National Library of Canada

Cataloguing Branch
Canadian Theses Division

Ottawa, Canada
K1A 0N4

Bibliothèque nationale du Canada

Direction du catalogage
Division des thèses canadiennes

NOTICE

The quality of this microfiche is heavily dependent upon the quality of the original thesis submitted for microfilming. Every effort has been made to ensure the highest quality of reproduction possible.

If pages are missing, contact the university which granted the degree.

Some pages may have indistinct print especially if the original pages were typed with a poor typewriter ribbon or if the university sent us a poor photocopy.

Previously copyrighted materials (journal articles, published tests, etc.) are not filmed.

Reproduction in full or in part of this film is governed by the Canadian Copyright Act, R.S.C. 1970, c. C-30. Please read the authorization forms which accompany this thesis.

THIS DISSERTATION
HAS BEEN MICROFILMED
EXACTLY AS RECEIVED

AVIS

La qualité de cette microfiche dépend grandement de la qualité de la thèse soumise au microfilmage. Nous avons tout fait pour assurer une qualité supérieure de reproduction.

S'il manque des pages, veuillez communiquer avec l'université qui a conféré le grade.

La qualité d'impression de certaines pages peut laisser à désirer, surtout si les pages originales ont été dactylographiées à l'aide d'un ruban usé ou si l'université nous a fait parvenir une photocopie de mauvaise qualité.

Les documents qui font déjà l'objet d'un droit d'auteur (articles de revue, examens publiés, etc.) ne sont pas microfilmés.

La reproduction, même partielle, de ce microfilm est soumise à la Loi canadienne sur le droit d'auteur, SRC 1970, c. C-30. Veuillez prendre connaissance des formules d'autorisation qui accompagnent cette thèse.

LA THÈSE A ÉTÉ
MICROFILMÉE TELLE QUE
NOUS L'AVONS REÇUE

UNIVERSITÉ D'OTTAWA
UNIVERSITY OF OTTAWA

ON THE INTERLEAVING OF ERROR CORRECTING CODES

by

R. B. CURRIE

Submitted to the School of Graduate Studies in partial fulfillment of the requirements for the degree

of

Master of Applied Science

Department of Electrical Engineering
Faculty of Sciences and Engineering
University of Ottawa
Ottawa, Ontario

July 1975

© R.B. Currie, Ottawa, Canada, 1975.

Abstract

The problem of errors in a digital communication system is discussed, and four models of a communication channel are presented. Of these, the Gilbert model is the most suitable for modelling a typical communication channel. A study of available error correcting codes is made, with emphasis on block codes. None of these codes seems suitable for combatting errors on the compound channel. Interleaved codes are then introduced, and it is demonstrated that codes formed by the interleaving of random error correcting codes with burst error correcting codes can guarantee the correction of more error patterns than comparable SBEC codes, product codes, concatenated codes, and Reed-Solomon codes in certain cases. Biased interleaving is introduced as a method of giving added protection to certain critical sections of a message. A brief discussion on the decoding of interleaved codes is included.

Acknowledgement

The author wishes to express his thanks to professor S.G.S. Shiva of the University of Ottawa, whose constant help and criticism was invaluable in the preparation of this thesis.

Thanks are also due to all the graduate students of the department of Electrical Engineering for providing an atmosphere suitable for study and research. Particular thanks are due to S.E. Guevara for many hours of assistance during tedious calculations.

Special thanks are due to Mrs. M. Benoit who spent hours typing the manuscript, and to my wife for being patient with me during the last six years.

The financial support for this research was provided in part by the Defense Research Board, and the National Research Council of Canada, under respective grant numbers 9931-31 and A 3371.

I wish also to thank the Canadian Armed forces for supplying the major financial support for my university education.

Table of Contents

	page
1: Digital Communications	1
1.1: Outline of the Thesis	3
2: Errors and Channel Models	4
2.1: Channel Characterization by Error Type	4
2.2: Channel Models	4
2.2.1: The Additive White Gaussian Noise (AWGN) Channel Model	5
2.2.2: The Gaussian Distributed Burst Model	6
2.2.3: The Uniformly Distributed Burst Model	8
2.2.4: The Gilbert Model	12
2.3: Error Control Through Coding--	13
3: Types of Codes	16
3.1: Block Codes	16
3.1.1: Random Error Correcting Block Codes	17
3.1.2: Burst Error Correcting Block Codes	20
3.1.3: Block Codes for the Correction of the Simultaneous Occurrence of Random Errors and Burst Errors	21
3.2: Convolutional Codes	24
4: Interleaving and Interleaved Codes	30
4.1: Interleaving of Random Error Correcting Codes	31
4.2: Interleaving of Burst Error Correcting Codes	40
5: Interleaving of Random Error Correcting Codes with Burst Error Correcting Codes	44
5.1: Comparison of Interleaved Random and Burst Error Correcting Codes with SBEC codes	45
5.2: Comparison of Interleaved Random and Burst Error Correcting Codes with Product Codes, Concatenated Codes, and Reed-Solomon Codes	65

Table of Contents (continued)

	page
5.2.1: Product Codes	65
5.2.2: Concatenated Codes	70
5.2.3: Reed-Solomon Codes	75
5.3: Biased Interleaving	79
6: Decoding of Interleaved Codes	87
6.1: Decoding of Interleaved Random Error Correcting Codes	87
6.2: Decoding of Interleaved Burst Error Correcting Codes	88
6.3: Decoding of Codes Formed by Interleaving Random Error Correcting Codes with SBEC codes	90
7: Concluding Remarks	92
References	94

List of Illustrations

Figure		page
1.1	A digital communication system	1
2.1	The AWGN channel model	5
2.2	A binary symmetric channel	5
2.3	Transition matrix for the gaussian distributed burst model	6
2.4	Model for uniformly distributed bursts	8
2.5	The Rayleigh distribution	9
2.6	A three-state Gilbert model	12
2.7	Pe vs Eb/No for no coding and for coding at channel capacity	14
3.1	Communication channel using a concatenated code	23
3.2	A general (mn_o, mk_o) convolutional encoder	25
3.3	A general decoder for a systematic (mn_o, mk_o) convolutional code	25
6.1	Decoder for a $(\mu n, \mu k)$ interleaved code	87
6.2	Timing and gate-enabling circuit for block interleaved burst error correcting codes	89

List of Tables

Table		page
4.1	Random error correcting codes interleaved to degree $\mu = 3$	36
4.2	Interleaving of burst error correcting codes (for b approximately 3000)	42
5.1	Interleaved codes formed from one burst and one random error correcting code	48
5.2	Interleaved codes formed from an SBEC code and a code that corrects single random or double-adjacent errors	51
5.3	SBEC codes interleaved with two codes that correct single random or double-adjacent errors	55
5.4	Interleaved codes formed from optimal SBEC codes and multiple random error correcting codes	64

Chapter 1. DIGITAL COMMUNICATIONS

Information can be transmitted in either of two forms, analog or digital. Speech and music are examples of analog information; teletype and telegraph are examples of digital information.

By far, the greatest part of communications in the world today is from machine to machine, and the information communicated is usually in digital form [1].

A large part of this communication is numerical information, and is already in digital form; hence no transformation is required.

Analog information is readily converted to digital form by the use of such techniques as pulse amplitude modulation (PAM), with pulse code modulation (PCM), delta modulation, pulse width (duration) modulation (PWM, PDM), or pulse position modulation (PPM). PCM is by far the most widely used technique to convert analog signals to digital signals.

