RICE UNIVERSITY

THE EVALUATION OF A HYBRID CODING SCHEME

by

Forrest Fox

A THESIS SUBMITTED
IN PARTIAL FULFILLMENT OF THE
REQUIREMENTS FOR THE DEGREE OF

MASTER OF SCIENCE
IN
ELECTRICAL ENGINEERING

Thesis Director's signature:

Frank L. Hubard

Houston, Texas

May, 1970

Abstract

THE EVALUATION OF A HYBRID CODING SCHEME

Forrest Fox

The performance of a hybrid sequential-algebraic coding scheme is evaluated taking into account the probability of error, the complexity of implementation, and the overall rate. The hybrid scheme is one suggested by Huband and Jelinek based on a similar proposal by Falconer. Upper and lower bounds on the performance of a straight sequential coding scheme have been utilized. The corresponding relationships for the algebraic portion of the hybrid code and the complete hybrid code are obtained. The performance of the hybrid scheme is compared to the performance of a straight sequential scheme, and found to obtain up to several orders of magnitude improvement in probability of error for the same overall rate and complexity of implementation.

TABLE OF CONTENTS

INTRODUCTION

The performance of any coding scheme can be measured by the inherent trade-off between speed, cost and reliability; i. e. increasing reliability generally requires increasing cost and/or decreasing speed. The exact relationship of these three parameters for a given code depends upon the communication channel to be utilized.

Both of the common coding techniques, sequential and algebraic, have sufficiently severe limitations to restrict their utilization in many practical communication systems. The algebraic codes perform at reasonable rates, cost, and probability of error only for relatively small channel crossover probabilities. The sequential codes perform better for high crossover probabilities, but the variability in the number of decoding computations can require prohibitive storage allocation to obtain an acceptable overall probability of error.[3] A hybrid sequential-algebraic coding scheme proposed by Huband and Jelinek[1] based on a similar proposal by Falconer[2] is capable of providing significant improvement in the probability of error when compared to a straight sequential code utilizing the same channel, at the same rate, and with comparable complexity.

The basic idea of the hybrid scheme is indicated in Figure 1. The "inner" convolutional encoder-channel-sequential decoder combination presents an apparent erasure channel to the "outer" Reed-Solomon algebraic encoder-decoder pair. The inner combination

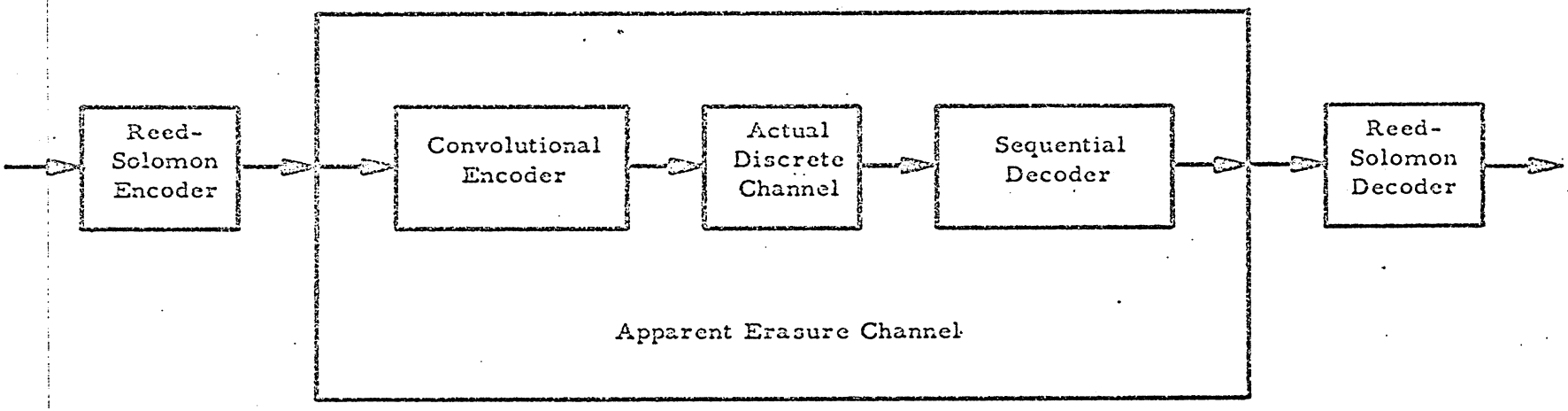


Figure 1

Schematic of a Hybrid Sequential-Algebraic Scheme

appears to be an erasure channel in the sense that any data going into the apparent channel is either received correctly at the output of the channel or is declared erased. The performance of the Reed-Solomon algebraic code for an erasure channel is analyzed in the Appendix. The performance of the hybrid code is analyzed and shown to compare favorably to the straight sequential code.

THE SEQUENTIAL CODE

The convolutional encoder in a sequential coding scheme maps each set of k_o input bits into a set of n_o output bits to be communicated over the channel, where k_o and n_o are small integers. If ν is the constraint length of the code, the mapping is done such that each set of n_o output bits depends only on the k_o input bits being encoded, and on the νk_o previously encoded bits. After Γk_o input bits have been encoded and transmitted, the encoder transmits a set of $n_o t$ resyncronization bits. The resyncronization bits are used to insure that the last νk_o information bits will be correctly decoded. The rate of the code is

$$R_o = \Gamma k_o/(\Gamma + t)n_o .$$

Reasonable values of Γ and t are 1000 and 25 respectively; therefore $(\Gamma + t) \approx \Gamma$ and $R_o \approx k_o/n_o \triangleq R_s$.

The performance of a sequential decoding algorithm depends more on the structure of the available buffer storage, etc. than on the specific algorithm involved. This discussion will be valid, in particular, for both the Jelinek and the Fano decoding schemes. It is assumed that the sequential decoder operates at a cycle rate of σ cycles per second, and that the average number of cycles available for decoding each information bit is Q_s cycles/bit. Thus the total time available for decoding a block of $k_o\Gamma$ information bits is $k_o\Gamma Q_s/\sigma$ seconds. It is assumed that the blocks are received at this rate, and the decoding is to be done in real time; i. e. the decoding

of a block must be completed in $k_o \Gamma Q_s/\sigma$ seconds. Failure to complete the decoding in the allotted time is the most probable source of error in a sequential coding scheme.

If the decoding is completed in the allotted time, some of the decoded information bits may be in error. An undetected error may occur in the first $(\Gamma - \nu)\, k_o$ information bits if the constraint length of the code is too small, or an error may occur in the last νk_o information bits if the number of resyncronization bits is too small. By an appropriate choice of ν and t, both the probability of undetected error, and of error in the last νk_o information bits can be made negligible with respect to the probability of incomplete decoding without significantly adding to the encoder complexity or decreasing the rate. Therefore, we shall assume that the probability of decoding error in the sequential code is the probability of incomplete decoding.

Both the upper and lower bounds[4, 5] on the probability P_{id}, of incomplete decoding for a sequential code have the form

$$P_{id} \propto (\Gamma/d)^{1-\gamma} (Q_s - q)^{-\gamma}$$

where γ and q are functions of the channel and the transmission rate, and d is a constant between 1 and ν. Experimental[6] evidence indicates that for sequential rates below R_{comp}, $\gamma = R_{comp}/R_s$ and q is approximately 1. For the purpose of analysis we shall set d=1 and use the relationship

$$P_s(e) = \Gamma^{1-\gamma_s} (Q_s - 1)^{-\gamma_s} \qquad (1)$$

where $P_s(e)$ is the probability of error for the sequential code, and $\gamma_s = R_{comp}/R_s$.

THE REED-SOLOMON ERASURE CORRECTING CODE

A codeword in the Reed-Solomon code is a sequence, $\{x_i\}$, of N symbols in GF(2^m) which satisfy the T equations

$$\sum_{i=0}^{N-1} x_i \alpha^{ij} = 0 \qquad 1 \le j \le T$$

where α is a primitive element of GF(2^m) and all operations are field operations. The word length, N, of the code is 2^m-1, the minimum distance, d, is T+1, and the number of information symbols, K, is N-T. Since the minimum distance is d, as many as d-1=T erasures may be corrected. The rate, R_a, of the code is K/N.

The encoding and decoding operations for the Reed-Solomon code are carried out by digital circuitry. In order to evaluate the complexity, or cost, of the encoder and decoder, both the amount of hardware and the required hardware speed must be taken into account. For the encoder, at least, the power consumption may also be an important cost consideration. With the availability of low cost integrated circuits containing multiple logic functions, the number of IC packages may be more important than the number of logic functions. However, since the extent of circuit integration and required power is changing very rapidly, a cost function involving the amount of hardware and the required speed is used; i.e. the product of amount of hardware and operations-required-per-information-bit is used as a cost function. This type of complexity index has been evaluated by John E. Savage[7, 8] and found to be a good measure of cost.

As obtained in the Appendix, the decoding can be segmented into six steps. In Table 1 the number of flip-flops, exclusive OR gates and AND gates, and the number of operations required per decoded word are shown for each of the six steps. Assuming that the cost of an exclusive OR gate is approximately twice that of an AND gate and half that of a flip-flop, we obtain from Table 1 that the cost per decoded word for each of the six decoding steps is as follows:

1) determination of $\{U_i\}$ and $\{U_i^{-1}\}$	$4m(2T+3)N$
2) computation of $\{S_j\}$	$6mTN$
3) computation of $\{\sigma_i\}$	$7m^2T^2$
4) computation $\{A_p^{-1}\}$	$20mT(mT+N-1)$
5) computation $\{B_p\}$	$14m^2T^2$
6) computation $\{V_p\}$	$7m^2T$

Since each codeword contains mK information bits, the total cost, Q_a, per decoded information bit can be written as

$$Q_a = c\left[4(2T+3)N + 6TN + 7mT^2 + 20T(mT+N-1) + 14mT^2 + 7mT\right]/K$$

$$Q_a = c\left[34TN + 41mT^2 + 12K + T(7m-8)\right]/K$$

where c is a measure of the cost of a single AND gate. In terms of the algebraic rate, $R_a = K/N$ we have

$$Q_a = c\left[34T + 41mT(1-R_a) + 12R_a + (7m-8)(1-R_a)\right]/R_a \,. \qquad (2)$$

Refering to Figure A2 in the Appendix, the encoding circuit requires 2mT flip flops, 2mT exclusive OR gates, and K shifts per encoded word. Using the same cost function, the cost of the algebraic encoder is 12cT per encoded information bit.

		Hardware			
	Refer to Figure	flip flops	exclusive OR gates	AND gates	Operations per decoded word
determination of $\{U_i\}$ and $\{U_i^{-1}\}$	A3	2m(T+1)	2m	--	N
computation of $\{S_j\}$	A4	mT	mT	--	N
computation of $\{\sigma_i\}$	A5	mT	mT	mT	mT
computation of $\{A_p^{-1}\}$	A6	3mT	3mT	2mT	mT+N-1
computation of $\{B_p\}$	A7	2mT	2mT	2mT	mT
computation of $\{V_p\}$	A8	mT	mT	mT	m

Table 1

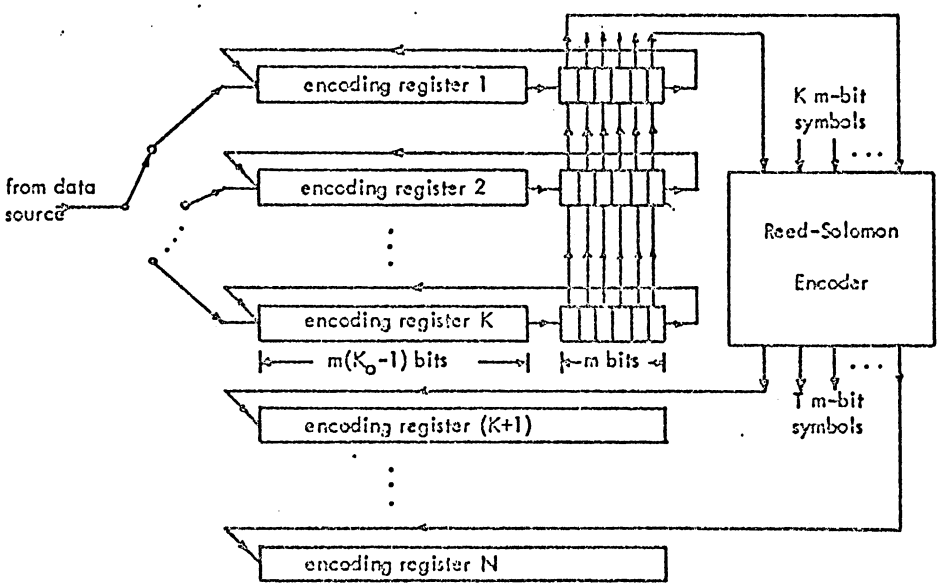
THE HYBRID SCHEME

The hybrid coding scheme suggested by Huband and Jelinek is composed of a sequential "inner" code and a Reed-Solomon "outer" code. As with most hybrid schemes, the inner code essentially conditions the channel over which the outer code operates.

