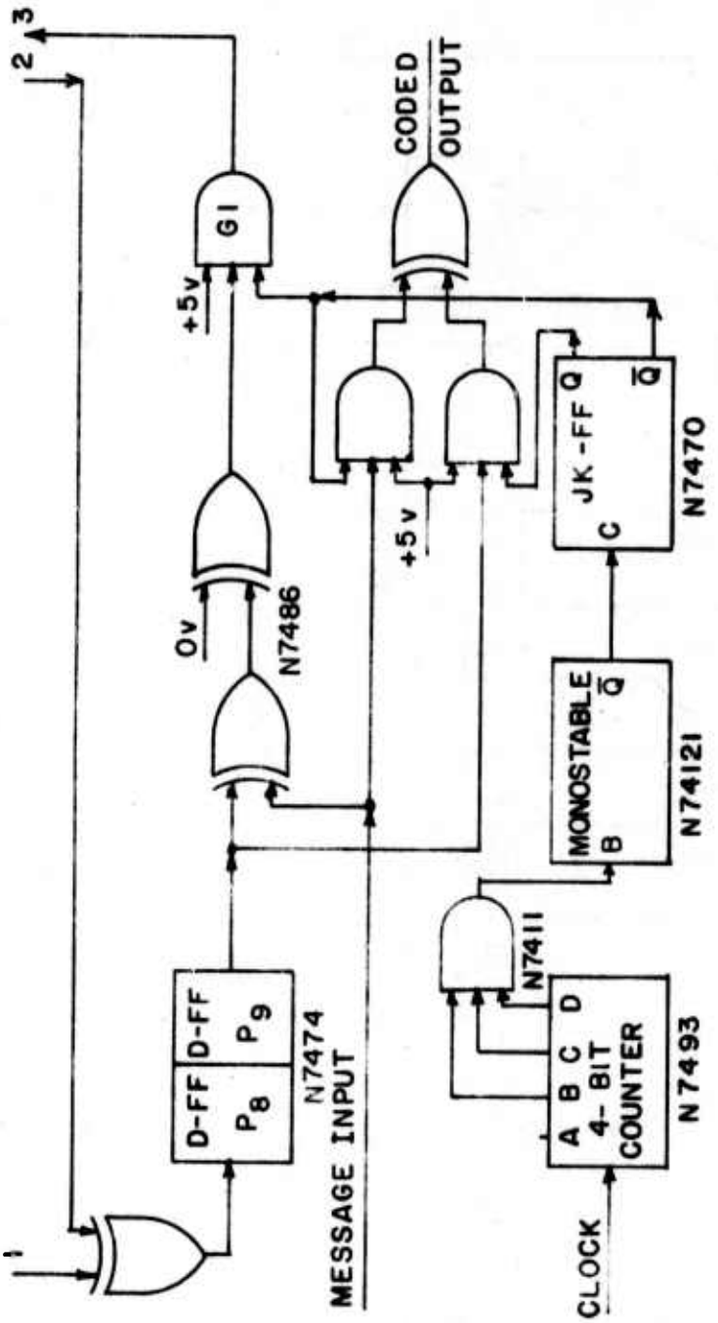


Figure 31. Encoder Circuit Diagram 2

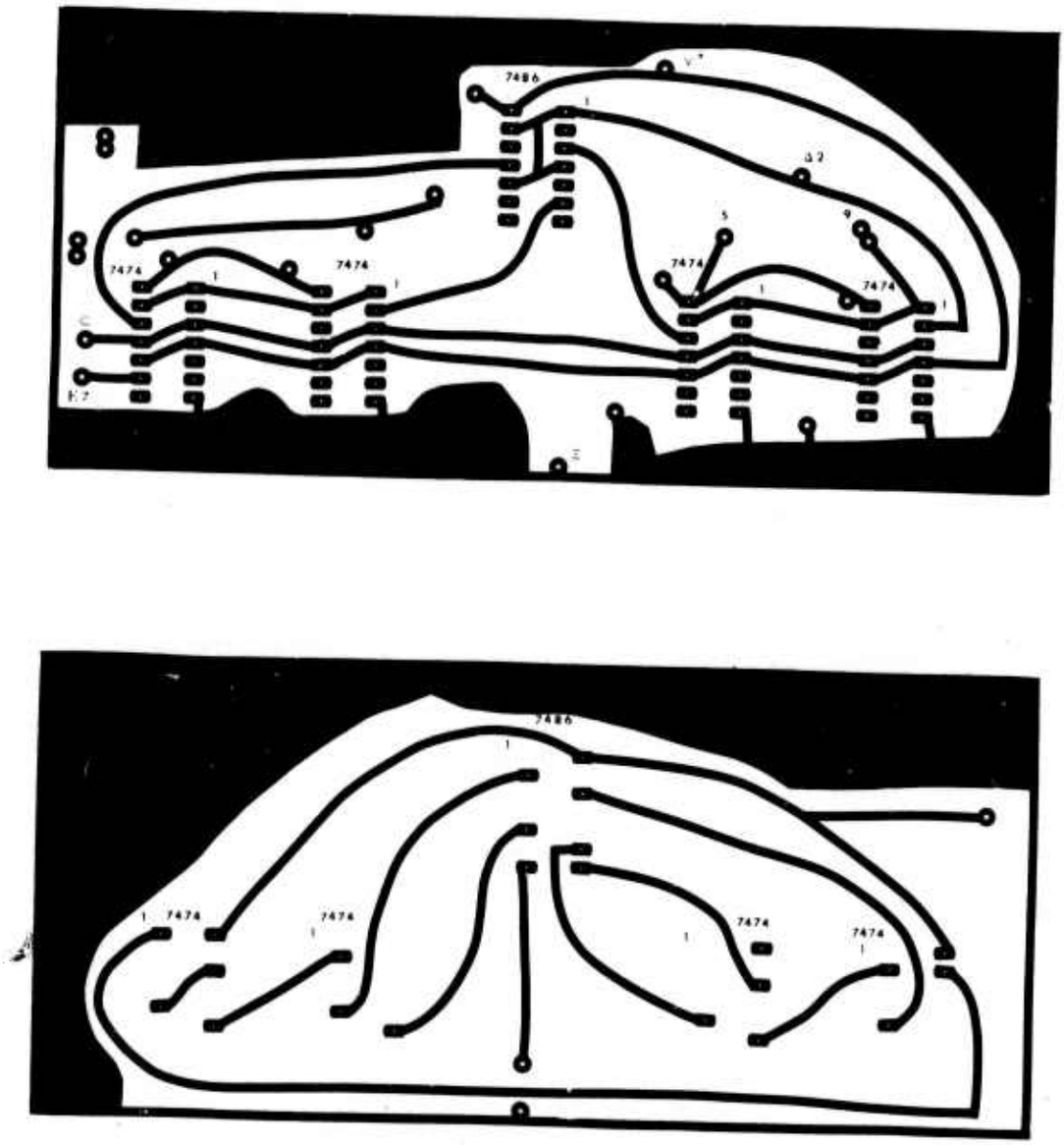


Figure 32. Encoder Printed Circuit Board 1

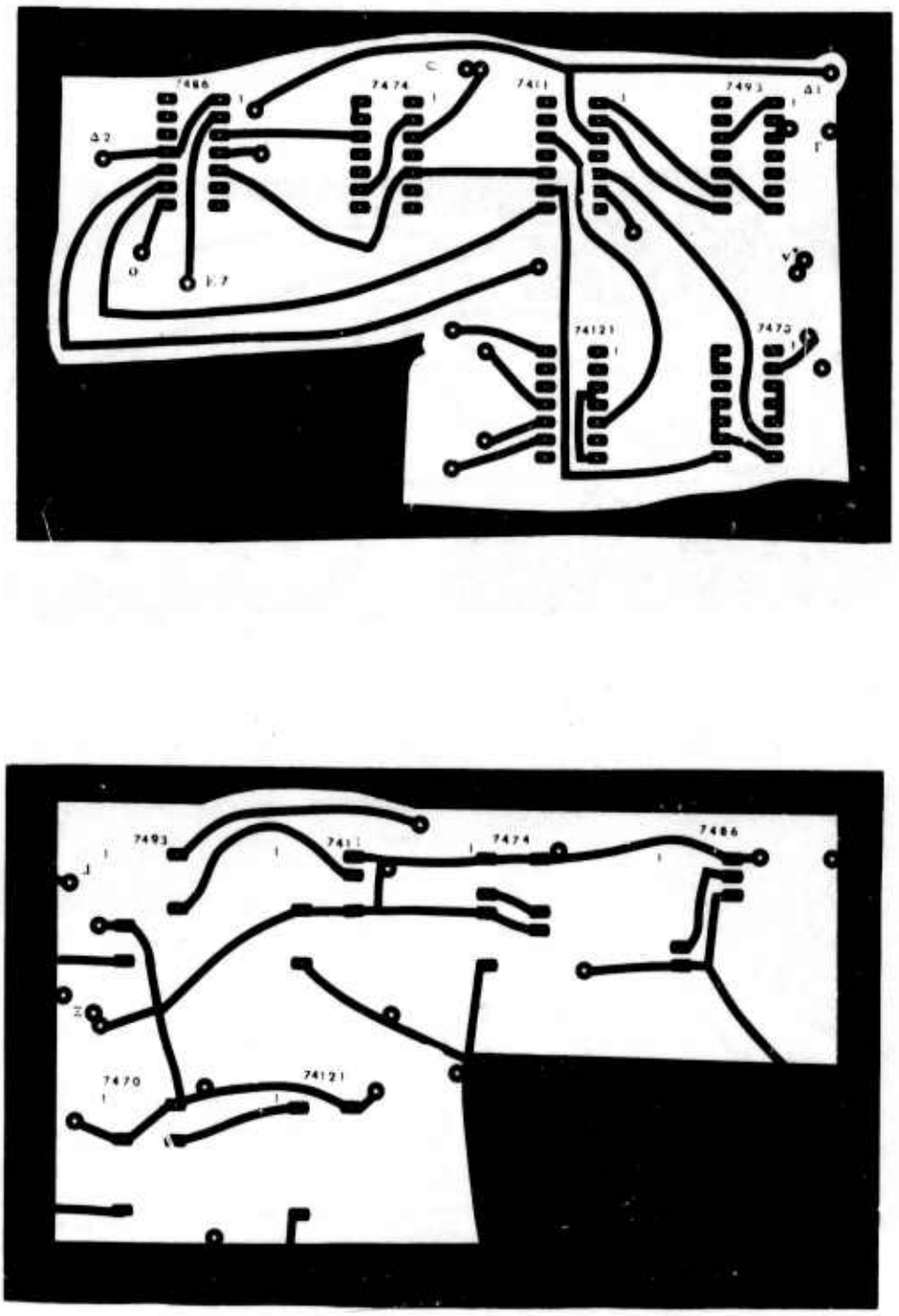


Figure 33. Encoder Printed Circuit Board 2

Now for the decoder, which is shown in Figure 28, a 10-bit shift register for the syndrome register, a 24-bit shift register to buffer the received signal, thirteen 2-input EXCLUSIVE OR gates, 2 gates, G1 and G2, a 6-input NOR gate to correct the burst errors, and a switch to disable the NOR gate during the first 24 clock pulses and to allow the NOR gate output to correct the errors during the next 24 clock pulses was needed. N7474 D flip-flops were used for the syndrome register, and N74164 shift registers were used for the buffer register. The N74164 is an 8-bit parallel-out serial-in shift register. For the EXCLUSIVE OR gates the N7486 was used and the N7405 hex inverter with open collector output was used as the 6-input NOR gate. The switch which controls the NOR gate is similar to the switch used in the encoder. This switch uses an N7493 and an N7472 (JK master slave flip-flop) to give a 5-bit counter which counts the desired 24 clock pulses. The output of this counter that corresponds to the $24^{\underline{th}}$ count feeds an N7400, a 2-input NAND gate, which will allow the NOR gate to control G1 and correct errors. An N7400 was also used for G1. The feedback of the counter turns the counter off, and the count of 24 is held until a clear button is activated. The decoder was mechanized on a single printed circuit board, and the circuit diagram for this decoder is given in Figure 34. Figure 35 gives the printed circuit board constructed from this circuit diagram. The switch, which is shown in dashed lines in Figure 34, is not shown on the printed circuit board as it was added later. A two-sided board was again used to minimize external wiring.

For future work CMOS (complementary MOSFET) logic would be desirable because of its lower power requirements, since switching speed is not critical. CMOS power levels are at the microwatt level rather than the milliwatt level for TTL logic, making it more suitable for battery-powered systems.

The printed circuit boards for the simulator used to demonstrate the encoding and decoding operations are shown in Figure 36. Also, light emitting diodes (LED) were used to display the ones and zeros of the encoded message and the corrected word. The printed circuit boards for the display are shown in Figure 37.