A digital communication system can be shown as:

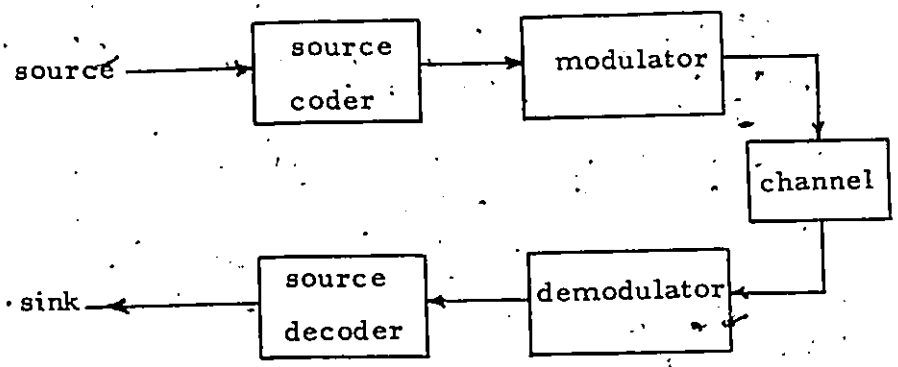

Figure 1.1 A digital communication system

The coder converts the source information to the required digital form and the decoder converts the digital information from the receiver into the form required at the output. Once the information has been converted to the digital form, the original source is unimportant in that all digital information is treated by the modulator-channel-demodulator in exactly the same fashion. It does not matter whether the original signal was a human voice, as in a telephone, a scene, as on a TV studio stage, a printed page, as in a facsimile machine, or a stream of bits from a computer memory; the channel treats them all the same. This is one great advantage of a digital communication scheme.

Other advantages of digital communications are increased accuracy, effective noise minimization, and better processing, and storage of information. Resolution can always be improved by increasing the number of bits used [2].

Analog signals are degraded further with each operation performed on them, while digital signals can be regenerated so degradation is not cumulative.

When an analog signal is distorted by noise the signal can never be recovered exactly, since the analog signal takes on an infinite number of values. On the other hand digital signals take on only a finite number of values (2 in the binary case) and hence can be recovered exactly as long as the noise does not exceed a given value.

Another compelling reason for the emergence of digital transmission systems is the rapid growth of large scale integration. Chips with digital computing power are relatively inexpensive and hence highly attractive for the design engineer.

The great number of digital systems in use today and the expected future proliferation of such systems have brought increased importance to the study of the types of errors that can affect digitally transmitted information.

1.1 Outline of the Thesis

In chapter 2 a communications channel will be categorized according to the type of error that is introduced and several simple models of a communications channel will be presented. Chapter 3 will deal with the type of error correcting codes that are available. Chapter 4 will present the principle of interleaving which will be extended in chapter 5 to cover the interleaving of different codes. Chapter 6 will deal with the decoding of interleaved codes.

Chapter 2 ERRORS AND CHANNEL MODELS

2.1 Channel Characterization by Error Type

With respect to the errors caused in a digital communication system, a communications channel can be categorized in two ways, memoryless, or with memory [3] . The category that a particular channel falls into is dependent on the type of noise or perturbation that a signal is exposed to when passed through the channel.

The noise on a communications channel can be of two basic types. These are random noise and impulse noise. If only random noise is present the channel is said to be memoryless, or to be a random channel. In such a channel, the probability of a bit being in error is the same for every bit, and does not depend on what has occurred with any previous bits. If, in place of or in addition to random noise, impulse noise is present on the channel, then the channel is said to have memory. Impulse noise is usually the result of switching transients, dropouts, lightning strokes or fading. These can affect many bits in succession. Hence the probability of bits being in error after an initial error is increased, that is, the channel has memory.

We call such a channel a bursty channel and define a burst as follows. A burst error of length b is defined to be a sequence of b error symbols, the first and last of which are non-zero [4] .

2.2 Channel Models

In order to analyze the effect of noise on signal, it is extremely useful to be able to give a mathematical model of the channel. It is relatively simple to model a random or memoryless channel and considerably more complex to model a channel with memory or a bursty channel.

In this section are presented four different models for various channels together with examples of calculations of the probability of certain error patterns. The first model is for a memoryless channel, the next two for channels where burst errors occur, and the last is for a channel where both burst and random errors occur.

2.2.1 The Additive White Gaussian Noise (AWGN) Channel Model

If the only noise that affects the signal is AWGN, we call the channel an AWGN channel and can model it in a simple fashion.

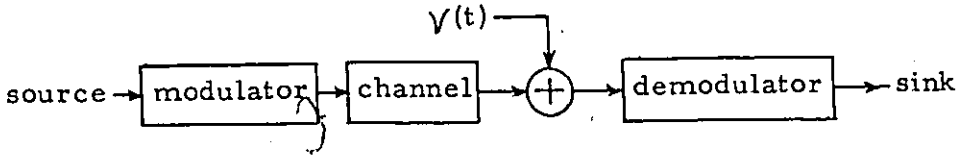

Figure 2.1 The AWGN channel model.

The noise, $\gamma(t)$, has a probability density function,

$$p(\gamma) = (1/\sqrt{2\pi\sigma^2})e^{(\gamma^2/2\sigma^2)}$$

where σ is the standard deviation of the noise. If the channel symbols are chosen from an alphabet of size L, then the effect of the noise on the signal can be completely described by an LxL transition matrix whose ij'th entry is the conditional probability, p(j/i), that the j'th symbol is received, given that the i'th symbol was transmitted [5].

The most common case is when L=2, the binary case, with p(0/1)= p(1/0). Such a channel is called a binary symmetric channel, (BSC), and can be completely characterized by the single parameter, p, the bit error probability, or the cross-over probability of the channel.

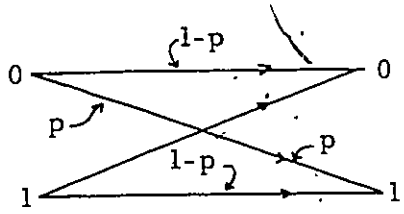

Figure 2.2 A binary symmetric channel (BSC)

Example 2.1 Error probability on the BSC.

Given a BSC, the parameter p, and a block of n bits, the probability of exactly t errors occurring is given by

$$p(t) = \binom{n}{t} p^t (1-p)^{n-t}$$

Let $p = .1$, $n=15$

Then, the probability of no errors occurring is

$$p(0) = (1-p)^{15} = .9^{15} = 0.205891$$

The probability of one error occurring is

$$p(1) = \binom{15}{1}(.1)(.9)^{14} = 0.343151$$

The probability of two errors occurring is

$$p(2) = \binom{15}{2}(.1)^2(.9)^{13} = 0.266896$$

The probability of three errors occurring is

$$p(3) = \binom{15}{3}(.1)^3(.9)^{12} = 0.128505$$

The probability of three or less errors occurring is

$$P = p(3) + p(2) + p(1) + p(0) = 0.944443$$

2.2.2 The Gaussian Distributed Burst Model

On certain channels errors may be of the burst type with the probability of an initial error, that is, the first error in a burst pattern, having a gaussian distribution. Such a channel can be accurately modelled through the appropriate choice of three parameters. These are $p(x_i)$, which is the probability of the i^{th} bit being the initial error in a burst, p which is the conditional probability that the $(j+1)^{th}$ bit is incorrect given that the j^{th} bit was correct and q which is the probability of the $(j+1)^{th}$ bit being correct, given that the j^{th} was incorrect. We form a transition matrix of conditional probilities as follows:

j \ j+1	R	W
R	1-p	p
W	q	1-q

Figure 2.3 Transition matrix for the gaussian distribtued burst model. R and W stand for right and wrong. Thus, for example, 1-p is the probability of the (j+1)st bit being right, given that the jth bit was right.