The encoding process for the hybrid scheme proceeds as shown in Figures 2 and 3. The information to be communicated is assumed to exist as blocks of $K_o m$ binary digits, i. e. as blocks of K_o m-bit "symbols." A group of K blocks is stored in K buffer registers of $K_o m$ bits each. After all KK_o symbols have been loaded into the registers, a Reed-Solomon encoder takes the first symbol of m bits from the front end of each of the K registers and obtains a set of $T=N-K$ "check symbols," each of which is shifted into the tail of an additional $K_o m$-bit register. The original K registers are then right-shifted (end-around), and the second set of digits is encoded. This proceeds until the entire set of KK_o symbols has been successfully encoded. (Note that only the last m bits of the $K_o m$-bit registers need be accessible.)

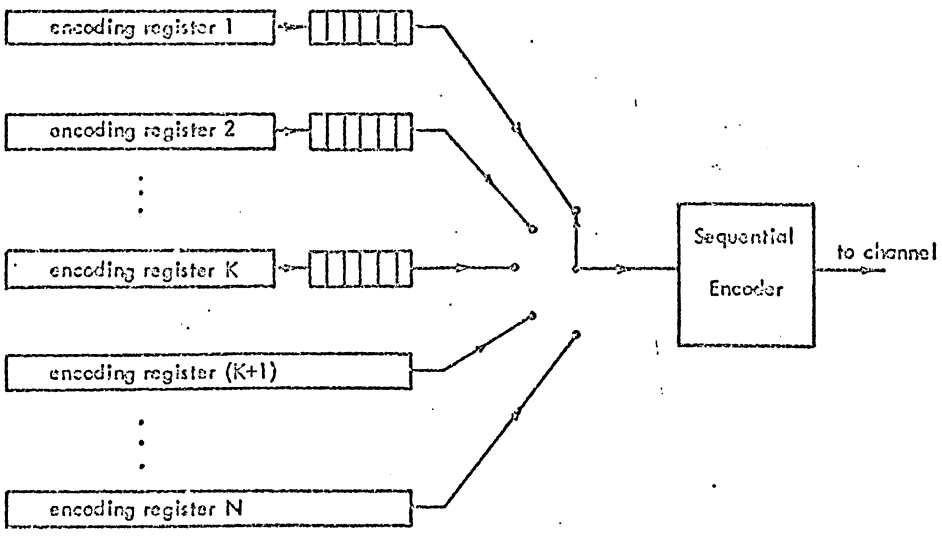
Each register or "track" is then coded in turn, block sequentially (with each track coded independently of the others), and communicated through the discrete memoryless channel. The coded length of each track is $N_o = K_o + T_o$ symbols.

The decoding process takes place in the opposite order. If a given track of mK_o information bits cannot be completely decoded



The Encoding Process (A)

Figure 2



The Encoding Process (B)

Figure 3

by the sequential decoder in a specified number, mK_oQ_s, of decoding operations, that track is declared an erasure. If the sequential decoder completes the decoding of a track, then with high probability the decoded bits are correct. Any number of erasures in a block, up to a maximum of T will be successfully decoded by the Reed-Solomon decoder.

The advantages obtained by this particular combination of codes are due largely to the ability to implement a sequential decoder so that the only significant failure is from incomplete decoding. As a result of this apparent channel transformation an erasure-correcting R-S decoder can be used which is inherently much simpler than an error-correcting decoder. The sequential and Reed-Solomon codes complement each other in a number of other significant ways also.[3] Sequential codes are easily adapted to non-symmetric channels whereas Reed-Solomon erasure-correcting codes are not. In the hybrid scheme the Reed-Solomon code is oblivious to any asymmetry in the channel. Sequential codes can achieve erasure probabilities of about 10^{-3} with relatively little computation even for large channel crossover probabilities. However, to reduce the erasure probability from 10^{-3} to 10^{-6} would require about 1000 times the number of computations. Reed-Solomon codes, on the other hand, are most effective for erasure probabilities less than or equal to 10^{-3}. In addition, the Reed-Solomon code is a maximum code in the sense that it can correct as many erasures as it has parity check symbols, and it is a systematic code which implies that if more than the

correctable number of erasures occurs, only those tracks actually erased are lost.

In order to evaluate the performance of the hybrid code we need an expression like equation (1) for the sequential code, or equation (2) for the Reed-Solomon code which relates the probability of error, the number of computations or cost, and the rate.

The hybrid coding scheme fails to decode all mKK_o information bits whenever more than T sequential track are erased. Therefore, the hybrid probability of error, $P_h(e)$, is the probability that more than T tracks are erased; i. e.,

$$P_h(e) = \sum_{j=T+1}^{N} \binom{N}{j} P_s(e)^j \left(1-P_s(e)\right)^{N-j} \tag{3}$$

where $P_s(e)$ is the probability that a single sequential track is erased. Since the Reed-Solomon code is systematic, if $j > T$ erasures occur, then the number of tracks erased will be exactly j. The probability of a given track being erased is j/N times the probability of j erasures occurring. Taking this factor into account in the hybrid probability of error, equation (3) becomes

$$P_h(e) = \sum_{j=T+1}^{N} \left(\frac{j}{N}\right)\binom{N}{j} P_s(e)^j \left(1-P_s(e)\right)^{N-j} . \tag{4}$$

For reasonably small track erasures probabilities ($P_s(e) \leq 10^{-2}$), the probability of more than T erasures occurring is very nearly the probability of T+1 erasures occurring. For the cases of interest we can, with very little error, simplify equation (4) as

$$P_h(e) \approx \binom{N-1}{T} P_s(e)^{T+1} \left(1-P_s(e)\right)^{N-T-1} . \tag{5}$$

We define the overall rate of the hybrid code as the product of the sequential and algebraic rates; i. e.

$$R_t \triangleq R_s \cdot R_a = K_o K / N_o N \tag{6}$$

and we define γ_t as the ratio of R_{comp} to the overall rate; i. e.

$$\gamma_t = R_{comp}/R_t. \tag{7}$$

From equations (1), (6) and (7), the probability that any sequential track in the hybrid scheme is erased is

$$P_s(e) = \Gamma_h^{1-\gamma_t R_a} (Q_s - 1)^{-\gamma_t R_a} \tag{8}$$

where Γ_h is the number of k_o-bit nodes per track for each track in the hybrid scheme.

The total cost, Q_t, of the hybrid decoder per decoded information bit can be estimated as the sum of the cost, Q_a, of the algebraic decoder per information bit, and the number of cycles, Q_s/R_a, of the sequential decoder per information bit, i. e.

$$Q_t = Q_a + Q_s/R_a \tag{9}$$

We assume that even for the sequential code alone, some storage will be provided at the decoder either as disk, tape, or registers to store incoming codewords. The input storage required for the hybrid code will therefore be an inherent part of the sequential decoder and will not be considered to contribute additional cost to the hybrid decoder.

One advantage of this particular hybrid scheme, as mentioned previously, is that the algebraic decoder is required to correct only erasures and no errors. In fact, the structure of this hybrid scheme

is such that the algebraic decoding is less difficult even than normal erasure correction. Note that in the hybrid scheme, when a sequential track is erased, the erasure affects K_o Reed-Solomon codewords. Therefore, the erasure locators are the same for the entire set of K_o codewords, and need be computed only once for the set. In addition, the coefficients of $\sigma(X)$ which depend only on the erasure locators, and the set $\{A_p^{-1}\}$ which depends only on the erasure locators and the coefficients of $\sigma(X)$, are also computed only once for a set of K_o Reed-Solomon codewords. Therefore, the cost per decoded information bit of each step in the Reed-Solomon decoder is as follows:

1) determination of $\{U_j\}$ and $\{U_i^{-1}\}$	$4(2T+3)N/KK_o$
2) computation of $\{S_j\}$	$6TN/K$
3) computation of $\{\sigma_i\}$	$7mT^2/KK_o$
4) computation of $\{A_p^{-1}\}$	$20T(mT+N-1)/KK_o$
5) computation of $\{B_p\}$	$14mT^2/K$
6) computation of $\{V_p\}$	$7mT/K$

For reasonable values of K_o, the cost of those steps which are done only once is negligible. Thus, the cost, Q_a, per decoded information bit for the Reed-Solomon decoder in the hybrid scheme is given by

$$Q_a = cT/K\ (6N+14mT+7m). \tag{10}$$

From (9) and (10)

$$Q_s = R_a Q_t - cT/N(6N + 14mT + 7m). \tag{11}$$

Equations (11), (8) and (5) make up the desired relationship for the

evaluation of the performance of the hybrid scheme. Substituting (11) and (8) into (5) gives $P_h(e)$ in terms of Q_t, γ_t, Γ_h, N, and c for all R_a such that $\gamma_t R_a \geq 1$.

The probability of error for the straight sequential scheme having the same overall rate R_t, number of nodes $\Gamma_s = K\Gamma_h$, and total complexity Q_t, is calculated by the equation

$$P_s(e) = \Gamma_s^{1-\gamma_t} (Q_t - 1)^{-\gamma_t}. \tag{12}$$

COMPARISON OF THE HYBRID AND SEQUENTIAL CODES

The performance of the hybrid code has been evaluated, using those expressions just obtained, and compared to the performance of the straight sequential code on the basis of probability of error. In comparing the performance of the two codes, each code is allowed the same number of computations per information bit, Q_t, and the same number of sequential nodes, Γ_s. In addition, the overall rate is assumed in each case to be the same fraction of R_{comp}. Thus in each case, $\gamma_t^{-1} = R_t/R_{comp}$. For a particular choice of these three parameters, the performance of the sequential code is evaluated by the expression

$$P_s(e) = \Gamma_s^{1-\gamma_t} (Q_t - 1)^{-\gamma_t} . \tag{12}$$

The number of sequential computations, the track erasure probability and, finally, the probability of error for the comparable hybrid scheme are evaluated from the expressions

$$Q_s = R_a Q_t - cT/N(6N + 14mT + 7m), \tag{11}$$

$$P_s(e) = (\Gamma_s / K)^{1-\gamma_t R_a} (Q_s - 1)^{-\gamma_t R_a} , \tag{8}$$

$$P_h(e) = \binom{N-1}{T} P_s(e)^{T+1} (1 - P_s(e))^{N-T-1} , \tag{5}$$

where N, K, and c depend on the algebraic portion of the hybrid code. (Note that the other algebraic parameters, T, R_a, and m, are all specified by the choice of N and K.)

In Figures 4 through 15 the probability of error for the two codes are compared using different values of N and K in the hybrid scheme and with c (the algebraic equation complexity coefficient) always equal zero. The effect of assuming that c=0 is to neglect the algebraic complexity. The curve for each hybrid code on the plots is denoted by the parameters of the algebraic portion of the code; i. e. (N, K) - L where L = T/N(6T + 14mT + 7m). Some degredation in the hybrid performance would certainly occur by the use of $c \neq 0$, since the number of sequential computations would be reduced by cL, therefore increasing the track erasure probability and likewise increasing the hybrid probability of error. An indication of the relative effect of non-zero c's is provided by the different values of L for the various (N, K) codes.

Certain restrictions have been adhered to in constructing these plots and it is worthwhile repeating those restrictions at this time. For $\Gamma_s = 10^4$ only N=15 is considered since $\Gamma_h = \Gamma_s / K$ and Γ_h must be on the order of 1000 or greater to justify ignoring the resyncronization node size, t. Only values of $R_a \geq \gamma_t^{-1}$ are considered since for $R_a < \gamma_t^{-1}$ sequential rates above R_{comp} are required. The expression $\gamma = R_{comp}/R_s$ does not hold for sequential rates above R_{comp}. In fact, it is precisely this assumption, $R_a \geq \gamma_t^{-1}$ which allows us to ignore the properties of the channel. For sequential rates below R_{comp}, $\gamma = R_{comp}/R_s$ independent of the channel crossover probability, but for sequential rates above R_{comp}, γ is strongly dependent on the channel.[3]

In Figures 16 through 23 the same variables are plotted, but with non-zero values of c considered. Each of these plots shows the effect on one hybrid code of c = .01 and .05.