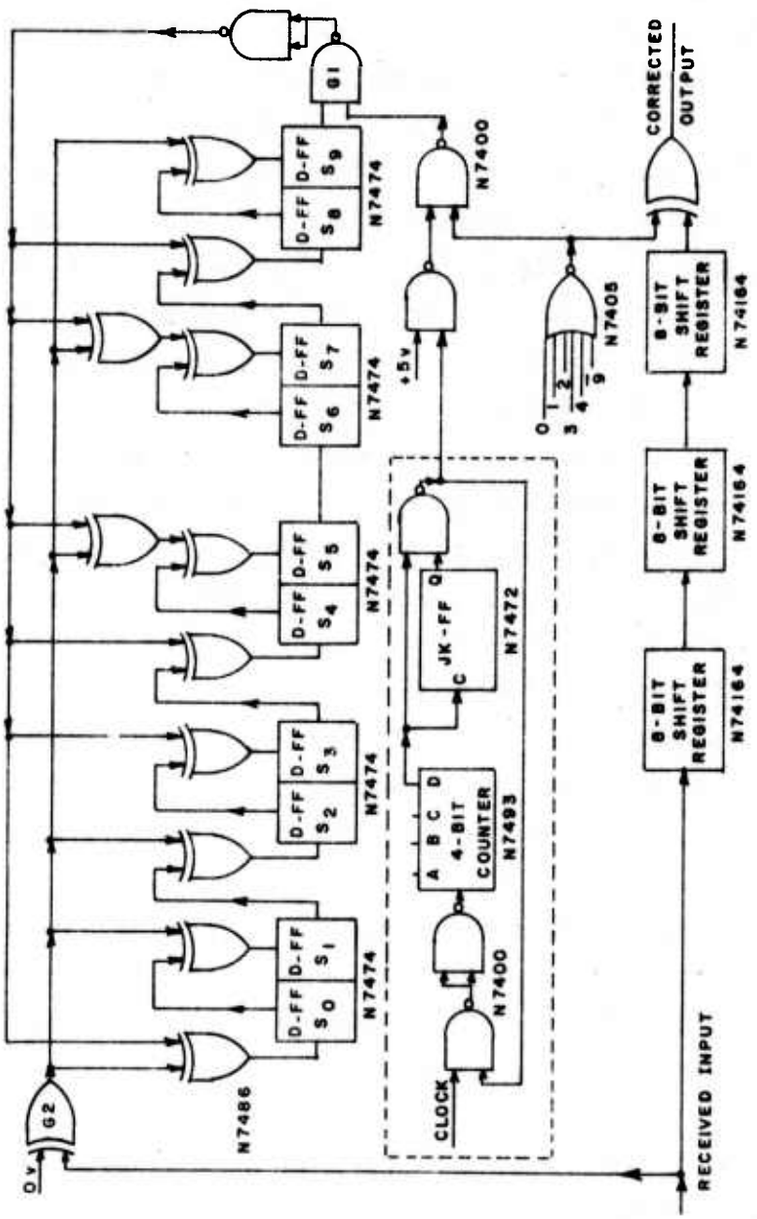


Figure 34. Decoder Circuit Diagram

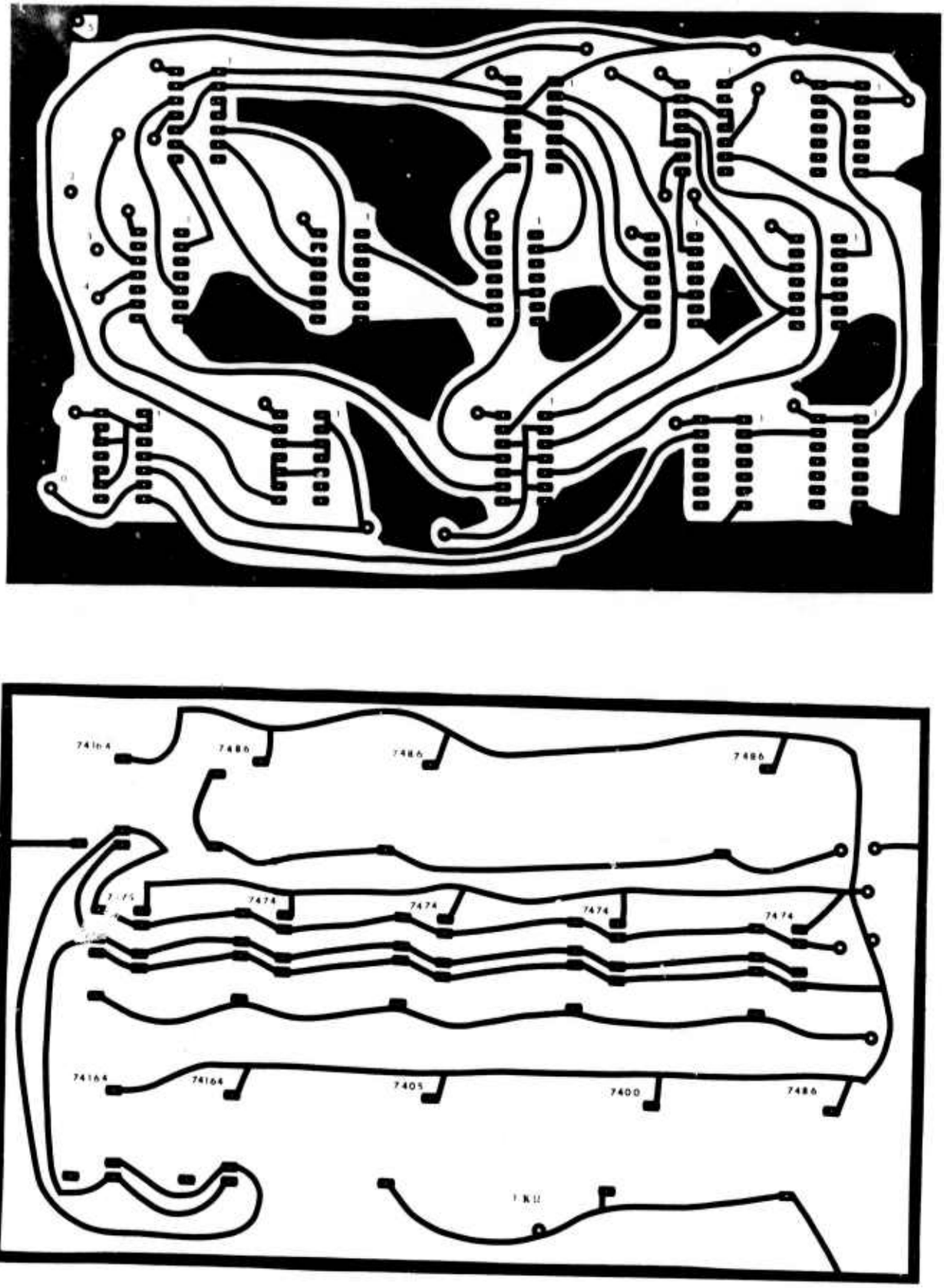


Figure 35. Decoder Printed Circuit Board

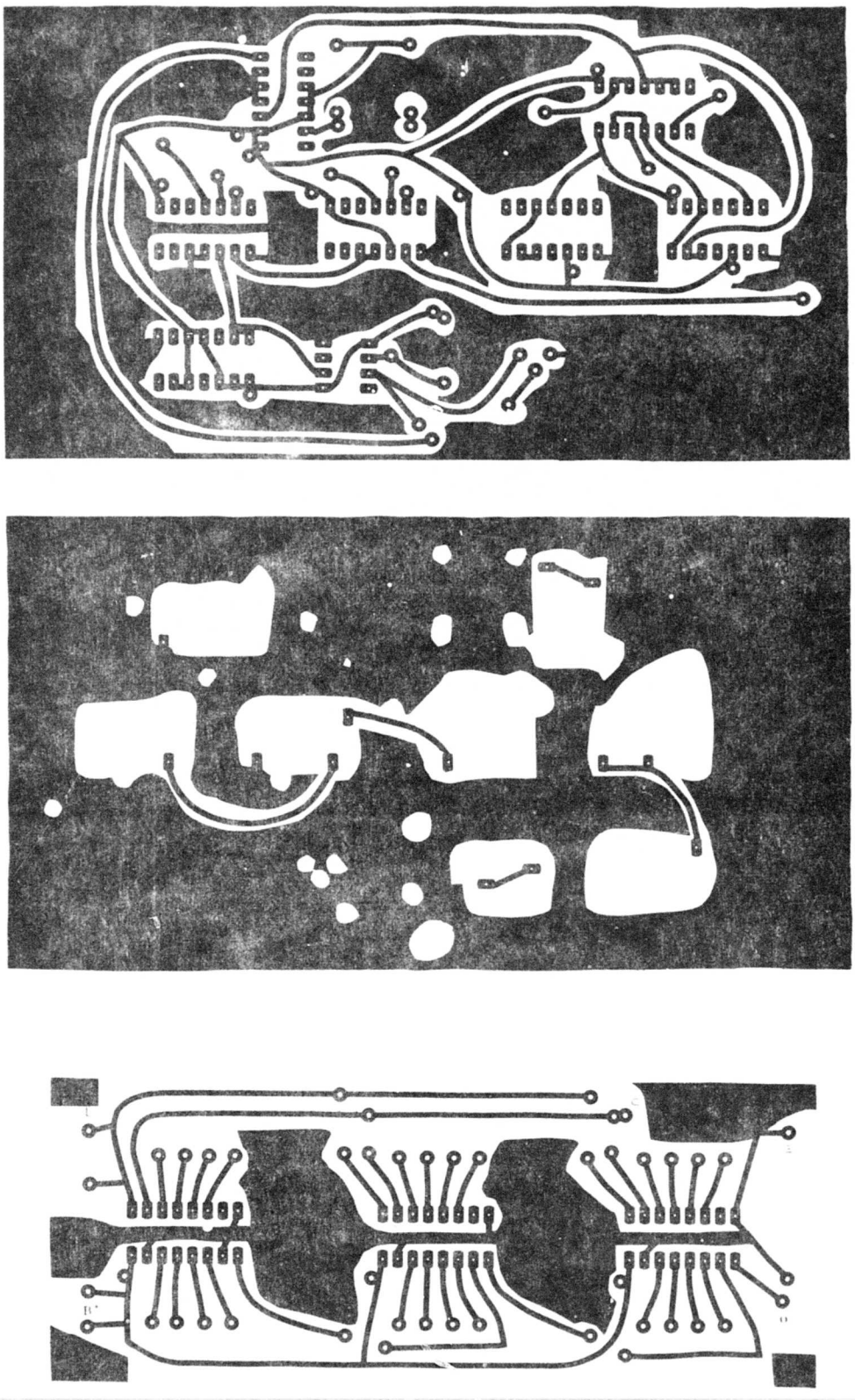


Figure 36. Simulator Printed Circuit Boards

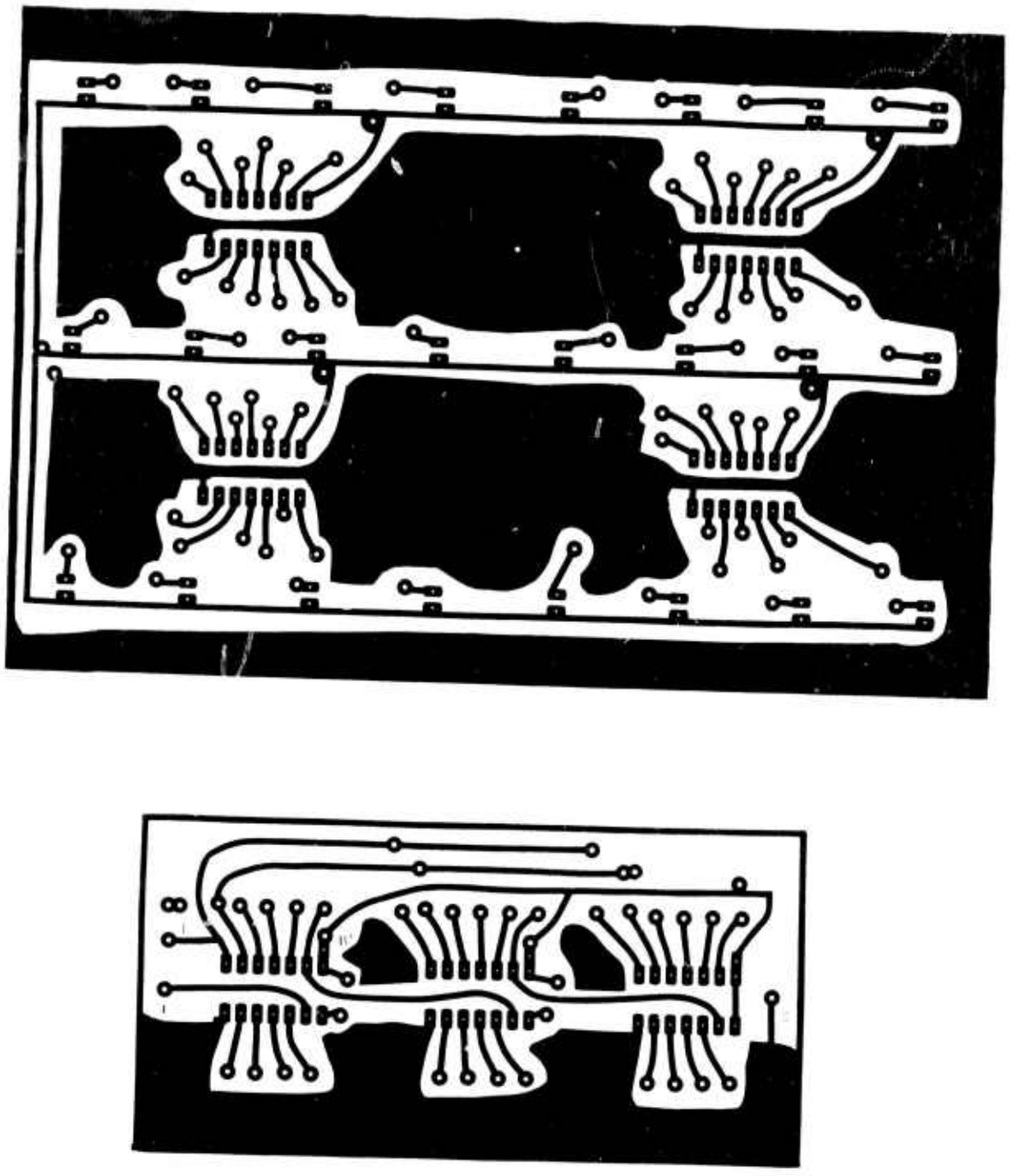


Figure 37. Display Printed Circuit Boards

SECTION VII

NON-CYCLIC BLOCK CODES

For cyclic coders, especially in the decoding operation, there is a delay in the processing of the encoded or decoded word. Depending upon the specific decoder, there can be a delay of up to two word lengths. Also, since the total word length is 48 bits of which 24 will be used for coding purposes and others passed unaltered, to use a cyclic code would require reformatting the word such that the 24 bits for coding are on one end of the word. An alternative to cyclic coders would be block coders which would not require reformatting the 48-bit word. Using a block code also enables the encoding and decoding to be accomplished with essentially no delay since there is no feedback shift register and the coding and decoding is done in parallel. The generator matrix, which corresponds to the generator polynomial for cyclic codes, for the (24,14) noncyclic block burst 5 error-correcting code is given as

$$G = \begin{bmatrix}
1&0&0&1&1&1&0&1&1&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&1&0&0&1&1&1&0&1&1&0&1&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&1&1&1&0&1&0&1&1&0&0&1&0&0&0&0&0&0&0&0&0&0&0\\