With this matrix and the value of $p(x_i)$, the probability of a burst of any given length occurring in a block of n bits can be calculated. It must be noted that the probability of the first bit being in error in a block is affected by the probability of a burst error occurring at the end of the previous block and overlapping. However, this change in probability can be made arbitrarily small by adding a tail of zeros to each block (with some lowering effect on the rate of transmission of information). In the following example it is assumed that overlapping errors have not occurred or have been compensated for.

Example 2.2 Error probability on the Gaussian Distributed Burst channel.

Given $p(x_i)$, p and q and a block of n bits, then the probability of no error, $p(b=0)$, is given by

$$p(b=0)=(1-p(x_i))^n$$

The probability of a burst of length one, $p(b=1)$, is given by

$$p(b=1)=\sum_{i=0}^{n-2}((1-p(x_i))^i p(x_i) q(1-p)^{n-2-i})+(1-p(x_i))^{n-1}p(x_i)$$

The probability of a burst of length two, $p(b=2)$, is given by

$$p(b=2)=\sum_{i=0}^{n-3}((1-p(x_i))^i p(x_i)(1-q)q(1-p)^{n-3-i}+(1-p(x_i))^{n-2}p(x_i)(1-q)$$

The probability of a solid burst of length three, that is, a burst of three consecutive incorrect digits, $p(b_s=3)$, is given by

$$p(b_s=3)=\sum_{i\ 0}^{n-4}((1-p(x_i))^i p(x_i)(1-q)^2 q(1-p)^{n-4-i}+(1-p(x_i))^{n-3}p(x_i)(1-q)^2$$

The probability of a split burst of length three is given by

$$p(b_{sp}=3)=\sum_{i\ 0}^{n-4}((1-p(x_i))^i p(x_i)qpq(1-p)^{n-4-i}+(1-p(x_i))^{n-3}p(x_i)qp$$

For $p(x_i)=0.05$, $p=0.01$, $q=0.5$, and $n=15$, we have
$p(b=0)=0.463291$, $p(b=1)=0.262533$, $p(b=2)=0.118673$,
$p(b_s=3)=0.060821$, $p(b_{sp}=3)=0.001216$, and finally
$p(b\leq 3)=0.906534$, which is the sum of all previous probabilities.

In this model, if the value of p is increased, the probability of longer bursts is increased, and decreasing the value of q will tend to make solid bursts more probable. Relatively high values of p and q will tend to produce long bursts that are not solid.

This model is suitable for channels where impulse noise occurs with a gaussian distribution and also for channels where random noise occurs but the errors tend to propagate.

2.2.3 The Uniformly Distributed Burst Model

In the previous model it was assumed that the start points of the bursts had a gaussian distribution and, since the probability of a burst starting was different from the probability of it continuing, it was found that bursts of the same length had different probabilities for different start points; that is, for example, a burst of length three starting in position two had a different probability from a burst of length three starting in position four or five.

A more realistic model assumes that all bursts of the same length and pattern have the same probability. Such a model requires more parameters and is more complex but is still reasonably tractable mathematically. The model is shown in figure 2.4.

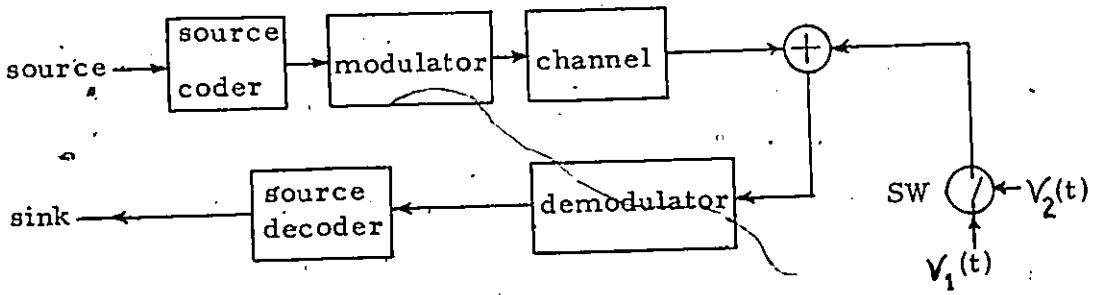

Figure 2.4 Model for Uniformly Distributed Bursts.

In figure 2.4, $\gamma_1(t)$ is a source of gaussian noise and $\gamma_2(t)$ is a source with gaussian distribution that controls the closing of the switch SW. That is, the switch will close whenever the amplitude of $\gamma_2(t)$ exceeds a predetermined value. We assign appropriate values to p(1), ..., p(i), ..., where p(i) represents the probability that, once the switch closes, it remains closed for i bits. From the distribution of $\gamma_1(t)$ and $\gamma_2(t)$, we have the probabilities p_1 and p_2, respectively, where p_1 is the probability of a bit error given that the switch is closed, and p_2 is the probability of the switch closing during any block of n bits.

With this model, the frequency of burst occurrence is varied by varying p_2. The probability of bursts of various lengths is determined by the values assigned to the p(i), and the probability of particular burst patterns is determined by the value of p_1. A low value of p_2 and a high value of p_1 will cause infrequent dense bursts, while high values of p_1 and p_2 will cause frequent dense bursts. A low value of p_1 will cause bursts with sparsely scattered ones.

A reasonable way to assign values to the p(i) is to use the Rayleigh distribution, which is given by [6] :

$$p(x) = (x/\sigma^2)e^{(-x^2/2\sigma^2)}U(x)$$

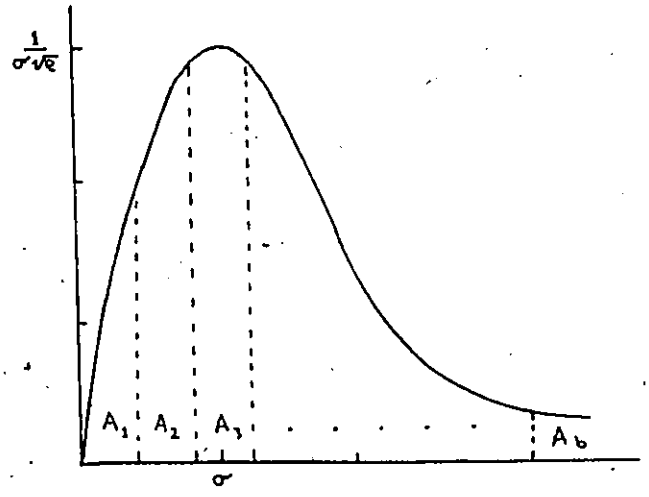

Figure 2.5 The Rayleigh Distribution.

If we assume that only burst errors of length b or less will occur and then subdivide the base-line in figure 2.5 into b sections, and then let the b areas, A_1, A_2, A_3,..., A_b represent the probability of bursts of length corresponding to the subindices, the result will make the model quite flexible and realistic. Variations in σ will vary the most probable burst lengths while bursts that differ in length from the most probable length will have correspondingly lower probabilities, as long as p_1 is sufficiently high to course many errors when the switch is closed.