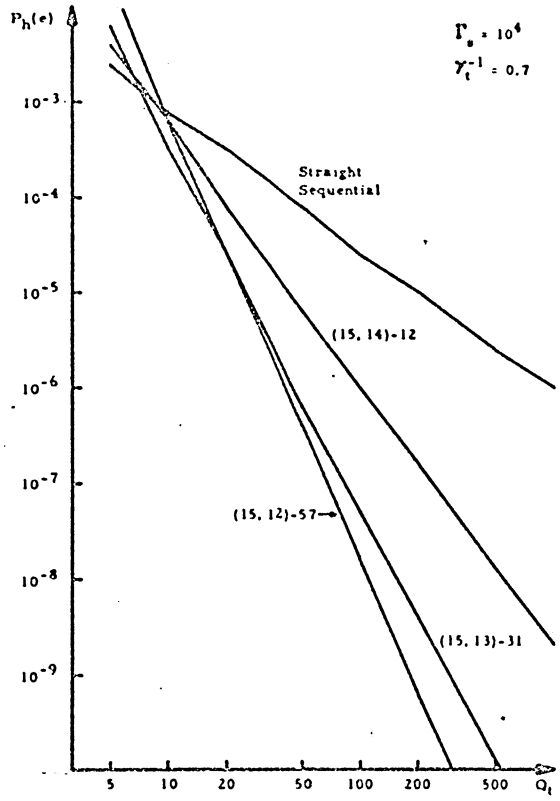


Figure 4

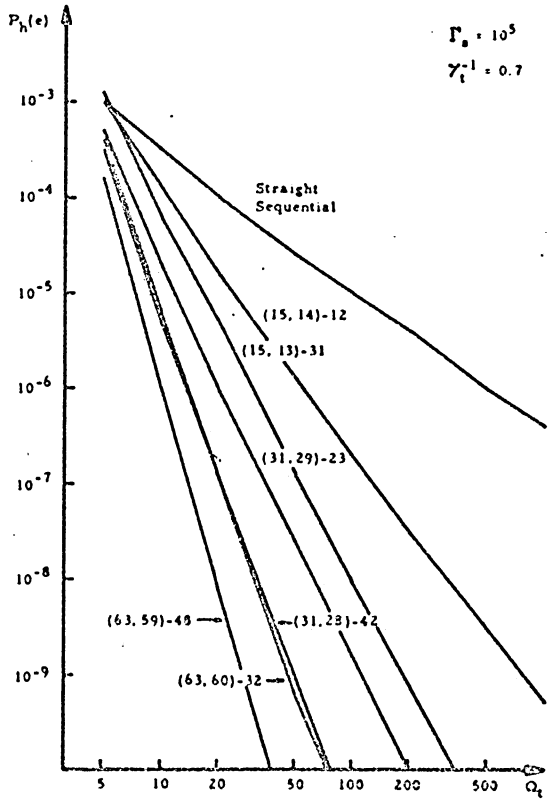


Figure 5

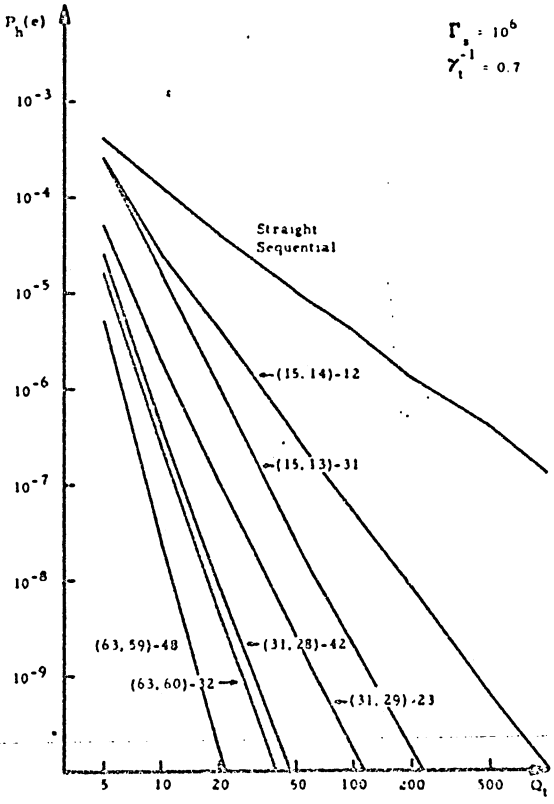


Figure 6

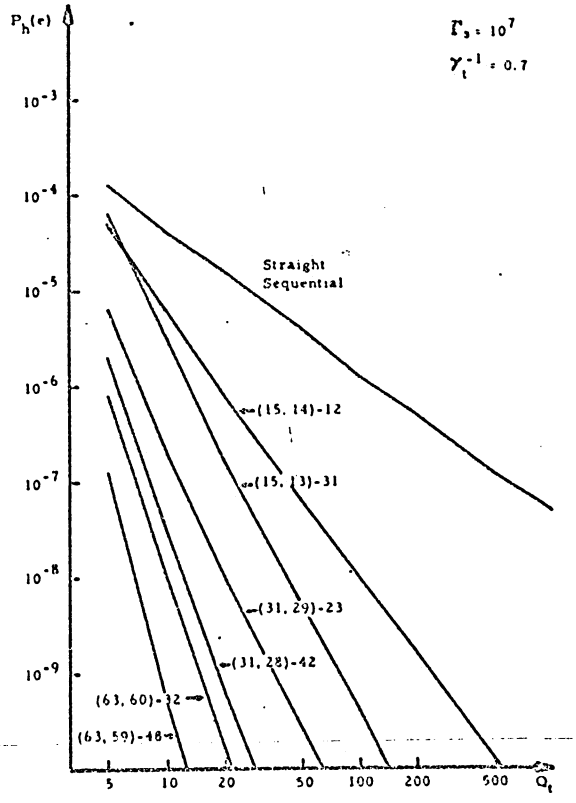


Figure 7

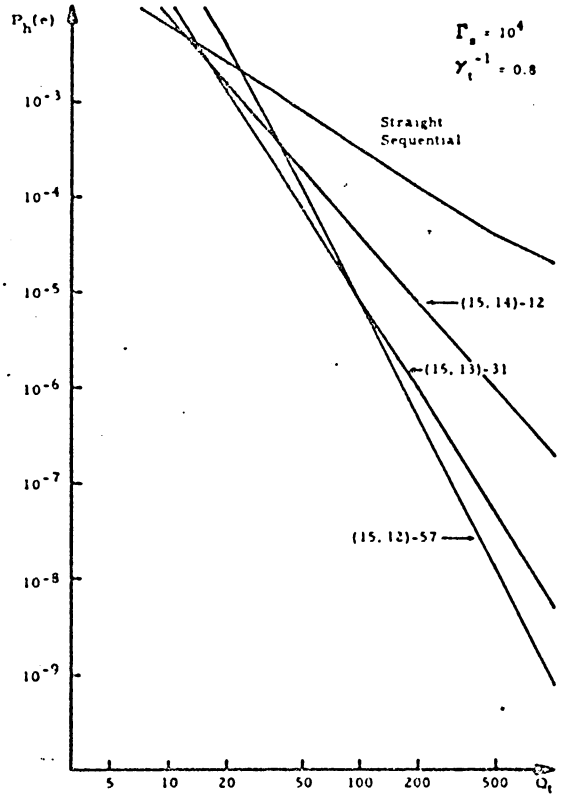


Figure 8

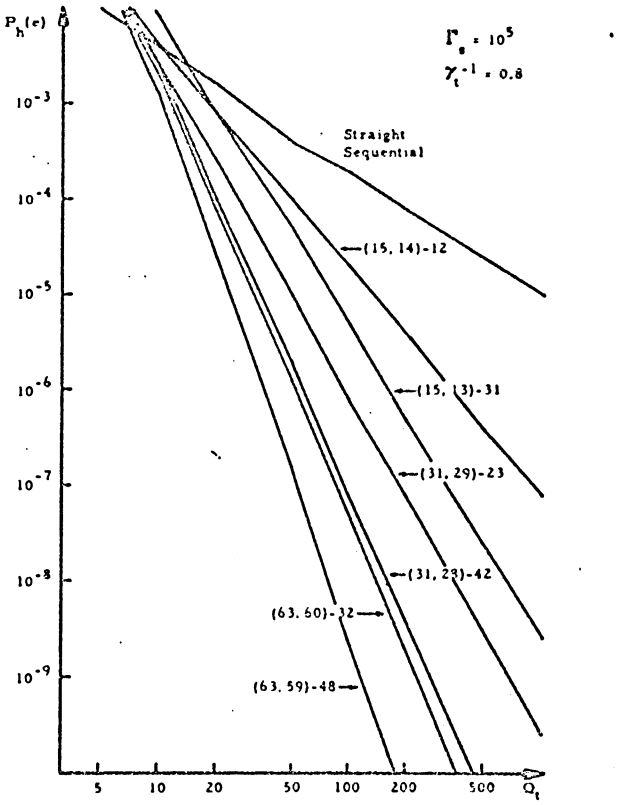


Figure 9

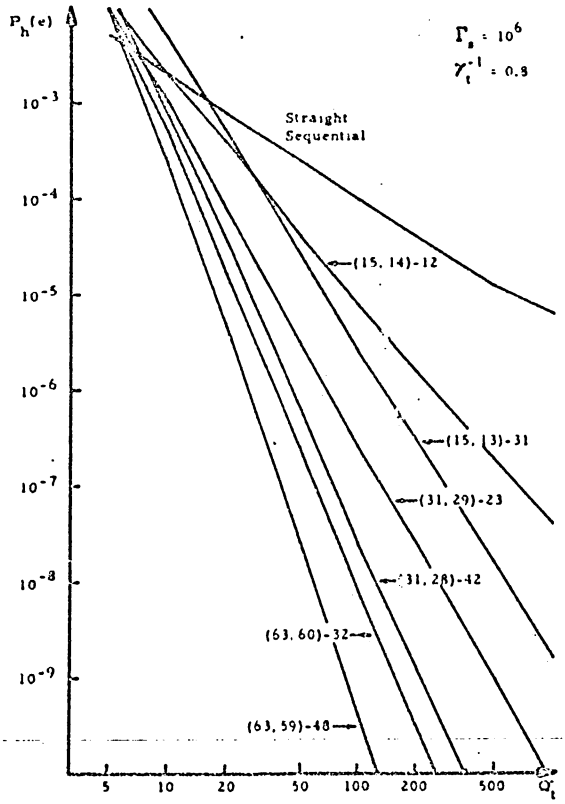


Figure 10

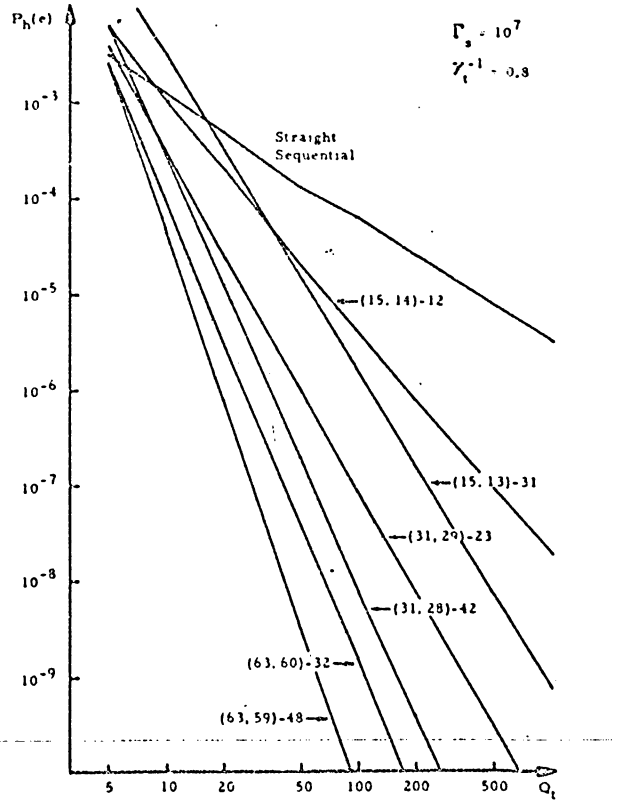


Figure 11

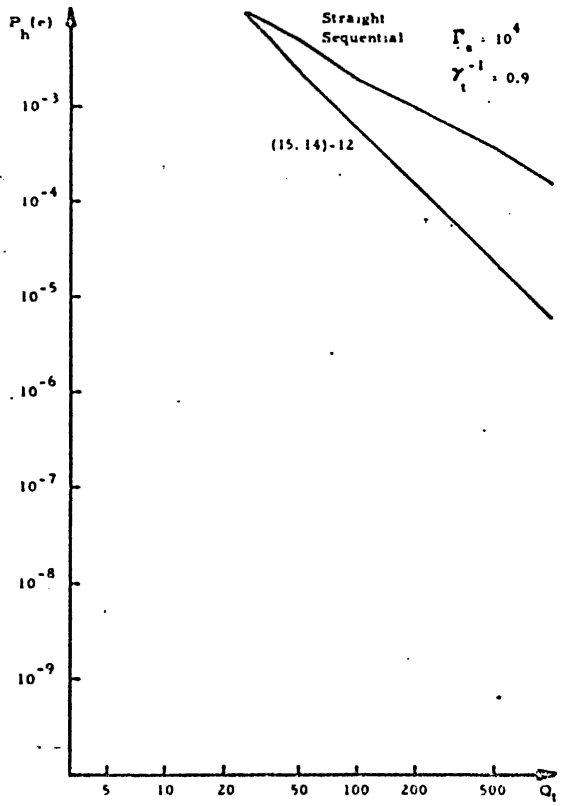


Figure 12

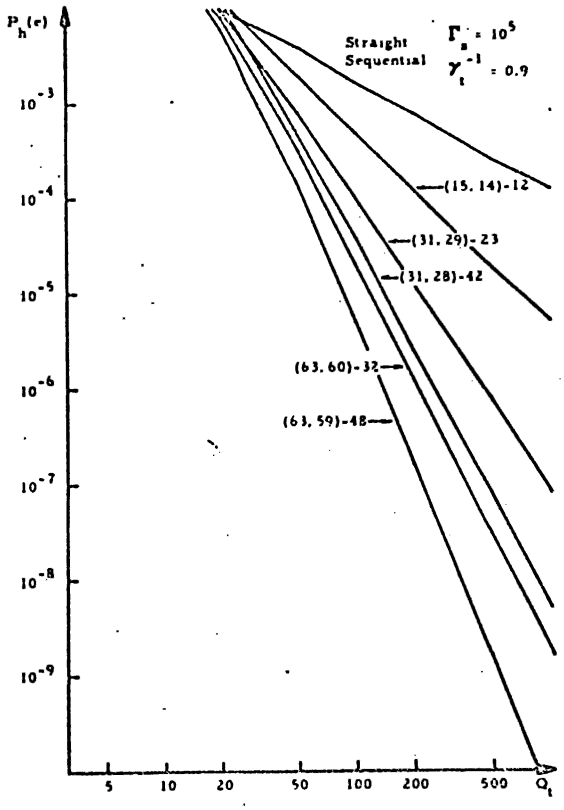


Figure 13

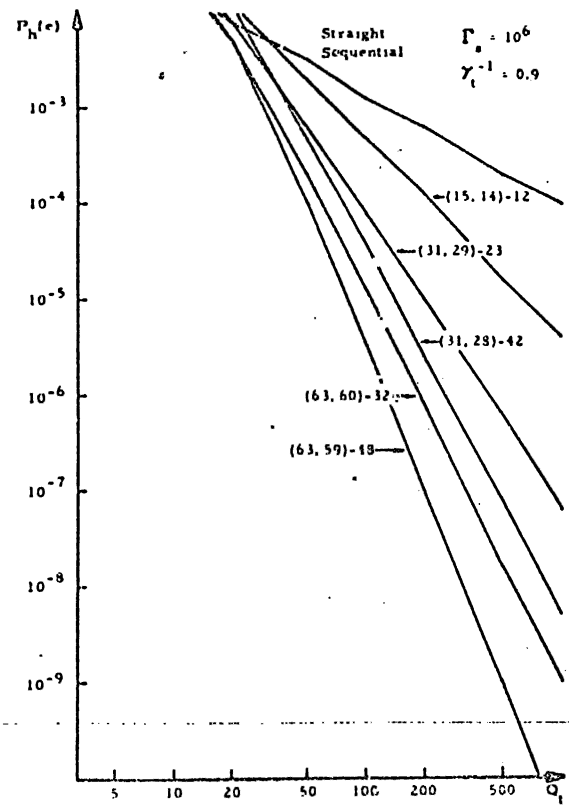


Figure 14

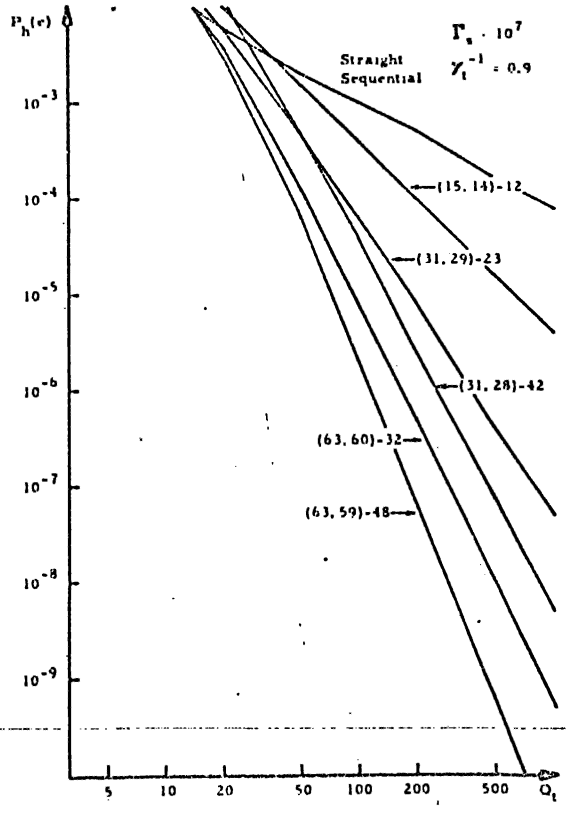


Figure 15

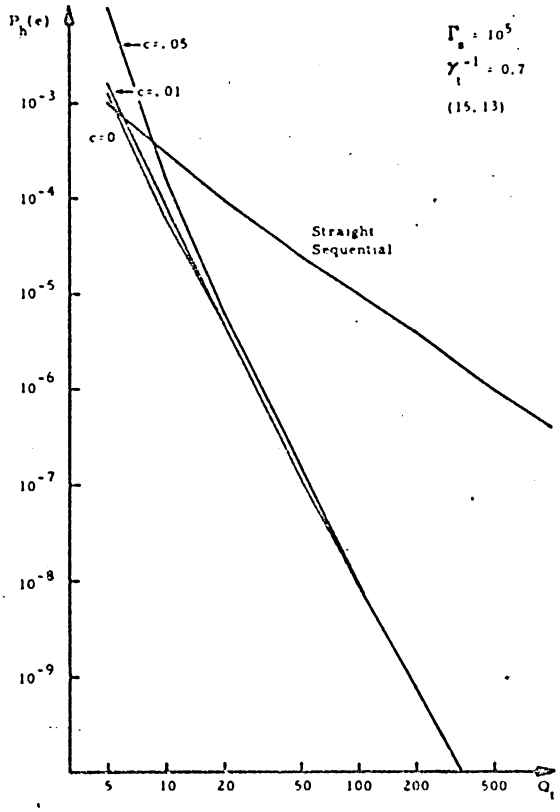


Figure 16

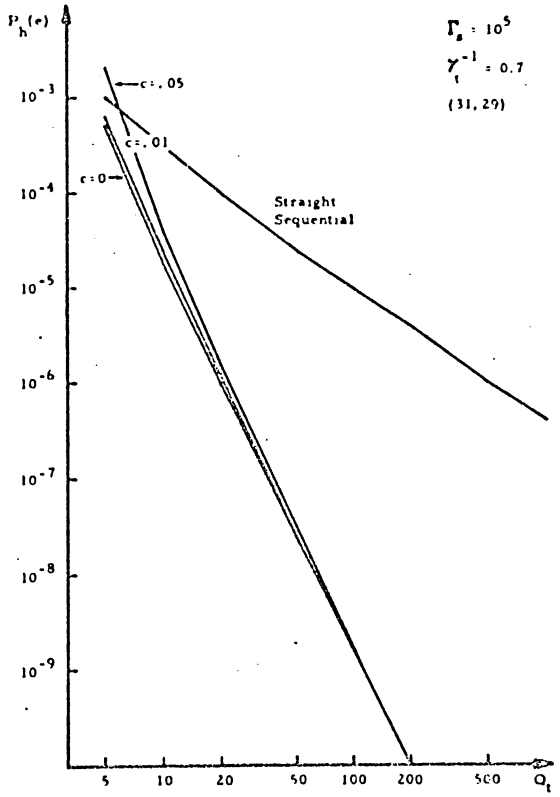


Figure 17

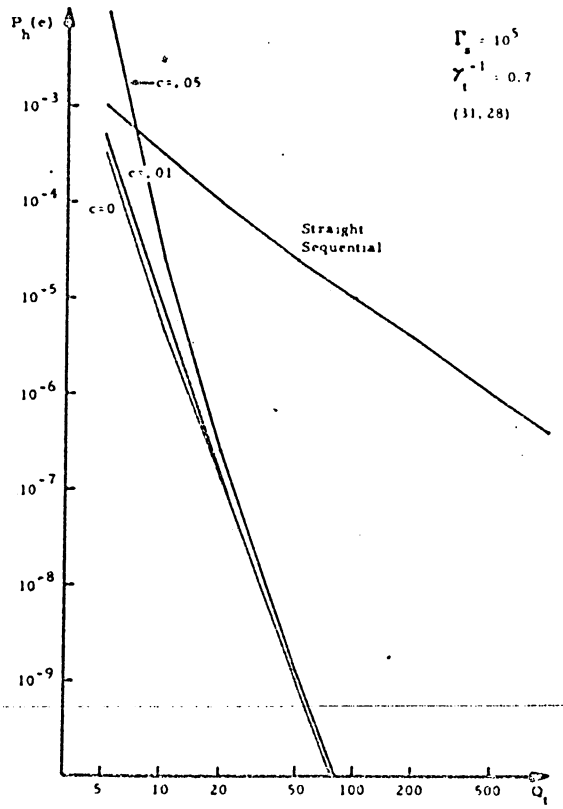


Figure 18

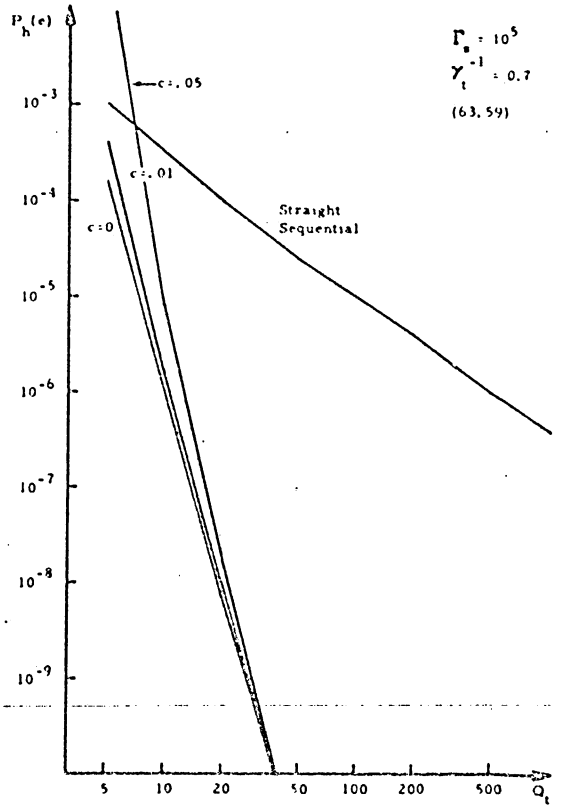


Figure 19

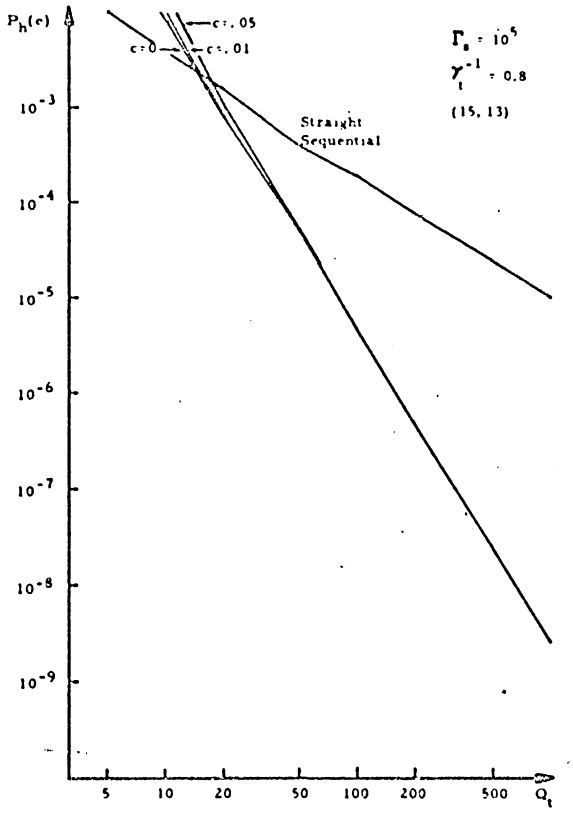


Figure 20

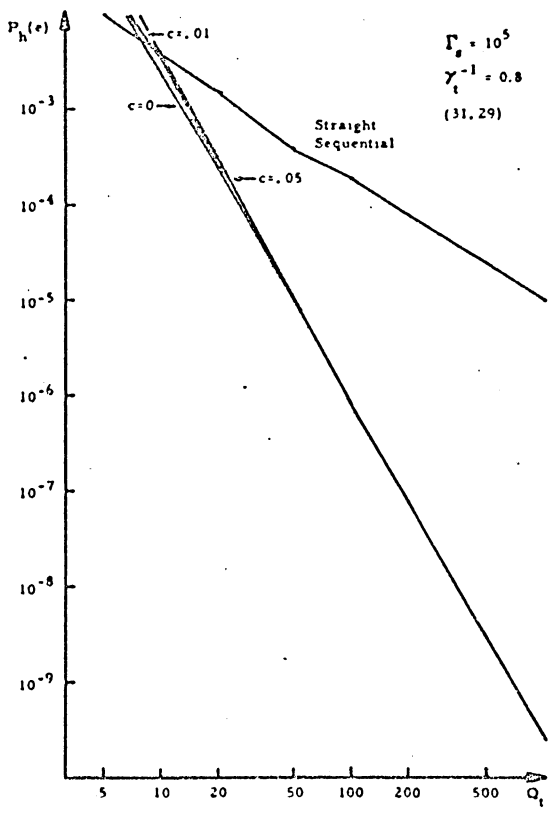


Figure 21

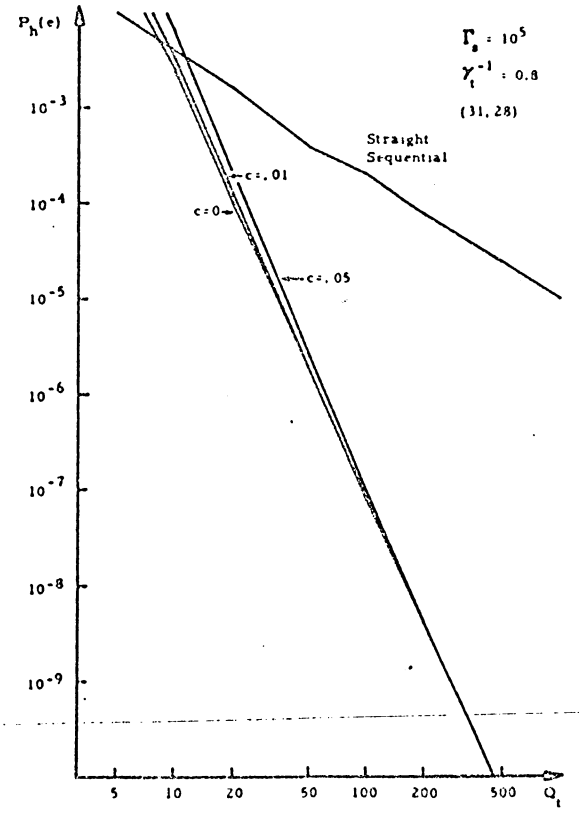


Figure 22

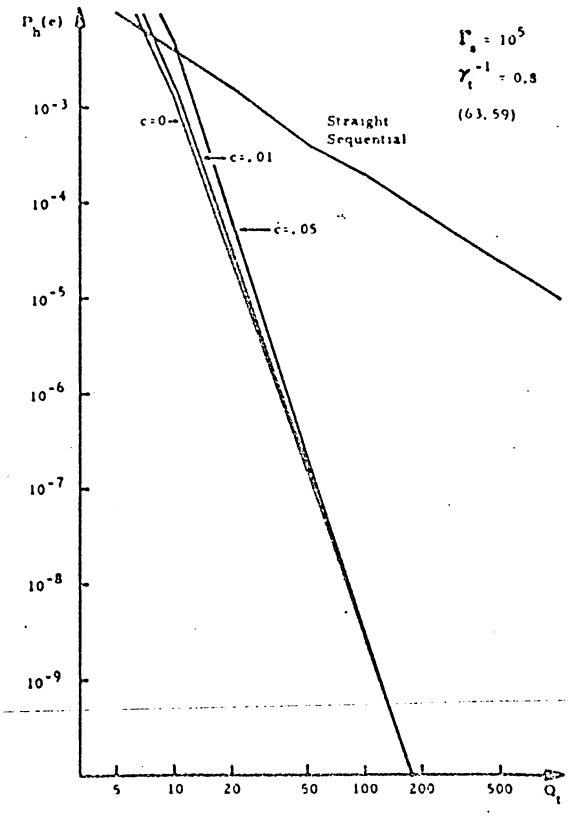


Figure 23

CONCLUSIONS

The hybrid code is capable of obtaining up to several orders of magnitude improvement in probability of error over the straight sequential code with the same overall rate and complexity of implementation. For relatively high overall rates, $R_t \approx 0.8\ R_{comp}$, error probabilities of about 10^{-6} can be obtained with a reasonable number of computations. For applications such as machine-machine communication where extreme reliability may be required, the hybrid code obtains error probabilities of about 10^{-9} with only a slight reduction in rate.

One particular advantage to this hybrid scheme is that the sequential code portion of the scheme is exactly the straight sequential code. Therefore, in many applications where sequential codes are now being utilized, the improved performance of the hybrid scheme could be realized by merely adding the Reed-Solomon section of the hybrid code. All of the hardware for the sequential encoder and decoder would be unchanged. In this type of application, the value of the constant, c, would be of little interest, and the plots with c=0 provide a good indication of the performance to be expected. In fact, the improvement actually obtained by adding the Reed-Solomon encoder-decoder pair would be greater than indicated by the plots since the length of each sequential track would be Γ_S nodes instead of Γ_S/K nodes.

Appropriate values for the constant, c, are difficult to determine. This constant should relate the complexity of one cycle of the sequential decoder to the complexity of one operation in the Reed-Solomon decoder. One operation of the Reed-Solomon decoder consists of a single register shift, whereas a sequential decoder cycle consists of several computations, comparisons, and logical decisions. It is not unreasonable then to expect that values of c on the order of .01 are probably appropriate.