1&1&0&0&0&0&0&0&1&1&0&0&0&1&0&0&0&0&0&0&0&0&0&0\\
1&1&1&1&1&1&0&1&1&1&0&0&0&0&1&0&0&0&0&0&0&0&0&0\\
1&1&1&0&0&0&1&1&0&1&0&0&0&0&0&1&0&0&0&0&0&0&0&0\\
1&1&1&0&1&1&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0\\
0&1&1&1&0&1&1&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0\\
0&0&1&1&1&0&1&1&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0\\
0&0&0&1&1&1&0&1&1&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0\\
0&0&0&0&1&1&1&0&1&1&0&0&0&0&0&0&0&0&0&0&1&0&0&0\\
1&0&0&1&1&0&1&0&1&1&0&0&0&0&0&0&0&0&0&0&0&1&0&0\\
1&1&0&1&0&0&0&1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0\\
1&1&1&1&0&1&0&1&1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&1
\end{bmatrix} \qquad (84)$$

This 14 X 24 generator matrix determines the connections for the encoder which is shown in Figure 38, where the 1's indicate a connection and 0's indicate no connection. It can be noted that the only delay in this encoding process is the delay through the multiple input EXCLUSIVE OR gate. The 10 X 24 parity check matrix which determines the decoder connections is obtained from Equation (84) as

$$H = \begin{bmatrix} 1&0&0&0&0&0&0&0&0&0&1&0&1&1&1&1&1&0&0&0&0&1&1&1 \\ 0&1&0&0&0&0&0&0&0&0&0&1&0&1&1&1&1&1&0&0&0&0&1&1 \\ 0&0&1&0&0&0&0&0&0&0&0&0&1&0&1&1&1&1&1&0&0&0&0&1 \\ 0&0&0&1&0&0&0&0&0&0&1&0&1&0&1&0&0&1&1&1&0&1&1&1 \\ 0&0&0&0&1&0&0&0&0&0&1&1&1&0&1&0&1&0&1&1&1&1&0&0 \\ 0&0&0&0&0&1&0&0&0&0&1&1&0&0&1&0&1&1&0&1&1&0&0&1 \\ 0&0&0&0&0&0&1&0&0&0&0&1&1&0&0&1&0&1&1&0&1&1&0&0 \\ 0&0&0&0&0&0&0&1&0&0&1&0&0&0&1&1&0&0&1&1&0&0&0&1 \\ 0&0&0&0&0&0&0&0&1&0&1&1&1&1&1&0&0&0&0&1&1&1&1&1 \\ 0&0&0&0&0&0&0&0&0&1&0&1&1&1&1&1&0&0&0&0&1&1&1&1 \end{bmatrix} \tag{85}$$

The decoder is shown in Figure 39, and the delay is only slightly more because of two stages of EXCLUSIVE OR gates and the logic decoder.

The price that is paid for the fast encoding and decoding is complexity which is shown in the encoder and decoder and also in Figure 40, depicting the logic decoder in detail. As shown, the encoder requires a 14-bit register for the message word, a 24-bit register for the code word, 96 connections from the message word to the code word, and 10 multiple input EXCLUSIVE OR gates. The decoder requires a 24-bit register for the received word, a 10-bit register for the syndrome, 92 connections from the received word to the syndrome register, 10 multiple input EXCLUSIVE OR gates, and a 10-line to 1024-line logic decoder which can be constructed from sixty-six 4-line to 16-line logic decoders (N74154). The decoder also requires a 14-bit register for the corrected message word, 14 more multiple input EXCLUSIVE OR gates, and 175 connections (only message bits corrected) from the logic decoder to the corrected message register.