Example 2.3 Error Probability on the Uniformly Distributed Burst Channel

Let us consider a uniformly distributed burst channel where only bursts of length five or less occur. Assume we are given p_1, p_2, p(1), p(2), p(3), p(4) and p(5), and n. Then the probability of no bursts occurring in a block of n bits is given by

$$p(b=0) = (1-p_2) + p_2 \left[p(1)(1-p_1) + p(2)(1-p_1)^2 + p(3)(1-p_1)^3 + p(4)(1-p_1)^4 + p(5)(1-p_1)^5\right]$$

The probability of a burst of length one is given by

$$p(b=1) = p_2 \left[p(1)\,p_1 + 2\,p(2)\,p_1(1-p_1) + 3\,p(3)\,p_1(1-p_1)^2 + 4\,p(4)\,p_1(1-p_1)^3 + 5\,p(5)\,p_1(1-p_1)^4\right]$$

The probability of a burst of length two is given by

$$p(b=2) = p_2 \left[p(2)\,p_1^2 + 2\,p(3)\,p_1^2(1-p_1) + 3\,p(4)\,p_1^2(1-p_1)^2 + 4p(5)p_1^2(1-p_1)^3\right] - p_2/n$$

The probability of a burst of length three is given by

$$p(b=3)= p_2 \left[p(3)\left[p_1^3 + p_1^2(1-p_1)\right] + 2p(4)\left[p_1^3(1-p_1)+p_1^2(1-p_1)^2\right] + 3p(5)\left[p_1^3(1-p_1)^2 + p_1^2(1-p_1)^3\right]\right] - 2p_2/n ,$$

where the first term in each of the square brackets is the probability of a solid burst and the second term is the probability of a split burst.

If $p_1 = 0.5$, $p_2 = 0.1$, $p(1)=0.2$, $p(2)=0.4$, $p(3)=0.2$, $p(4)=0.15$, and $p(5)=0.05$, then

$p(b=0) = 0.923594$

$p(b=1) = 0.042031$

$p(b=2) = 0.0142713$

$p(b=3) = 0.001355$

and finally, $p(b\leq 3) = 0.98125$.

In this example, a higher value of p_1 would have redistributed the probabilities and made $p(b=2)$ larger in relation to $p(b=1)$. Changing p_1 from 0.5 to 0.8 and leaving all other parameters the same as before yields

$p(b=0) = 0.9057856$

$p(b=1) = 0.034136$

$p(b=2) = 0.0253077$

$p(b=3) = 0.00369067$

and $p(b\leq 3) = 0.96591996$.

In the extreme, if we let $p_1=1$, then

$p(b=0) = 0.9$

$p(b=1) = 0.02$

$p(b=2) = 0.04$

$p(b=3) = 0.02$

and $p(b\leq 3) = 0.98$.

We observe that for large p_1, the distribution of burst probabilities follows the pattern of the $p(i)$.

2.2.4 The Gilbert Model

This model was devised in 1960 by E. N. Gilbert [7]. In this model the channel is considered to be made up of a number of BSCs. These BSCs represent states in a Markov process. The state in which the Markov process is when a bit is transmitted determines which BSC the bit goes through. This model is a generalization of the preceding models.

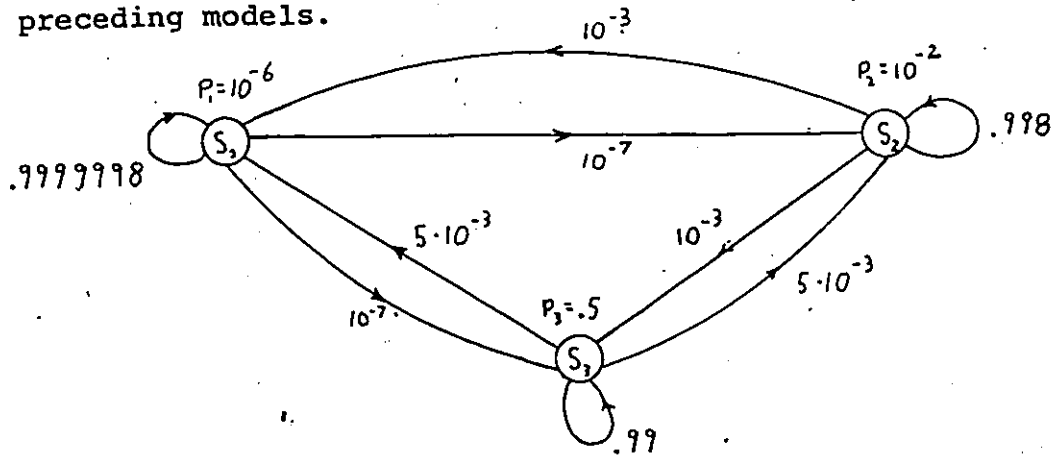

Figure 2.6 A Three State Gilbert Model

Usually, two or three states are sufficient to accurately model a real channel, on which burst and random errors occur simultaneously.

In figure 2.6 is a three state Gilbert model. Note that the probability of remaining in a state is much higher than the probability of leaving that state. Therefore, the probability of error will remain constant for long periods of time. In the figure, $p_1=10^{-6}$. When the system is in state one, only random errors will occur, and they will occur, on the average, at a rate of one error for every million bits transmitted. When the system is in state two, $p_2=10^{-2}$, so errors will occur, on the average, at a rate of one error for every hundred bits transmitted. This corresponds to random errors caused by a higher level of noise. In state three, $p_3=0.5$, so errors will occur in bursts when the system is in state three. The overall error rate is controlled by the probabilities

of the system changing from one state to another.

This model is quite flexible and is the best known model for systems when the channel behavior changes with time. This model is widely used for the telephone channel.

2.3 Error Control Through Coding

We have shown in the preceding sections that digital communication systems are affected by errors. The uses of digital information in the modern world make the occurrence of errors undesirable. High error rates can not be tolerated. Therefore, the error rate must be drastically reduced by some method.

One method of reducing the error rate is to increase the energy per bit to noise ratio (Eb/No). If $Eb \gg No$, then the probability of making an incorrect decision can be made vanishingly small. However, in some practical systems, the channel may be so noisy that the Eb/No required to bring the error probability to a reasonable value is so high that it may not be practical. This makes it impossible to achieve very low error rates by this method.

A second method of reducing the error rate is through the use of error correcting codes (ECCs). Error correcting codes cannot increase Eb/No, but they permit the same quality of reception at a lower value of Eb/No, which is as real a gain as that effected by increasing the transmitted energy.

Shannon has shown that the capacity of a linear channel with additive, white gaussian noise affecting the signal is given by [8]:

$$C = W \log_2 (1 + P/NoW) \text{ bits/sec.},$$

where

P = received signal power

No = single sided noise spectral density

W = nominal bandwidth.

He showed that , if the information rate, R, is less than C, then there exists a coding scheme that will make the probability of error arbitrarily small. If R=C, then

$$P/NoC = (2^{C/W}-1)(W/C)$$

and for the case of no bandwidth restriction, we can derive

$$\lim_{W\to\infty} P/NoC = -1.6dB$$

where P/NoC = Eb/No.

What this tells us is that, for a power limited system, any desired bit error rate (BER) can be achieved with Eb/No = -1.6dB.

In figure 2.7 , below, is plotted the probability of error, P_e, for a theoretically optimum system and for a system with no coding.