APPENDIX

The Implementation of a Reed-Solomon Erasure-Correcting Encoder and Decoder

The encoding and decoding operations for the Reed-Solomon code are carried out completely by digital circuitry. Each codeword is a sequence of N symbols, or field elements, from $GF(2^m)$. The basic operations to be performed on field elements by digital circuitry include storage, addition, multiplication, and inversion.

Since any element of $GF(2^m)$ may be represented by a sequence of m binary digits, storage of a field element requires an m-bit storage register; i. e. any field element may be stored in m flip-flops. Two field elements can be added by adding their corresponding bits modulo 2. Therefore, addition of field elements requires m modulo 2 adders; i. e. the sum of two field elements is computed with m exclusive OR gates. Since addition is done modulo 2, each element is its own additive inverse, and addition and subtraction are the same operation. Multiplication of any field element in $GF(2^m)$ by the primitive element requires one shift of an m-bit shift register with feedback. The feedback connections are made through W(m)-2 modulo 2 adders, where W(m) is the Hamming weight of the irreducible polynomial.[9] Multiplication by constants other than the primitive element is also done with one shift of an m-bit shift register with feedback. This operation is slightly more complicated in that more feedback connections, i. e. more modulo 2 adders, are required. However, irreducible

polynomials can be found[10] such that for constants which are small powers of the primitive element, i. e. α^i, $i \leq m$ the number of modulo 2 adders required is approximately m. Therefore, the product of a field element and a constant is computed with m flip-flops, m exclusive OR gates, and one shift. Multiplication of two field elements requires m modulo 2 multipliers and m shifts of an m-bit accummulating register with feedback. The accummulating register is identical to the constant multiplier which multiplies by the primitive field element and therefore requires m modulo 2 adders. The product of two field elements is computed with m flip-flops, m exclusive OR gates, m AND gates, and m shifts.[11] Computing the inverse of a field element requires two m-bit shift registers with feedback and as many as $2^m-2 = N-1$ shifts, i. e. the inverse of a field element is computed with 2m flip-flops, 2m exclusive OR gates and N-1 shifts.[12] In Figure A1 symbols of devices for field element operations are diagramed, and their operations are shown in terms of two field elements, A and B. In Table A1 the amount of hardware and the number of register shifts for each of the above operations is tabulated.