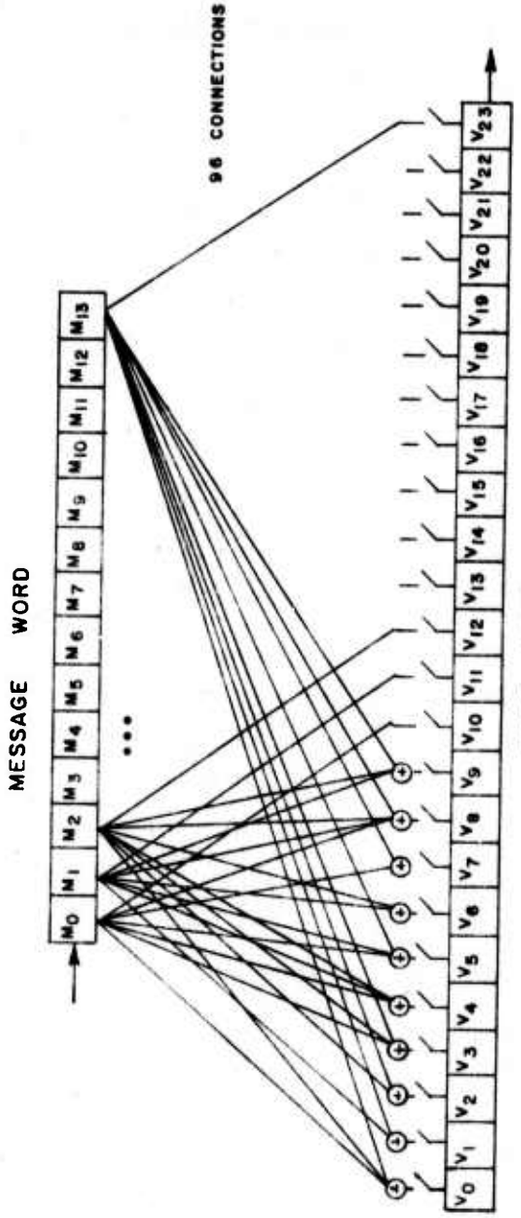


Figure 38. Encoder for (24,14) Noncyclic Block Burst 5 Error-Correcting Code

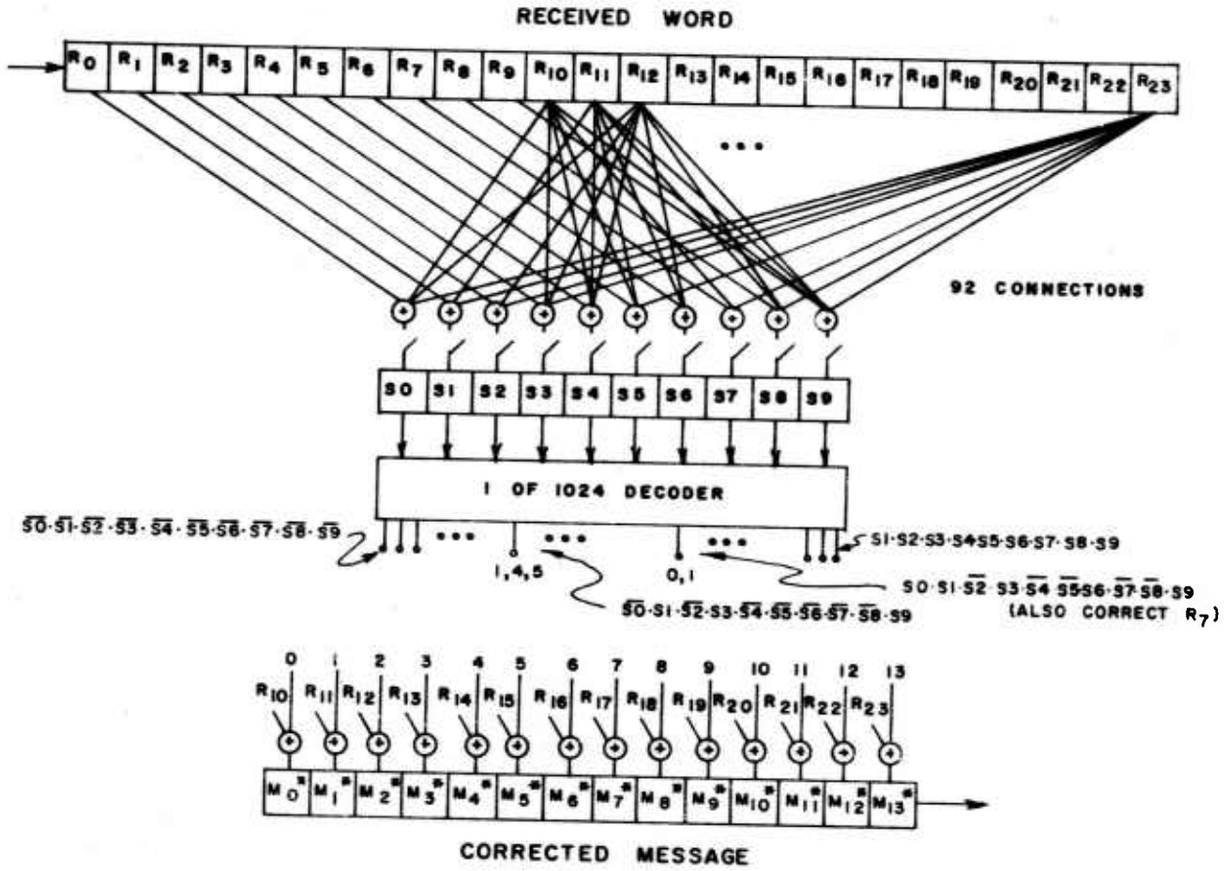


Figure 39. Decoder for (24,14) Noncyclic Block Burst 5 Error-Correcting Code

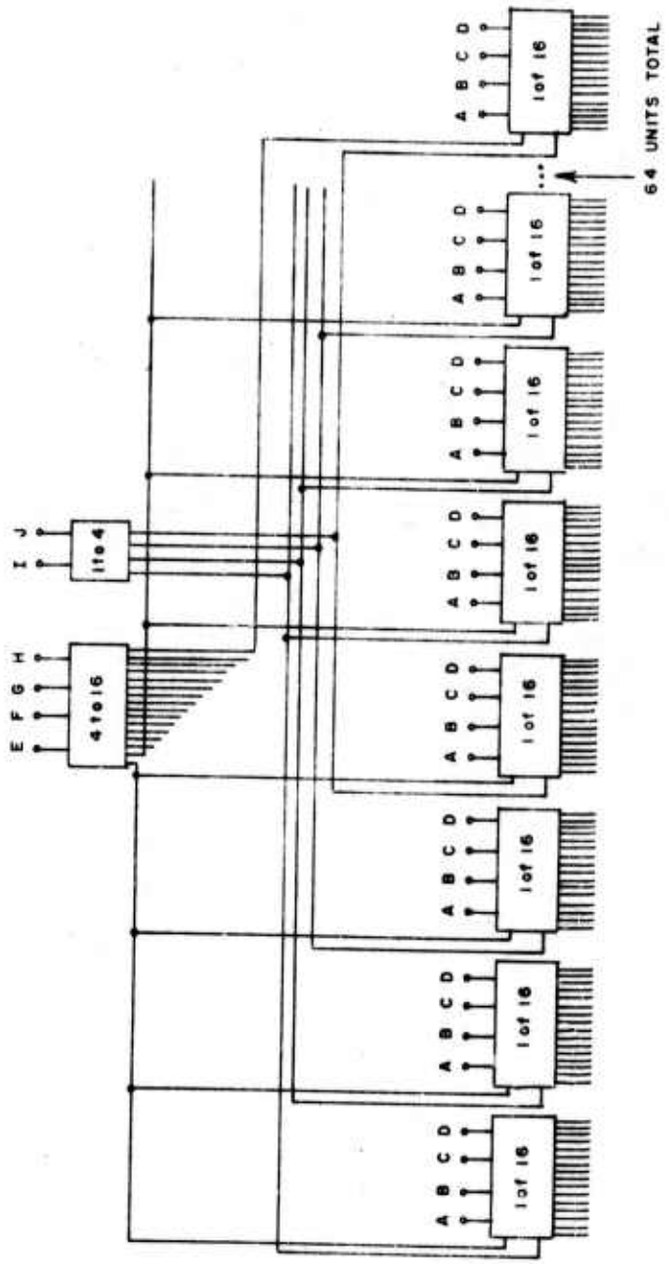


Figure 40. 10-Line to 1024-Line Logic Decoder

As a comparison of how the complexity increases with increasing burst error-correcting capability, the (24,14) code is compared to a (24,17) noncyclic block burst 3 error-correcting code. The 17 X 24 generator matrix is