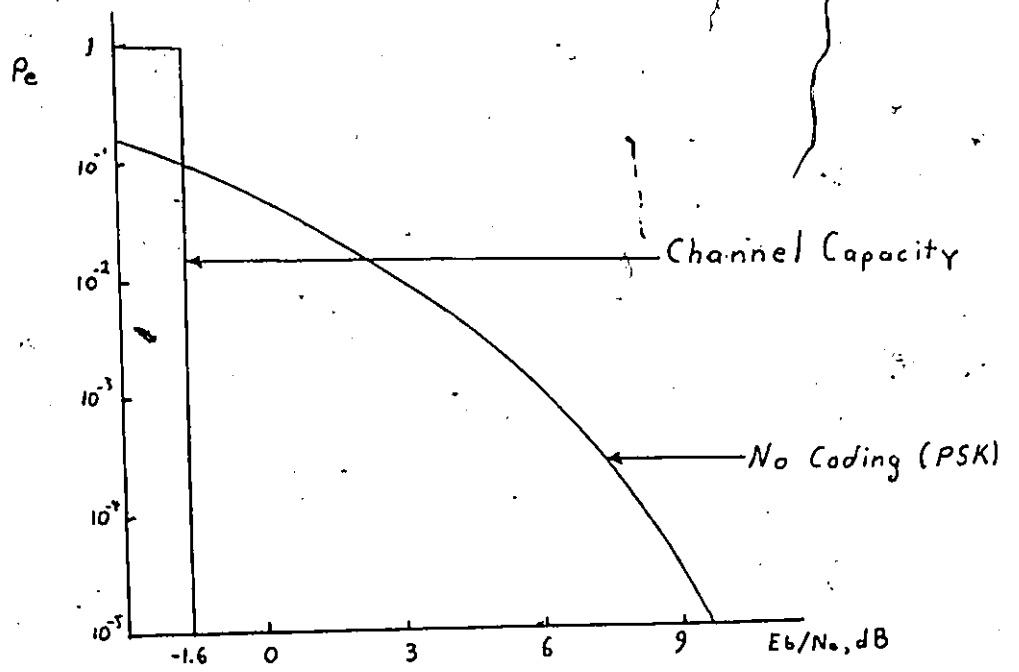

Figure 2.7 P_e vs Eb/No for no coding and for coding at channel capacity.

It can be seen from the figure that, for a BER of 10^{-5}, with no coding, 9.6 dB of Eb/No is required, while if Shannon's limit is achieved, we can obtain the same BER with Eb/No: -1.6 dB. Therefore, a potential coding gain of 11.2 dB is available. Coding schemes are available which will realize a substantial fraction of this potential gain. Some of them are in use in digital communication systems today [9,10,11].

The next chapter will deal with the types of ECCs that have been developed and are in use today.

Chapter 3. TYPES OF CODES

Error correcting codes can be classified as belonging to one or the other of two fundamentally different types [12]. In block codes, the information sequence is divided up into blocks of k digits, which are then encoded into blocks of n digits, $n>k$. Each block is encoded and decoded independently of all other blocks. The number n is called the code length, or the block length of the code.

The other type of code is called a tree code. When such a code is used, each long, perhaps semi-infinite, information sequence is encoded into a somewhat longer code sequence. The information sequence is divided into blocks of k_o symbols. Then the encoder outputs a section of the code sequence of length n_o, where the particular pattern of the n_o symbols depends not only on the k_o symbols being processed but also on some previous symbols. Such codes are called tree codes because the encoding rule is described most conveniently by a tree diagram.

The convolutional codes form a subset of the class of tree codes, which is easier to implement than other tree codes. In this thesis, the only tree codes considered will be convolutional codes.

The following sections will deal with block and convolutional codes for the correction of random, burst, and simultaneous burst and random errors.

3.1 Block Codes [13]

Block codes are formed by multiplying an information polynomial, c(x), of degree k-1 or less, by a generator polynomial of degree n-k. The resultant is a codeword polynomial of degree n-1 or less. The n coefficients of this codeword polynomial form a codeword of an (n,k) block code. Such a word has length n, has k information digits and has n-k redundant digits which are called parity check digits. The information rate of such a code is the ratio k/n.

Considering only the binary case, there are 2^k codewords out of 2^n n-tuples. We define the Hamming distance between two words to be the number of positions in which they differ.

Example 3.1

Consider two codewords, A, and B, where

A = 101101011

B = 110100111

A differs from B in positions 2, 3, 6, and 7. Hence the Hamming distance between A and B is 4.

The minimum distance, d, of a code is the smallest Hamming distance between any two of the 2^k codewords. An (n,k) code with minimum distance d, can correct $\lfloor (d-1)/2 \rfloor$ random errors, where $\lfloor x \rfloor$ indicates the largest integer smaller than x. (Similarly, $\lceil x \rceil$ denotes the smallest integer larger than x.)

3.1.1 Random Error Correcting Block Codes

The most important class of block codes is that of the cyclic codes. A cyclic code is a code that has the following property.

If

$$v(x) = a_o + a_1 x + a_2 x^2 + \ldots + a_{n-1} x^{n-1}$$

is a word of a code V, then

$$v'(x) = xv(x) \bmod (1 + x^n)$$

$$= a_{n-1} + a_o x + a_1 x^2 + \ldots + a_{n-2} x^{n-1}$$

is also a word of the same code V.

The most important class of random error correcting cyclic codes was discovered independently by Hocquenghem [14], in 1959, and by Bose and Ray-Chaudhuri [15,16], in 1960, and has come to be known as the class of BCH codes.

The BCH codes form a large class of good, constructive, codes for the correction of random errors. Furthermore, a fairly simple

decoding procedure has been developed [17], making these codes very attractive.

There exists a binary BCH code of length $n = 2^m - 1$, for any positive integer m, that will correct up to t errors, where $t < n/2$; and the code has no more than mt check bits. BCH codes can be constructed that have lengths not of the form $n = 2^m - 1$, for any n which is a submultiple of some $n' = 2^m - 1$.

Another class of random error correcting block codes is the class of majority logic decodable codes [18]. The decoding procedure is best illustrated by an example.

Example 3.2 Majority Logic Decoding for the (15,7,2) BCH Code.

The (15,7,2) BCH code is generated by $g(x) = 1 + x^4 + x^6 + x^7 + x^8$, and has minimum distance $d = 5$. Thus it can correct all patterns of two or fewer random errors. A codeword,

$$v(x) = a_o + a_1 x + a_2 x^2 + \ldots + a_{14} x^{14},$$

is formed by multiplying

$$c(x) = c_o + c_1 x + c_2 x^2 + \ldots + c_6 x^6$$

by g(x). We observe the following relationships between the coefficients of v(x) and c(x).

$a_o = c_o$

$a_1 = c_1$

$a_2 = c_2$

$a_3 = c_3$

$a_4 = c_o + c_4$

$a_5 = c_1 + c_5$

$a_6 = c_o + c_2 + c_6$

$a_7 = c_o + c_1 + c_3$

$a_8 = c_o + c_1 + c_2 + c_4$

$a_9 = c_1 + c_2 + c_3 + c_5$

$a_{10} = c_2 + c_3 + c_4 + c_6$

$a_{11} = c_3 + c_4 + c_5$

$a_{12} = c_4 + c_5 + c_6$

$a_{13} = c_5 + c_6$

$a_{14} = c_6$

Let the received word be $R(x) = v(x) + e(x)$, where $e(x)$ is an error polynomial. Let the coefficients of $R(x)$ be r_i, $i = 0, 1, 2, \ldots, 14$. We can estimate the first information bit, c_o, by:

$$c_o = r_o$$

$$c_o = r_1 + r_3 + r_7$$

$$c_o = r_4 + r_{12} + r_{13}$$

$$c_o = r_2 + r_6 + r_{14}$$

$$c_o = r_8 + r_9 + r_{11}$$

Now if $e(x)$ consists of two or fewer terms, then at least three of the estimates of c_o will be correct, since no r_i appears in more than one estimate. The decoding rule is to assign to c_o the value on which the majority of the estimates agree. Since the (15,7,2) BCH code is cyclic, the same procedure can be used to estimate the rest of the c_i's.

In general, a code with minimum distance d is said to be majority logic decodable in one step, if it is possible to form d orthogonal estimates of each information symbol. Used in this sense, orthogonal means that no r_i appears in more than one of the estimates.

If this can be done, then $t = (d-1)/2$ or fewer errors cannot affect more than $(d-1)/2$ of the estimates. Hence the majority of the estimates will be correct.