The sequence of N symbols which comprise a codeword may be thought of as the sequence of N coefficients of a polynomial of degree N-1 over $GF(2^m)$. Every codeword consists of K information symbols and T check symbols. The encoding process consists of finding the T check symbols for a given set of K information symbols. Since the

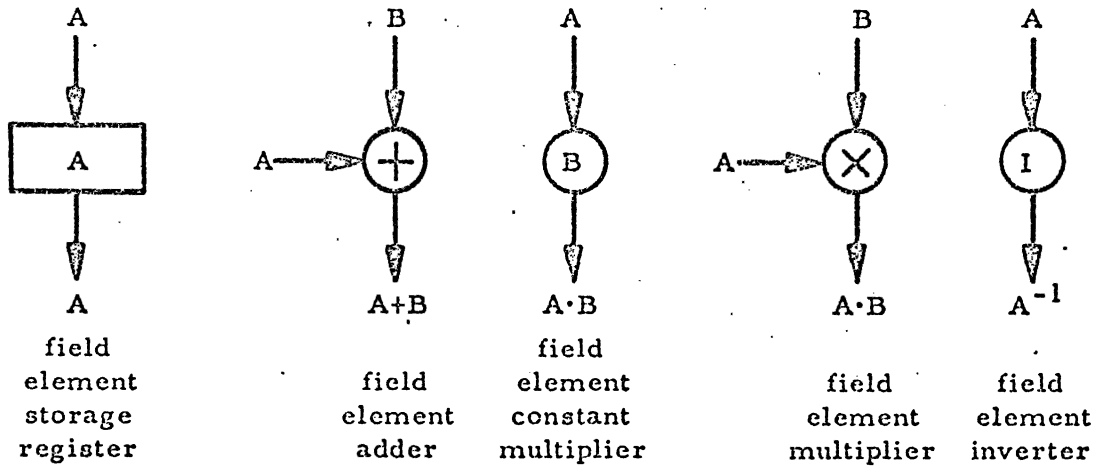


Figure A1

Field Element Operators

	Hardware			
	flip flops	exclusive OR gates	AND gates	register shifts
field element storage	m	--	--	--
field element addition	--	m	--	--
field element constant multiplication	m	m	--	1
field element multiplication	m	m	m	m
field element inversion	2m	2m	--	2^m-2

Table A1

Field Element Operations

Reed-Solomon code is cyclic, there exists a unique generator polynomial of degree N-K for the code. This generator polynomial has the property that every codeword (that is, every polynomial whose coefficients are a codeword) is divisible by the generator polynomial, and any polynomial divisible by the generator polynomial is a codeword. The encoding procedure is based on this property of the generator polynomial.

Consider a polynomial, f(X), of degree N-1 such that the K highest order coefficients are the K information symbols to be encoded, and the N-K lowest order coefficients are zero. If this polynomial were divided by the generator polynomial, g(X), the result would be

$f(X) = g(X)\, q(X) + r(X)$ where r(X) is the remainder and is a polynomial of degree $\leq$ N-K-1.

Since every polynomial divisible by the generator polynomial is a codeword, $g(X)\, q(X) = f(X) - r(X)$ is a codeword. Encoding the information symbols is accomplished by computing the remainder when f(X) is divided by g(X). Note also that f(X) contains no terms of degree less than N-K, and r(X) no terms of degree greater than N-K-1. The encoding process is, therefore, systematic in that the information symbols appear unaltered in the encoded word.

Peterson[13] has described a circuit to compute the remainder when a polynomial of degree N-1 is divided by a polynomial of degree T over the field of 2^m elements. The encoding circuit is shown in Figure A2. The T m-bit storage registers initially all contain zero.

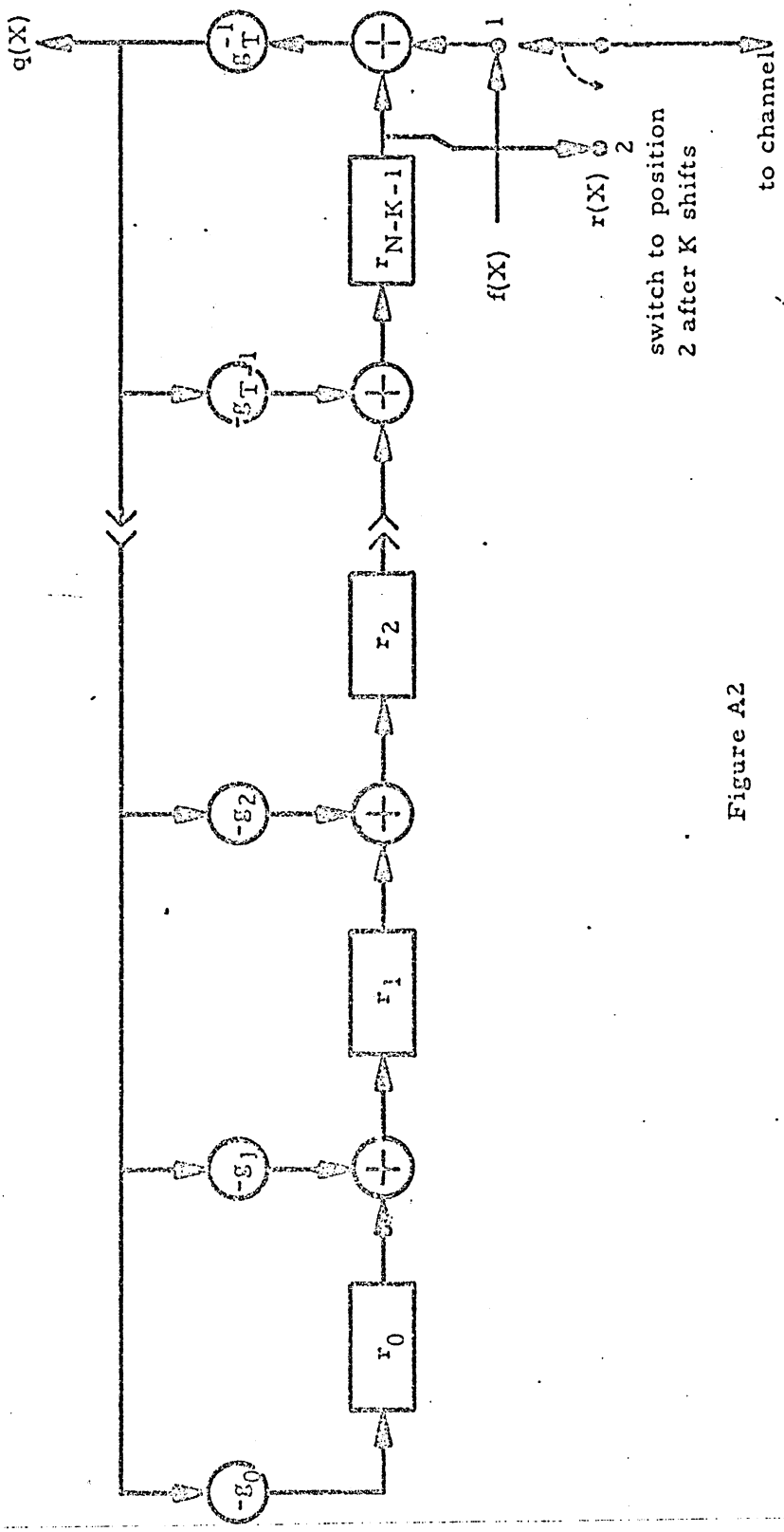


Figure A2

Reed-Solomon Encoder

When f_{N-1}, the highest order coefficient of f(X), appears at the input, $f_{N-1}\, g_T^{-1} = q_{N-T-1}$, the highest order coefficient of the quotient, appears in the feedback loop. For each coefficient, q_j, of the quotient the polynomial $q_j\, g(X)$ is subtracted from the dividend by the feedback connections. After K shifts all the information symbols have been shifted into the encoder and the channel, and the coefficients of the polynomial of check symbols, r(X), are in the T storage registers. Then the switch on the encoder is thrown to the r(X) position, and the coefficients of r(X) are shifted into the channel.