$$G = \begin{bmatrix}
\overline{1} & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \overline{0} \\
0 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
1 & 1 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
0 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\
\underline{0} & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \underline{1}
\end{bmatrix} \tag{86}$$

and the encoder is given in Figure 41. From Equation (86) the 7 X 24 parity check matrix is

$$G = \begin{bmatrix}
\overline{1} & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & \overline{0} \\
0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 1 & 0 & 0 & 1 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 1 & 0 & 0 & 1 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 1 & 1 \\
\underline{0} & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & \underline{1}
\end{bmatrix} \tag{87}$$

and the decoder is shown in Figures 42 and 43.

The encoder requires a 17-bit message register, a 24-bit code word register, 76 connections between these registers, and 10 multiple input

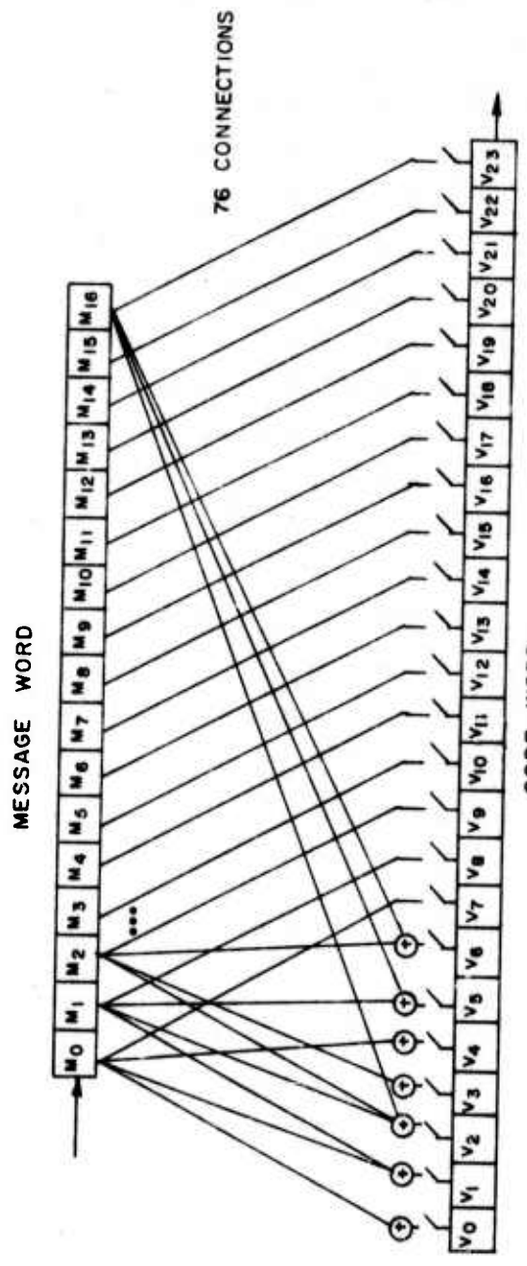


Figure 41. Encoder for (24,17) Noncyclic Block Burst 3 Error-Correcting Code

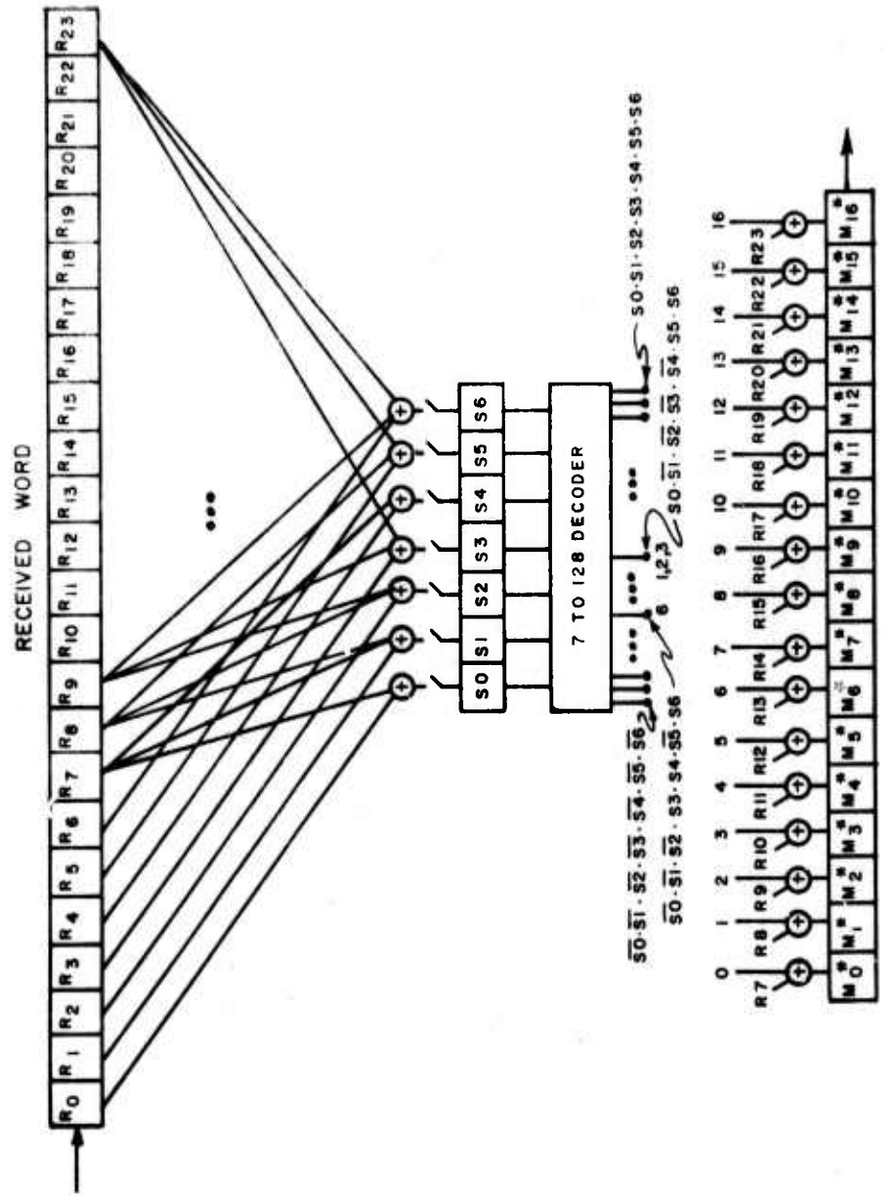


Figure 42. Decoder for (24,17) Noncyclic Block Burst 3 Error Correcting Code

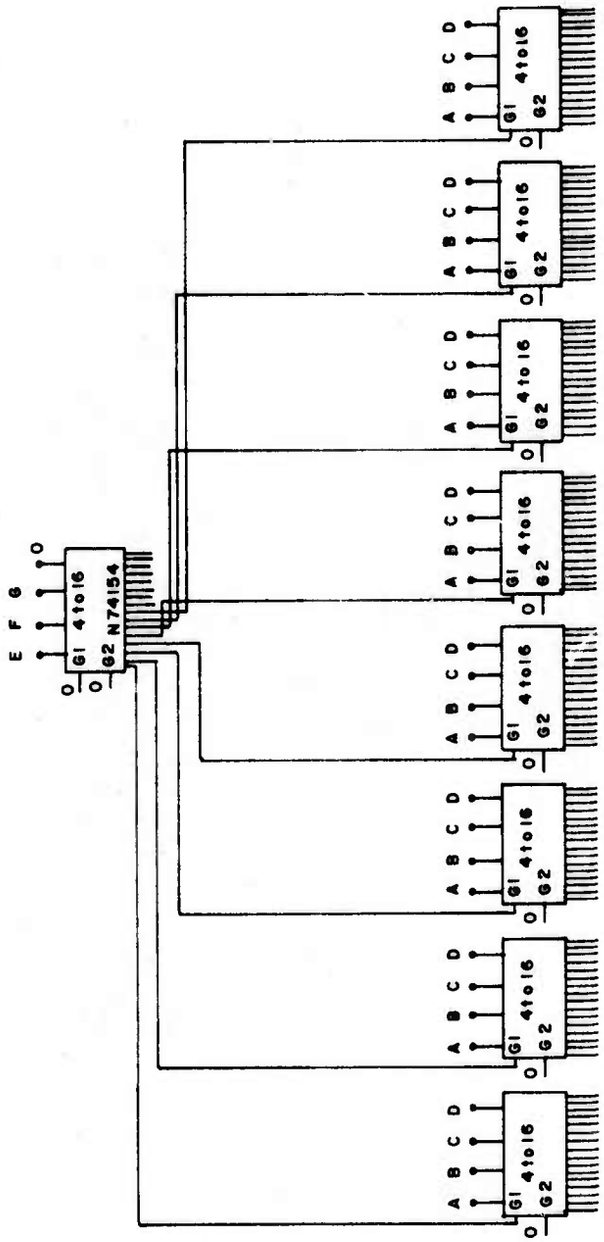


Figure 43. 7-Line to 128-Line Logic Decoder

EXCLUSIVE OR gates. The decoder requires a 24-bit received word register, a 7-bit syndrome register, 66 connections between these registers, a 7-line to 128 line logic decoder (constructed from nine 4-line to 16-line logic decoders), a 17-bit corrected message register, 63 connections from the logic decoder to the corrected message register, and a total of 24 EXCLUSIVE OR gates.

REFERENCES

1. Stein, S., and Jones, J., Modern Communication Principles, McGraw-Hill, New York, NY, 1965.

2. Peterson, W., and Weldon, E., Error Correction Codes Second Edition, The MIT Press, Cambridge, Mass., 1972.

3. Kasami, T., "Optimum Shortened Cyclic Codes for Burst-Error-Correction," IEEE Trans. on Information Theory, IT-9, pp. 105-109, April, 1963.

4. Lin, S., An Introduction to Error-Correcting Codes, Prentice-Hall, Englewood Cliffs, NJ, 1970.

5. Trafton, P., et al., "Error Protection Manual for AFCS," Report No. AFCS-TR-73-1, November, 1972.

INITIAL DISTRIBUTION

ASD/ENO	1
ASD/ENYS	1
ASD/ENV	2
ASD/SD	1
AUL (AUL-LSE-70-239)	2