In some cases where it is not possible to form d estimates in this way, it may be possible to form the orthogonal estimates on sums of symbols, and then to form new estimates from those. If this can be repeated L times and the final estimates are for single symbols, then the code is said to be L-step majority logic decodable.

Some BCH codes as well as Euclidean Geometry and Projective Geometry codes [18] are decodable in this way and hence belong to the

class of majority logic decodable codes.

3.1.2 Burst Error Correcting Block Codes

A measure of the efficiency of linear block codes used for correcting burst errors is the Reiger bound [19]. Reiger proved that for a linear block code to correct all bursts of length b or less, it must have at least 2b check bits. Such codes are called single burst error correcting (SBEC) codes. Thus, if $z=n-k-2b = 0$ for an SBEC code, then that code is an optimum code.

It has been shown [20,21] that any cyclic code, with minimum distance d, can be used to correct bursts that satisfy

$$b \leq (3d-8)/4.$$

A class of SBEC codes with high rates, although not optimum in the Reiger bound sense, was developed by P. Fire and is called the Fire Codes [22]. These codes are generated by

$$g(x) = (x^{2b-1}-1)p(x),$$

where p(x) is an irreducible polynomial of degree $c \geq b$, with exponent e, and $(p(x), x^{2b-1}-1) = 1$. The length of the code is given by $e(2b-1)$, and the code has the form $(n, n-2b-c+1)$. Hence for these codes, the minimum value of z is given by $z = n-(n-2b-b+1)-2b = b-1$, or, to put it another way, the efficiency of these codes is given by $2b/(3b-1)$, which asymptotically approaches 2/3.

Another class of SBEC codes is that discovered by Burton [23]. These codes are of the form $(n, n-2b)$ and are generated by $g(x) = (x^b-1)p(x)$, where p(x) is a polynomial of degree b, and exponent e, and $(p(x), x^b-1) = 1$, and $n=eb$. They can correct any burst of length b or less that is confined to positions $x^{ib}, x^{ib+1}, \ldots, x^{ib+b-1}$, for $0 \leq i \leq e-1$. Interleaving techniques to be introduced in chapter four make the Burton codes asymptotically optimum with respect to the Reiger bound.

Other codes for the correction of burst errors have been computer generated and are listed in [24,25,26,27].

3.1.3 Block Codes for the Correction of the Simultaneous Occurrence of Random Errors and Burst Errors

In sections 3.1.1 and 3.1.2 we discussed codes for the correction of random errors and of burst errors. These codes will perform well on channels where error patterns that do not belong to these well defined classes occur with a low probability.

However, most real channels are more accurately modelled by the Gilbert model. That is, burst and random errors occur simultaneously, either due to the presence of both random and impulse noise or due to macroscopic changes in the channel state.

For this reason, the class of codes which will correct both burst and random errors is important. Unfortunately, this class of codes is the least well understood.

The most effective method of dealing with channels on which both burst and random errors occur or where multiple bursts occur is interleaving, which will be dealt with extensively in chapter four. However, there some classes of codes which have a built in ability to deal with such error paterns.

One simple technique which is useful in some cases is to take an (n,k) cyclic code, generated by g(x), with minimum distance $d=2t+1$. An (n,k-1) code generated by $g'(x)=g(x)(1+x)$ will correct all random error patterns of weight t or less and all bursts of length $t+1$ or less [28]. For example, from the (15,11,1) BCH code, a (15,10,1) code can be obtained which will correct all bursts of length two.

There is a small subclass of cyclic codes, called cyclic product codes which can correct both types of errors [29,30]. However, to be efficient, the code must have a low rate or be very long. Product codes are formed by using two or more random error correcting codes. If two codes are used, the method of encoding is to form arrays, where each

row of the array is a word in one of the constituent codes and each column is a word in the other code.

If an (n_1,k_1) code with $t_1=(d_1-1)/2$ and an (n_2,k_2) code with $t_2=(d_2-1)/2$ are used to form a product code, the resultant code is an (n_1n_2,k_1k_2) code with $t=(d_1d_2-1)/2$ and the code can also correct all bursts of length $b=\max(n_1t_2,n_2t_1)$ [31].

Example 3.3

If a (15,7,d=5) BCH code is used twice to form a product code, the resultant code is a (225,49) code. This code can correct all error patterns of weight 12 or less or all bursts of length 30 or less.

The array formed is a 15x15 matrix. If the matrix is transmitted row by row, then any burst of length 30 or less will result in at most two errors in each of the words making up the columns. Thus the burst will be corrected by the column code. Any random error pattern of weight eight or less can be decoded by simply decoding rows and columns in the correct order. The decoding procedure for patterns of weight between nine and twelve is, in general, more complex, but is still possible.

In addition to the guaranteed error correcting capability of a product code, many other error patterns of shorter multiple bursts and random errors are also correctable. It should be pointed out that, since a product code can be decoded in many ways, the error correction achieved will be that of the decoder, and may differ from that guaranteed by the minimum distance of the code.

Another type of code that has some burst and random error correcting capability is the class of concatenated codes developed by Forney [32]. These codes have some resemblance to product codes, in that two codes are combined to form a new code with greater error correcting ability. The communication channel using a concatenated code is shown in figure 3.1.

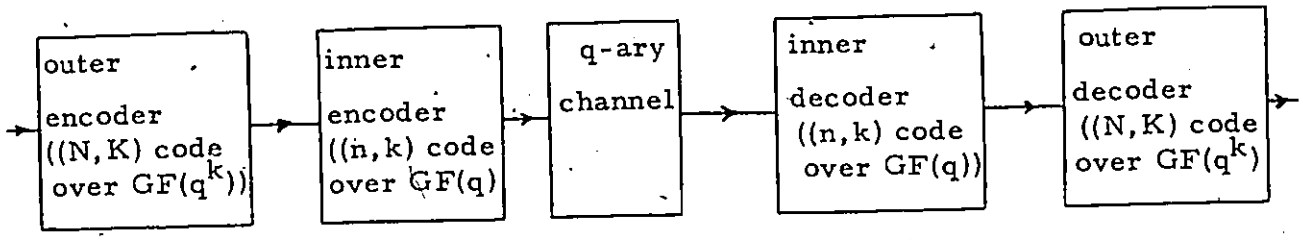

Figure 3.1 Communication Channel using a Concatenated Code.

An (N,K) outer code over GF(q^k) is coupled with an inner (n,k) code over GF(q). The result is an (nN,kK) code over GF(q). The encoding and decoding procedure is as follows. The block of kK information symbols is subdivided into K sub-blocks of k symbols each. Any symbol from GF(q^k) can be represented as a k-tuple; over GF(q). Now, the K symbols over GF(q^k) are encoded into N symbols over GF(q^k) by the outer encoder. Then each of the N symbols over GF(q^k) is considered as a k-tuple over GF(q) and encoded by the inner encoder as an n-tuple over GF(q).

The decoding procedure follows naturally. First, the n-tuples are decoded into k-tuples by the inner decoder. Then the outer decoder treats the N k-tuples as N symbols over GF(q^k), and decodes them into K k-tuples, which forms the decoded output. Errors will occur in the decoded output if and only if the errors that are not corrected by the inner code form a pattern that is not correctable by the outer code.

Example 3.4

Consider a concatenated code formed by using the (7,4), binary single error correcting Hamming code as the inner code, and the (15,7), four error correcting Reed-Solomon code over GF(2^4)

as the outer code. Each symbol in $GF(2^4)$ can be represented as a 4-tuple over GF(2). The resultant code is a (105, 28) code which can correct any error pattern where not more than four of the 15 words of the (7, 4) inner code have more than a single error. It can correct any burst of length 24 or less, since such a burst can affect at most five of the words of the inner code and one of the five will have at most a single error. Thus such a burst will appear to the outer code as a four error pattern and will be corrected.