The encoding circuit requires T m-bit storage registers, T m-bit adders, and T m-bit constant multipliers. The amount of encoding hardware is proportional to mT and the number of shifts required is K.

After the K information symbols are encoded, the codeword is transmitted over a binary erasure channel. For each of the N m-bit symbols, the Reed-Solomon decoder either receives the symbol correctly, or it receives a signal that the symbol was erased. If T or fewer erasures are signaled, the Reed-Solomon decoder will recover the lost symbols.

Decoding represents very nearly the same problem as encoding, i. e. at least K symbols are known and as many as T symbols must be found such that the K+T=N symbols satisfy the defining equation for the code. Although the locations of the erasures are known, decoding is more difficult than encoding because the erasures may not be clustered as the parity check symbols are.

Let $\{x_i\}$ denote the sequence of transmitted symbols, and $\{y_i\}$ denote the sequence of received symbols. The syndrome, $\{S_j\}$, for the received sequence is defined as

$$S_{j-1} \triangleq \sum_{i=0}^{N-1} y_i \alpha^{ij} \qquad 1 \le j \le T \tag{A1}$$

where $y_i = x_i$ if the $i^{\underline{th}}$ symbol was not erased, and $y_i = 0$ if the $i^{\underline{th}}$ symbol was erased. Let the sequence $\{z_i\}$ denote the difference between the transmitted and received sequences, i.e. $z_i = x_i - y_i = x_i + y_i$. From (A1)

$$S_{j-1} = \sum_{i=0}^{N-1} (x_i + z_i)\alpha^{ij} = \sum_{i=0}^{N-1} z_i \alpha^{ij} \qquad 1 \le j \le T \tag{A2}$$

Since the number of non-zero z_i must be less than or equal to T for successful Reed-Solomon decoding, equation (A2) may be written

$$S_{j-1} = \sum_{i=0}^{\xi-1} z_{n_i} \alpha^{n_i j} \qquad 1 \le j \le \xi \qquad \xi \le T \tag{A3}$$

Define $U_i = \alpha^{n_i}$ as the $i^{\underline{th}}$ erasure locator, and $V_i = z_{n_i}$ as the $i^{\underline{th}}$ erased value. From (A3)

$$S_j = \sum_{i=0}^{\xi-1} V_i U_i^{j+1} \qquad 0 \le j \le \xi - 1 \qquad \xi \le T \tag{A4}$$

Since the occurance of an erasure is a signaled event, the $\{n_i\}$ and $\{U_i\} = \{\alpha^{n_i}\}$ are known to the decoder. The $\{S_j\}$ can be computed by the decoder from the received sequence $\{y_i\}$. To decode one must find the unknown $\{V_i\}$ from the known $\{U_i\}$ and $\{S_j\}$ according to equation (A4).

Equation (A4) may be written as the following matrix equation

$$\overline{S} = \overline{V}\left[U\right]$$

where $\bar{S}$ is the syndrome vector, $\bar{V}$ is the vector of erased values, and $[U]$ is a Van der Monde matrix. Decoding the Reed-Solomon erasure-correcting code is equivalent to inverting the matrix $[U]$ and finding $\bar{V}$ by

$$\bar{V} = \bar{S}\,[U]^{-1} .$$

In order to understand how this matrix inversion is realized by digital circuitry, we first define the polynomial

$$\sigma_p(X) \triangleq \sum_{j=0}^{\xi-1} \sigma_{pj} X^j \triangleq \prod_{\substack{i=0 \\ i\neq p}}^{\xi-1} (X+U_i) \qquad 0\le p\le \xi-1 \quad \xi\le T \tag{A5}$$

Multiplying (A4) by σ_{pj} and summing gives

$$\sum_{j=0}^{\xi-1} \sigma_{pj} S_j = \sum_{j=0}^{\xi-1} \sigma_{pj} \sum_{i=0}^{\xi-1} V_i U_i^{j+1}$$

$$= \sum_{i=0}^{\xi-1} V_i U_i \sum_{j=0}^{\xi-1} \sigma_{pj} U_i^j \qquad 0\le p\le \xi-1 \quad \xi\le T$$

From (A5) $\sigma_p(U_i) = 0 \;\forall\; i \neq p$ and thus

$$\sum_{j=0}^{\xi-1} \sigma_{pj} S_j = V_p U_p \sum_{j=0}^{\xi-1} \sigma_{pj} U_p^j \qquad 0\le p\le \xi-1 \quad \xi\le T$$

$$V_p = \left[\sum_{j=0}^{\xi-1} \sigma_{pj} U_p^{j+1} \right]^{-1} \sum_{j=0}^{\xi-1} \sigma_{pj} S_j \tag{A6}$$

The erased values can be computed by the decoder from equation (A6), but this computation requires ξ^2 σ_{pj}'s to be computed as an intermediate step. A further simplification is possible by defining the polynomial

$$\sigma(X) \triangleq \sum_{j=0}^{\xi} \sigma_j X^j \triangleq \prod_{i=0}^{\xi-1} (X+U_i) \tag{A7}$$

By expanding equations (A5) and (A7) and equating coefficients we obtain

$$\sigma_{pj} = (\sigma_{pj-1} + \sigma_j) U_p^{-1} = U_p^{-1} \sum_{i=0}^{j} \sigma_i U_p^{i-j} \tag{A8}$$

Substituting from (A8) into (A6)

$$V_p = \left[\sum_{j=0}^{\xi-1} \left(U_p^{-1} \sum_{i=0}^{j} \sigma_i U_p^{i-j} \right) U_p^{j+1} \right]^{-1} \sum_{j=0}^{\xi-1} \left(U_p^{-1} \sum_{i=0}^{j} \sigma_i U_p^{i-j} \right) S_j$$

$$V_p = \left[\sum_{j=0}^{\xi-1} \sum_{i=0}^{j} \sigma_i U_p^{i+1} \right]^{-1} \sum_{j=0}^{\xi-1} \sum_{i=0}^{j} \sigma_i U_p^{i-j} S_j \qquad \text{(A9)}$$

$$V_p \triangleq A_p^{-1} B_p$$

From equation (A9) the decoder can compute each erased symbol, V_i. The computation can be broken down into the basic field operations in terms of the known $\{U_i\}$ the $\{\sigma_i\}$ which are computed from the $\{U_i\}$, and the $\{S_j\}$ which are computed from the received sequence. Although equation (A9) appears more complicated than (A6), its implementation is less difficult since only ξ σ_i's need be computed.

A diagram of the way in which the sequence $\{U_i\}$ is determined by the decoder as the sequenced $\{y_i\}$ is received is shown in Figure A3a. The U_i register is initialized to α^{N-1}. If the first symbol from the codeword, y_{N-1}, is erased the erasure detector closes the switch and $U_0 = \alpha^{N-1}$ goes into the locator storage register. As each symbol is either erased or received, the content of the U_i register is multiplied by α^{-1}, so that when y_i is received the U_i register contains α^i. (The multiplication can be obtained with feedback connections around the U_i register thereby eliminating the need for the constant multiplier.) Only when a symbol is erased is the switch closed and the current content of the U_i register shifted into the locator storage register. The locator storage register is a T-symbol, series input, parallel

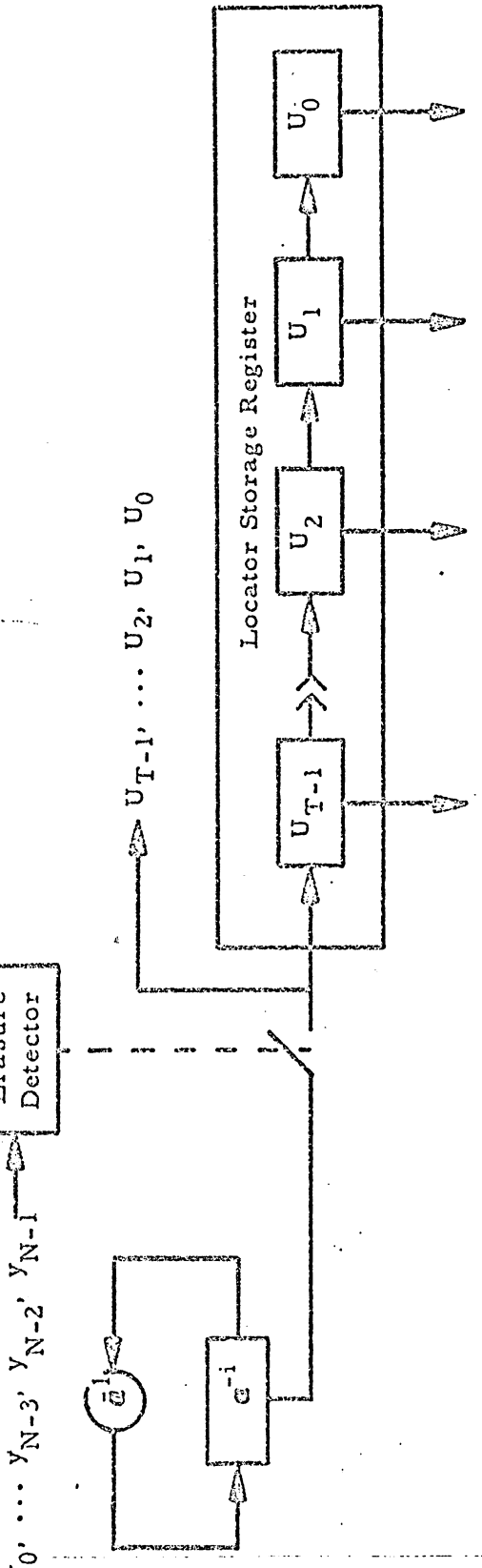


Figure A3a

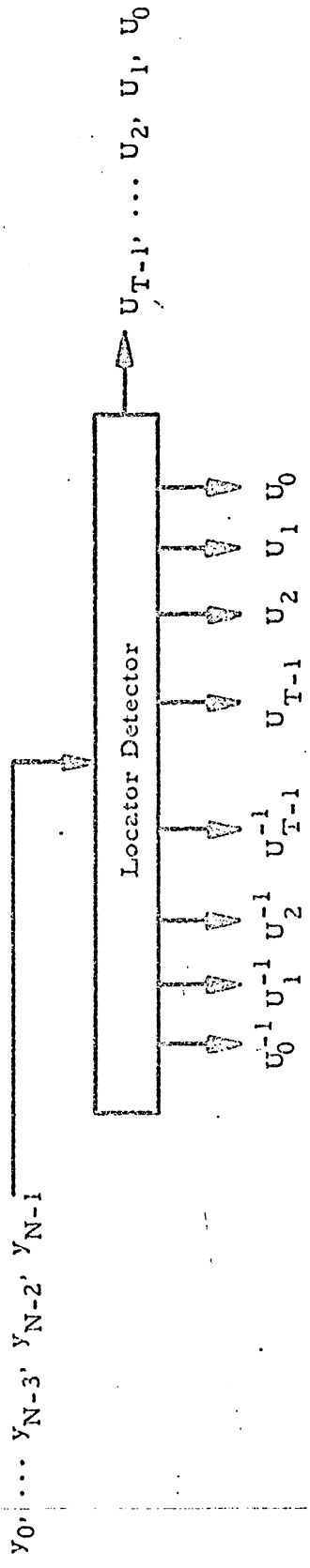


Figure A3b

Determination of the Erasure Locators

output register. The erasure locators are available to the rest of the decoder in two forms: 1) as a series string as the codeword is being received, and 2) as the parallel outputs of the locator storage register after the entire codeword is received.