DARPA (TIO)	1
NAVAIR SYS COMD	1
COMDR NWC/CODE 753	2
DDC	2
AFATL/DL	1
SDT	1
AFATL/DLMT	1
AFATL/DLMI	1
Hq USAF/SAMI	1
TAWC/TRADOCLO	1
AFATL/DLOSL	2
ASD/ENYEIM	1
Ogden ALC/MMNOP	2
AFWL/LR	2

(The reverse of this page is blank)

UNCLASSIFIED
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)
Electrical Engineering Department
University of Missouri
Columbia, Missouri

2a. REPORT SECURITY CLASSIFICATION
Unclassified

2b. GROUP

3. REPORT TITLE
CODING FOR FREQUENCY - SHIFT - KEYED (FSK) COMMUNICATION SYSTEM.

4. DESCRIPTIVE NOTES (Type of report and inclusive dates)
Final Report, Jan[illegible] - December 1973

5. AUTHOR(S) (First name, middle initial, last name)
John J. Komo

6. REPORT DATE
December 1973

7a. TOTAL NO. OF PAGES
142

7b. NO. OF REFS
5

8a. CONTRACT OR GRANT NO.
F08635-73-C-0078

8b. PROJECT NO.
AF-670D

8c. Task No. 679

8d. Work Unit No. 025

9a. ORIGINATOR'S REPORT NUMBER(S)

9b. OTHER REPORT NO(S) (Any other numbers that may be assigned this report)
AFATL-TR-73-245

10. DISTRIBUTION STATEMENT
Distribution limited to U. S. Government agencies only; this report documents test and evaluation; distribution limitation applied December 1973. Other requests for this document must be referred to the Air Force Armament Laboratory (DLMT), Eglin Air Force Base, Florida 32542.

11. SUPPLEMENTARY NOTES
Available in DDC

12. SPONSORING MILITARY ACTIVITY
Air Force Armament Laboratory
Air Force Systems Command
Eglin Air Force Base, Florida 32542

13. ABSTRACT

Coding schemes for the correction of random errors or burst errors for fixed block length digital data words are presented in this report. The word format considered is a 48-bit word with six 8-bit subwords, and the codes developed are for the correction of errors in two subwords with a third subword for the parity bits. Several codes are considered and analyzed for their error correcting capability. Also, computer simulations were carried out on several of these codes, and a burst length 5-error correcting coding system was mechanized. Performance gains based on comparisons with uncoded, coherent frequency-shift-keyed system are examined.

DD FORM 1 NOV 65 1473

UNCLASSIFIED
Security Classification

401 391

14 KEY WORDS	LINK A		LINK B		LINK C	
	ROLE	WT	ROLE	WT	ROLE	WT
Frequency-Shift-Keyed Communication System						
Random Error-Correcting Codes						
Burst Error-Correcting Codes						
Burst Code Construction						
Non-Cyclic Block Codes						