Such a code will also correct many multiple burst and burst and random error patterns. As with product codes, concatenated codes either have low rates or are very long.

Reed-Solomon codes over $GF(q^m)$, with error correcting ability t, can be used as burst and random error correcting codes, when considered as codes over GF(q). In such a case, the codes can correct bursts of length $b = mt - m + 1$ [33]. If the burst which occurs is shorter, then other random errors or short bursts may also be correctable.

Other codes for correcting burst and random errors have been computer generated, and are listed in [34]. Many of the codes tabulated in this reference have slightly larger values of b for a given t than the shortened Reed-Solomon code of the same length and rate.

3.2 Convolutional Codes [35]

Since the main purpose of this thesis is to discuss the interleaving of block codes, convolutional codes will not be treated in depth. Only the basic properties of the convolutional codes will be introduced, with mention of some codes for the correction of the various types of errors.

A general (mn_o, mk_o) convolutional encoder is shown in figure 3.2. This encoder accepts the information symbols in blocks of k_o at a time, and outputs $n_o > k_o$ code symbols. The output symbols are linear combinations over GF(q) of the input symbols in the preceding m blocks.

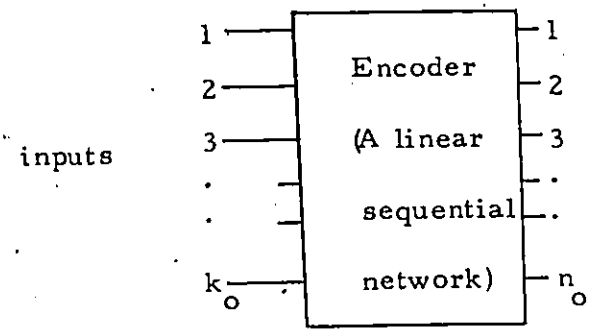

Figure 3.2 A General (mn_o, mk_o) Convolutional Encoder.

We say that the code has a decoding constraint length of $n=mn_o$, which is the block length, n_o, times the number of blocks between the first and last blocks (inclusive) that are checked by the parity bits in a block.

The encoding constraint length for a systematic code is usually the same as the decoding constraint length, but the same code in non-systematic form has a shorter encoding constraint length, and hence, fewer memory elements are required in the encoder. (A systematic code is a code where the first k_o of the n_o output symbols are the information symbols and the last n_o-k_o are the parity check symbols.)

A general decoder for a systematic convolutional code is shown in figure 3.3.

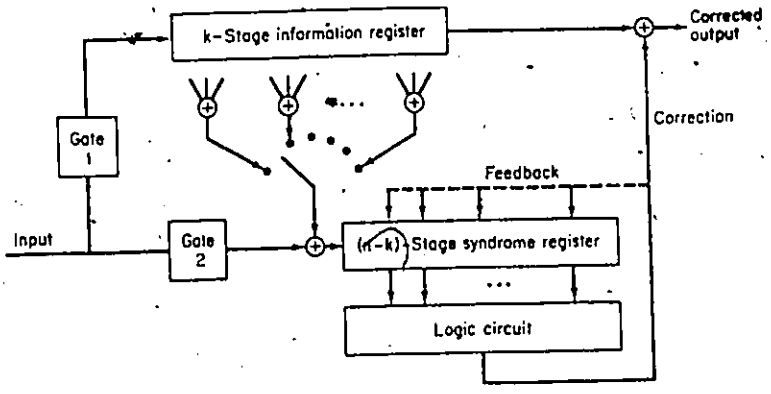

Figure 3.3 A General Decoder for a Systematic (mn_o, mk_o) Convolutional Code.

The operation of the circuit is as follows. With gate 1 open and gate 2 closed, k_o information digits are fed into the information register. Then with gate 1 closed and gate 2 open, the received parity check digits and recalculated parity check digits are compared. The difference in these two sets of parity check digits is the error syndrome which is unique for each correctable error pattern. This syndrome is fed into the syndrome register. The logic circuit then determines from the syndrome whether or not a correctable error pattern has occured in the k_o information digits ready to be outmet from the information register. If so, this block of information digits is corrected as it leaves the decoder.

In order to make use of the full error correcting capability of the code, the effect of incorrect digits on the syndrome must be removed as the errors are corrected. The dotted feedback connection does this. However if an uncorrectable error pattern occurs, the feedback connection can cause error propagation, in that the decoder can continue to decode incorrectly even though no further channel errors occur. If the effect of the feedback digits is always to reduce the weight of the syndrome, the propogation will cease after a finite, and usually small number of blocks [36] . If this condition is not met, the output of the feedback register may be periodic and infinite, or catastrophic error propogation occurs. If no feedback is used, the code does not perform up to its capability, but error propogation does not occur.

For any m, there is a single error correcting convolutional code with parameters $n_o=2^{m-1}$ and $k_o=n_o-1$, and a double error correcting convolutional code with parameters $n_o=2(2^m-1)$ and $k_o=n_o-2(m+1)$. In addition there are two classes of random error correcting convolutional codes,

the self-orthogonal codes and the orthogonalizable codes [37]. Of these the orthogonalizable codes are more efficient, but their construction is not well understood at present. Both types of codes are majority logic decodable which makes them attractive.

Other random error correcting convolutional codes have been found by computer search and are listed in [38,39].

When convolutional codes are used for burst correction, it is necessary to define both b, the burst correcting capability, and g, the guard space required to allow the burst to be corrected. By guard space, we mean the number of correct digits that must follow a burst. Any error within this guard space may cause incorrect decoding.

The burst correcting ability of an (mn_o, mk_o) convolutional code is upper bounded by [40]:

$$b \leq \frac{(m-1)(n_o-k_o)}{1 \quad k_o/n_o} + n_o - 1$$

and the guard space is upper bounded by:

$$g \leq mn_o - 1$$

For practical decoders, g cannot be much less than $mn_o - b$.

An (mn_o, mk_o) convolutional code is said to have Type-B 2 burst correcting $b_2 = rn_o$, if all bursts of length b_2 which are confined to r consecutive blocks are correctable, but at least one burst of length $(r+1)n_o$ is uncorrectable. The burst correcting ability of a code with Type-B2 burst correcting ability b_2 is bounded by:

$$b_2 + n_o - 1 \geq b \geq b_2 - n_o - 1$$

and also $b_2 \leq (n-k)/2$.

The Berlekamp-Preparata-Massey (BPM) codes [41,42,43] have Type-B2 burst correcting ability $b_2 = n_o$, and can be interleaved to produce codes with large values of b. The basic BPM codes have

parameters $n=2n_o^2$, $k=2n_o(n_o-1)$, for any n_o, and can correct any burst confined to a single n_o-bit block, provided that the burst is followed by an $(n-n_o)$-bit guard space. These codes are optimal in that $b_2=(n-k)/2$.

Iwadare codes [44] have the following parameters: $m=n_o(n_o-1)/2+(2n_o-1)$, $k_o=n_o-1$, $b=n_o$, $g=n-1$. These codes are simpler to implement than BPM codes but require a slightly larger guard space.

It is possible to construct convolutional codes that correct both burst and random errors. One way to construct such a code is to interleave a random error correcting code.