Note that the inverse of each erasure locator can be determined by a similar circuit as the sequence $\{y_i\}$ is received. (The need for the sequence $\{U_i^{-1}\}$ will be shown later.) If the register is initialized to $\alpha = (\alpha^{N-1})^{-1}$, and multiplied by α with each incoming symbol, then when y_i is received the register will contain α^{-i}. The inverse of each erasure locator may be stored in a register like the locator storage register.

The overall function of the locator detector circuitry is to determine and store the value of each erasure locator and its inverse. The amount of hardware required for this circuit is proportional to mT and the number of shifts required is N. The input to the circuitry is the sequence $\{y_i\}$ in serial form and the outputs are the sequence $\{U_i\}$ in serial and parallel form and the sequence $\{U_i^{-1}\}$ in parallel form. In Figure A3b a block diagram of the locator detector with its inputs and outputs is shown.

This basic strategy will be used in the description of each functional block of the Reed-Solomon decoder: each circuit will be explained in terms of the registers, multipliers, etc. from which it may be built; then a block diagram of the circuit showing only inputs and outputs will be given. After discussing each of the functional

blocks, the entire decoder will be diagramed showing the interconnection between the blocks.

The syndrome is calculated with T m-bit storage registers, T m-bit adders, and T m-bit constant multipliers which multiply by α through α^T respectively with each shift. The circuit for computing the syndromes is shown in Figure A4a. Each of the m-bit registers is initially set to zero. The symbols are received from the channel, high order symbols first, and fed in parallel into the T m-bit adders. When y_{N-1} is received, the contents of each storage register becomes y_{N-1}. Then y_{N-2} is received and the contents of the $j^{\underline{th}}$ register is changed to $y_{N-1}\alpha^j + y_{N-2}$. This process continues until y_0 is received and the contents of the $j^{\underline{th}}$ register is

$$((y_{N-1}\alpha^j + y_{N-2})\alpha^j + \ldots + y_1)\alpha^j + y_0 = \sum_{i=0}^{N-1} y_i \alpha^{ij} = S_{j-1}.$$

The constant multipliers for this circuit also may be realized by feedback connections around the storage registers.

The amount of hardware required for computing the syndrome is proportional to mT and the number of shifts required is N. Recall that the encoded sequence is shifted out of the encoder high order symbols first. Therefore, the decoder can compute the syndrome as the sequence is being received.

The sequence $\{\sigma_i\}$ is computed by a direct expansion of the defining equation; i. e. $\sigma(X) = \prod_{i=0}^{\xi-1} (X+U_i)$. A set of T field elements can be considered as the coefficients of a polynomial, say $\sigma(X)$, of

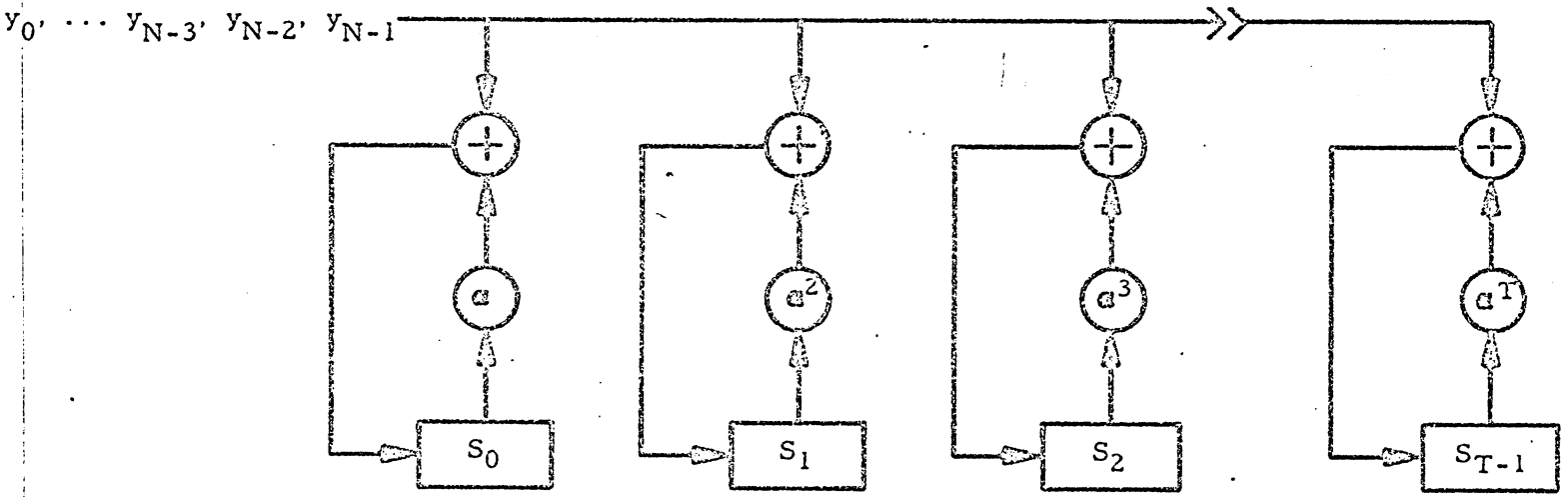


Figure A4a

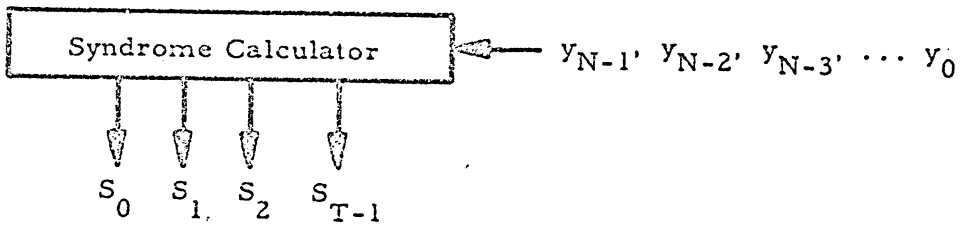


Figure A4b

Computation of the Syndrome

degree T-1 or less, and can be stored in a set of T m-bit storage registers. Multiplication of this polynomial by a field element, say U_i, is equivalent to multiplying each coefficient by U_i. Multiplication of this polynomial by X is equivalent to shifting the contents of the T registers so that the coefficient of X^i becomes the coefficient of X^{i+1}. Therefore multiplication by a factor $(X+U_i)$ may be carried out by a multiplication, and a shift, and the addition of two results.

The circuit shown in Figure A5a computes the sequence $\{\sigma_i\}$. All of the T storage registers are initialized to zero except the first which is initialized to $\alpha^0 = 1$; i. e. the storage registers contain the coefficients of the constant polynomial $\sigma(X) = 1$. When U_0 is received, the content of each register is multiplied by U_0 and the registers are right shifted. The content of the first register is then U_0 and the content of the second register is 1; i. e. the storage registers contain the coefficients of the polynomial $\sigma(X) = X+U_0$. This process continues until U_{T-1} is received, at which time the storage registers contain the coefficients of the polynomial $\sigma(X) = \prod_{i=0}^{T-1}(X+U_i)$.

The amount of hardware required for computing sequence $\{\sigma_i\}$ is proportional to mT. The storage registers must shift T times, and the field multipliers require m shifts for each shift of the storage registers i. e. the number of shifts required is mT.

The erasures are signaled, or detected, as the sequence is being received, so $\{U_i\}$ is available serially as the erasures occur. Therefore, the decoder can compute $\{\sigma_i\}$ as the sequence is being received.

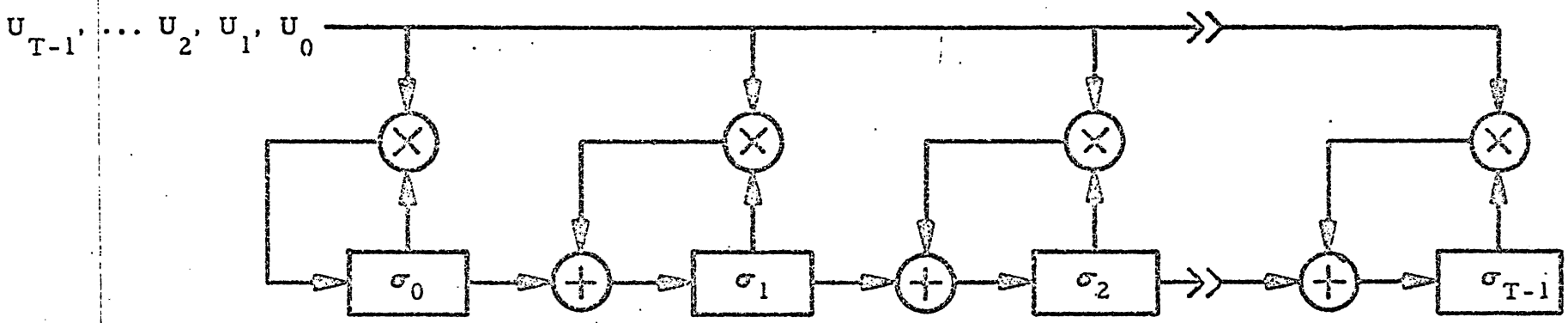


Figure A5a

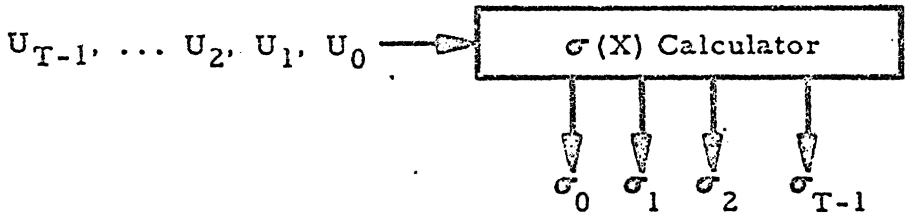


Figure A5b

Computation of $\sigma(X)$

After the N^{th} symbol is either received or declared erased, the erasure locators, $\{U_i\}$, the syndrome, $\{S_j\}$, and the $\{\sigma_i\}$ are all known, and the decoder can compute the erased values from equation (A9).

The circuit in Figure A6a computes the term $A_p^{-1} = \left[\sum_{j=0}^{T-1} \sum_{i=0}^{j} \sigma_i U_p^{i+1} \right]^{-1}$. Recalling that the summation is a modulo 2 sum, this expression for A_p can be simplified to $\sum_{i \text{ even}} \sigma_i U_p^{i+1}$ for T odd or $\sum_{i \text{ odd}} \sigma_i U_p^{i+1}$ for T even. The m-bit accumulating register A_p, is initially set to zero, and the U_p^i-register is initialized to $U_p^0 = 1$. Considering the case of T odd, when σ_0 is input to the multiplier, the contents of the accumulating register becomes $\sigma_0 U_p$. Then the U_p^i register is shifted twice, the input is switch to σ_2, and the contents of the accumulator becomes $\sigma_0 U_p + \sigma_2 U_p^3$. After the input is switched to each of the σ_i's (i even) the total in A_p is $\sum_{i \text{ even}} \sigma_i U_p^{i+1}$. A_p is then shifted into an inverter which after, at most, N-1 shifts will contain A_p^{-1}.

The amount of hardware required for the circuit in Figure A6a is proportional to m. There must be T such circuits in the decoder to compute the T A_p^{-1}'s. The number of shifts required at the input is T, and the field multipliers require m shifts for each input. In addition, the inverter requires N-1 shifts. Therefore, the amount of hardware required to compute the $\{A_p^{-1}\}$ is proportional to mT, and the number of shifts required is mT+N-1.

The circuit shown in Figure A7a computes the term $B_p = \sum_{j=0}^{T-1} \sum_{i=0}^{j} \sigma_i U_p^{i-j} S_j$. The two m-bit accumulating registers, b_p

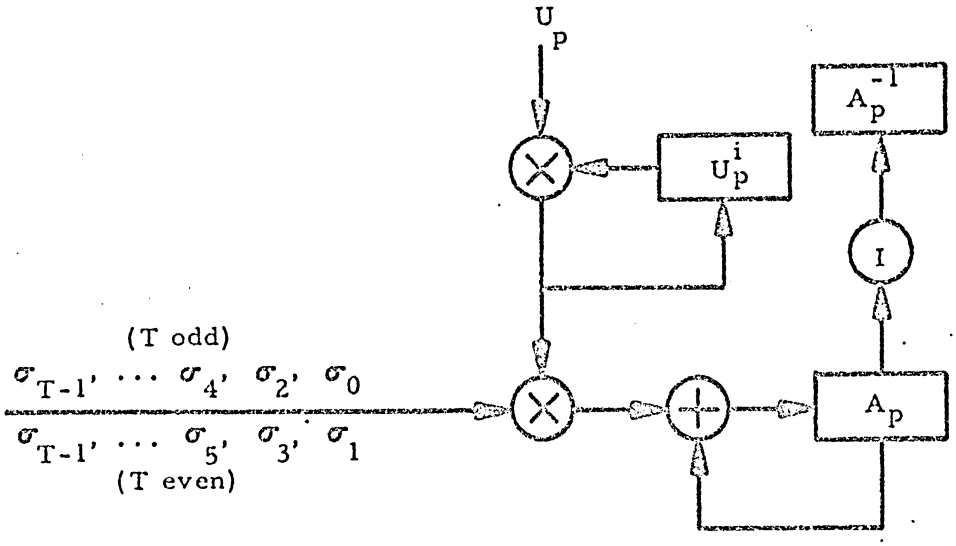


Figure A6a

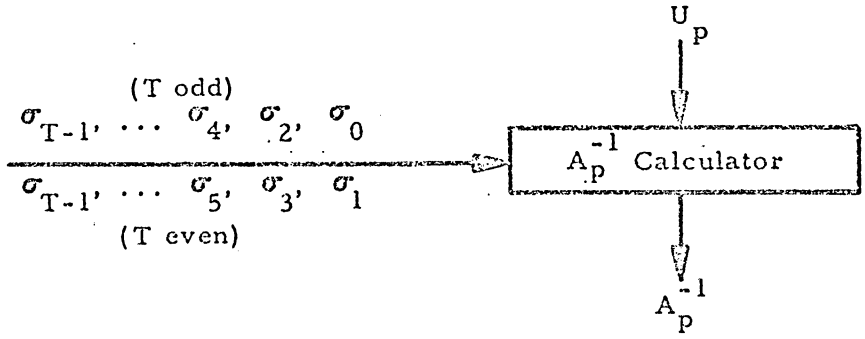


Figure A6b

Computation of A_p^{-1}

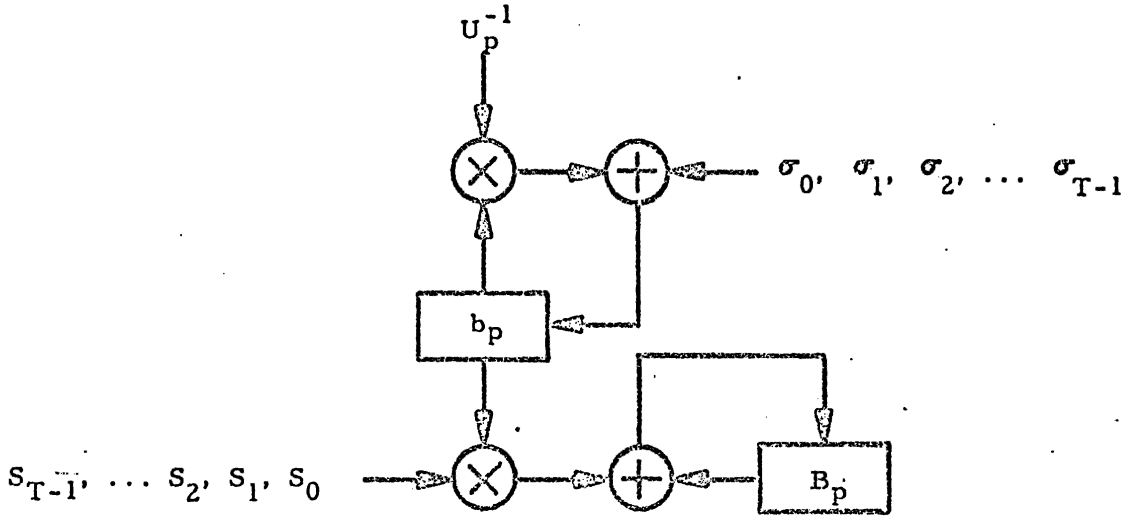


Figure A7a

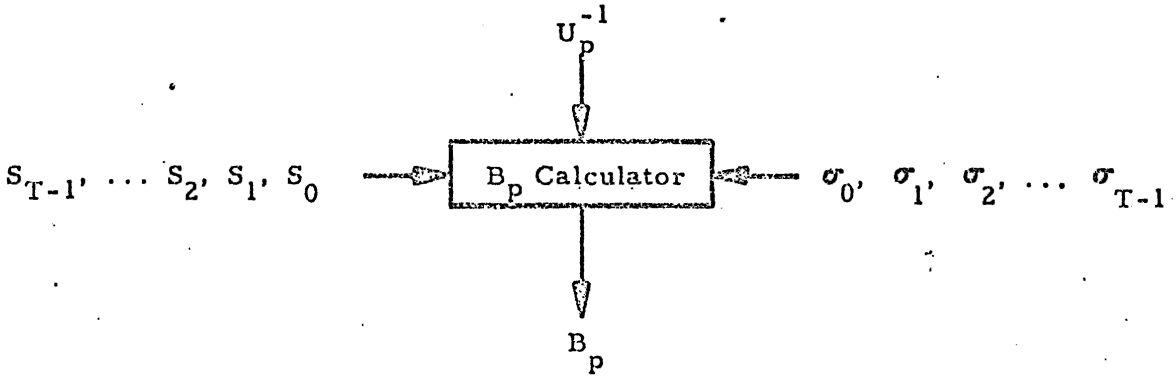


Figure A7b

Computation of B_p

and B_p, are initialized to zero. When σ_0 and S_0 are input to their respective multipliers, the contents of b_p becomes σ_0 and the contents of B_p becomes $\sigma_0 S_0$. Then the inputs are switched to σ_1 and S_1 and the contents of the accumulators become $\sigma_0 U_p^{-1} + \sigma_1$, in b_p and $(\sigma_0 U_p^{-1} + \sigma_1)S_1 + \sigma_0 S_0$ in B_p. After the inputs are switched T times, the total in B_p is $\sum_{j=0}^{T-1} \sum_{i=0}^{j} \sigma_i U_p^{i-j} S_j$.

The amount of hardware required for the circuit in Figure A7a is proportional to m. There must be T such circuits in the decoder to compute the T B_p's. The number of shifts required at the input is T, and the field multipliers require m shifts for each input. Therefore, the amount of hardware required to compute the $\{B_p\}$ is proportional to mT, and the number of shifts required is mT.

Each erased value, V_p, is computed as the product of an A_p^{-1} and a B_p. The amount of hardware required for the T field multipliers is proportional to mT and the number of shifts required is m.

The entire decoder for the Reed-Solomon erasure-correcting code is shown in Figure A8. Each block could be built exactly as indicated in Figures A3 through A7 with integrated circuit digital circuitry. The m-bit registers would be realized by m J-K flip-flops, the m-bit adders by m exclusive OR gates, and the field element multipliers by m AND gates and an m-bit accumulator with feedback. Two circular Tm-bit shift registers could serve the function of the ganged switch shown in Figure A8. The only required circuitry not explicitly shown is a clock.

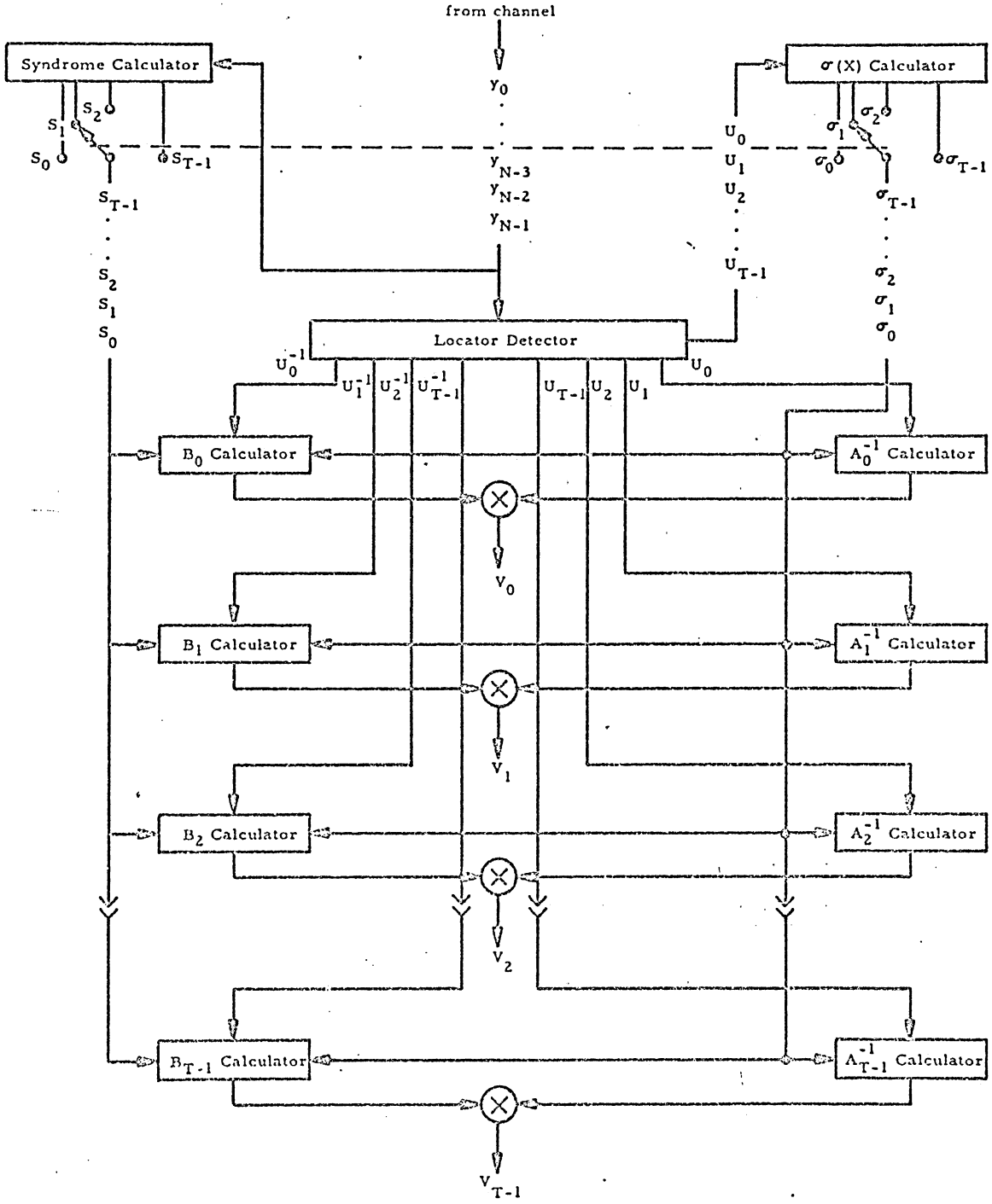


Figure A8

Reed-Solomon Decoder

To summarize, the encoding and decoding for the Reed-Solomon code is carried out completely by digital circuitry. The decoding can be segmented into six operations:

1) determination of each erasure locator and its inverse,

2) computation of the syndrome,

3) computation of the coefficients of $\sigma(X)$,

4) computation of the A_p^{-1}'s,

5) computation of the B_p's,

6) computation of the V_p's.

Operations 1), 2), and 3) above are carried out as the codewords are received and are complete for each codeword when the entire codeword is received. Operations 4) and 5) are performed simultaneously after the codeword is complete, and operation 6) is performed when 4) and 5) are complete. The hardware required for each operation is proportional to mT. A constant of proportionality for each of the operations can be easily determined by counting the number of registers, multipliers, and adders shown in Figures A3 through A7 and weighting their relative costs.

REFERENCES

1. Huband, F. and F. Jelinek, "Practical Sequential Decoding and a Simple Hybrid Scheme", Proceedings of the First Hawaii International Conference on System Sciences, p. 478, 1968.

2. Falconer, D., "A Hybrid Sequential and Algebraic Decoding Scheme", Ph. D. Thesis, M. I. T., Department of Electrical Engineering, 1967.

3. McCartney, S., "A Comparison of Four Coding Methods," M. S. Thesis, Rice University, Department of Electrical Engineering, 1968.

4. Jelinek, F. and F. Huband, "Bounds on the Performance of an Economical Sequential Decoder", Proceedings of the First Hawaii International Conference on System Sciences, p. 804, 1968.

5. McCartney, S. and F. Huband, "Numerical Evaluation of Some Sequential Decoding Parameters", Proceedings of the Second Hawaii International Conference on System Sciences, p. 249, 1969.

6. Jordan, K. L., Jr., "The Performance of Sequential Decoding in Conjunction with Efficient Modulation". IEEE Transactions on Comm. Tech., COM-14(3), June, 1966.

7. Savage, J. E., "Complexity of Decoders: I-Classes of Decoding Rules", IEEE Transactions on Information Theory, Vol. IT-15, No. 6, November 1967.

8. Savage, J. E., "Complexity of Decoders - Part II: Computational Complexity" National Science Foundation NSF GK-3302/1, National Aeronautics and Space Administration, NASA NGR 40-002-082/1 July, 1969.

9. Berlekamp, E. R., Algebraic Coding Theory, McGraw-Hill, New York, 1968 p. 46.

10. Peterson, W. W., Error Correcting Codes, The M. I. T. Press, 1961, p. 251.

11. Gallager, R., Information Theory and Reliable Communication, John Wiley and Sons, 1968, p. 234.

12. Peterson, W. W., p. 116.

13. Peterson, W. W., p. 150.