The diffuse convolutional codes have inherent burst correcting ability as well as random error correcting ability. A convolutional code is said to be b-diffuse if 1) no burst of length b or less which begins on the first information bit to be decoded affects more than t-1 of the syndrome bits used to decode that bit, and 2) no burst of length b or less which begins elsewhere affects more than t of these bits [45]. Such a code can correct simultaneously t random errors and a burst of length b.

A method called adaptive decoding [46] can be used to allow a convolutional code to correct both burst and random errors. The basic code is designed so that only a small fraction of its cosets are used for random error correction. The decoder operates in a random error correcting mode until such time as it detects an uncorrectable error pattern. Since only a small fraction of the cosets are used for random error correction, the probability of detecting patterns which are uncorrectable, before erroneous digits have left the decoder, is high. Once the decoder has detected an uncorrectable error pattern, it assumes that a burst error has occurred and switches to a

burst correcting mode. When a number of blocks have been decoded for which the syndrome contains all zeros, the decoder considers that the burst has ended and returns to the random error correcting mode. With such a scheme the ratio of burst length to guard space can be made close to unity.

In the succeding chapters on interleaved codes, only block codes will be considered although most of what will be said can be applied to convolutional codes as well.

Chapter 4 INTERLEAVING AND INTERLEAVED CODES

Interleaving is a technique that is used to spread the effect of a burst of errors among a number of codewords of a basic code. This is accomplished by spreading the symbols of a word of the basic code in time. Then, in the spaces thus produced symbols from other words are introduced.

The constituent codes of an interleaved code will be referred to as component codes. The component codes may be symbol interleaved or block interleaved. If symbol interleaving is employed, then no two symbols from any component codeword will be adjacent in the word of the interleaved code. If blocks interleaving is employed, then the component codewords are divided into blocks of length $\ell\rho$ where $\ell = b$ is usual, and each such block of a component word is separated from other blocks of the same word by a block of every other component codeword.

The number of component codewords that make up an interleaved word is called the degree of interleaving and is designated μ.

Interleaved codes are efficient in a probabilistic sense. No elaborate decoding scheme is necessary to take advantage of this extra error correcting ability. They are successful because they can correct many more error patterns than their guaranteed error correcting ability would indicate.

The following result stated as a theorem, is applicable whenever a cyclic code is interleaved to degree μ to form an interleaved code.

Theorem 4.1, [47].

If g(x) generates an (n, k) cyclic code capable of correcting all bursts of length l or less ($l = b$ for burst error correcting codes, and $l = t$ for random error correcting codes), then $g(x^{\mu})$ generates an $(\mu n, \mu k)$ cyclic code with burst error correcting ability μl.

The proof of this theorem is presented in the reference. It is proved by showing that any codeword in the $(\mu n, \mu k)$ code can be represented as an $\mu \times n$ matrix, each row of which is a codeword in the (n, k) code, and that any burst of length μl can affect at most l consecutive symbols in each row.

4.1 Interleaving of Random Error Correcting Codes

Codes which are designed to correct random errors can be interleaved to produce codes which will correct long bursts or multiple short bursts on short bursts and some random errors. If an (n, k, t) code is interleaved to degree μ, the new code is a $(\mu n, \mu k, t)$ interleaved code which is guaranteed to correct any single burst of length μt, or any t random errors. However this code will also correct at least one pattern of μ bursts of length t and another of μt random errors.

At this point we present an example to detail the error correcting ability of an interleaved code formed by interleaving a random error correcting code.

Example 4.1

Consider the (15, 7) BCH code generated by $g(x) = 1+x^4+x^6+x^7+x^8$. The information rate of this code is $R = 7/15 = .4667$, and it corrects any error pattern of weight 2 or less since the minimum distance of the code is 5. This code is majority logic decodable using the estimates.

$$c_o = r_o$$

$$c_o = r_1 + r_3 + r_7$$

$$c_o = r_4 + r_{12} + r_{13}$$

$$c_o = r_2 + r_6 + r_{14}$$

$$c_o = r_8 + r_9 + r_{11}$$

If this code is interleaved to degree $\mu=10$, the new code is a

(150, 70, d=5) interleaved code generated by

$$g'(x) = 1 + x^{40} + x^{60} + x^{70} + x^{80}.$$

The information rate, R, does not change, nor does the minimum distance; and the new code is still majority logic decodable. (Decoding techniques will be discussed in chapter 6.)

The interleaved code can correct any burst of length 2x10=20 or less, or any single or double random error. In addition, it has the following error correcting ability.

There are $\binom{150}{3} = 551,300$ error patterns of weight 3.

This code will correct

$$\binom{10}{1}\binom{15}{2}\binom{9}{1}15 + \binom{10}{3}15^3 = 546,750$$

or 99.18% of all error patterns of weight 3.

There are $\binom{150}{4} = 20,260,275$ error patterns of weight 4.

This code will correct

$$\binom{10}{2}\binom{15}{2}^2 + \binom{10}{1}\binom{15}{2}\binom{9}{2}15^2 + \binom{10}{4}15^4 = 19,632,375$$

or 96.9% of all error patterns of weight 4.

There are $\binom{150}{5} = 591,600,030$ error patterns of weight 5.

This code will correct

$$\binom{10}{2}\binom{15}{2}^2\binom{8}{1}15 + \binom{10}{1}\binom{15}{2}\binom{9}{3}15^3 + \binom{10}{5}15^5 = 548,572,500$$

or 92.73% of all error patterns of weight 5.

There are $\binom{150}{6} = 1.4297\text{x}10^{10}$ error patterns of weight 6.

This code will correct

$$\binom{10}{3}\binom{15}{2}^3 + \binom{10}{2}\binom{15}{2}^2\binom{8}{2}15^2 + \binom{10}{1}\binom{15}{2}\binom{9}{4}15^4 + \binom{10}{6}15^6 = 1.2354\text{x}10^{10}$$

or 86.41% of all error patterns of weight 6.

There are $\binom{150}{7} = 2.9411\text{x}10^{11}$ error patterns of weight 7.

This code will correct

$$\binom{10}{3}\binom{15}{2}^3\binom{7}{1}15 + \binom{10}{2}\binom{15}{2}^2\binom{8}{3}15^3 + \binom{10}{1}\binom{15}{2}\binom{9}{5}15^5 + \binom{10}{7}15^7 = 2.2932\text{x}10^{11}$$

or 77.97% of all error patterns of weight 7.

There are $\binom{150}{8} = 5.2572\text{x}10^{12}$ error patterns of weight 8.

This code will correct

$$\binom{10}{4}\binom{15}{2}^4 + \binom{10}{3}\binom{15}{2}^3\binom{7}{2}15^2 + \binom{10}{2}\binom{15}{2}^2\binom{8}{4}15^4 + \binom{10}{1}\binom{15}{2}\binom{9}{6}15^6 + \binom{10}{8}15^8$$

$$= 3.5600\text{x}10^{12}$$

or 67.72% of all error patterns of weight 8.

There are $\binom{150}{9} = 8.2947\text{x}10^{13}$ error patterns of weight 9.

This code will correct

$$\binom{10}{4}\binom{15}{2}^4\binom{6}{1}15 + \binom{10}{3}\binom{15}{2}^3\binom{7}{3}15^3 + \binom{10}{2}\binom{15}{2}^2\binom{8}{5}15^5 + \binom{10}{1}\binom{15}{2}\binom{9}{7}15^7$$

$$+ \binom{10}{9}15^9 = 4.6647\text{x}10^{13}$$

or 56.24% of all error patterns of weight 9.

There are $\binom{150}{10} = 1.1696\text{x}10^{15}$ error patterns of weight 10.

This code will correct

$$\binom{10}{5}\binom{15}{2}^5 + \binom{10}{4}\binom{15}{2}^4\binom{6}{2}15^2 + \binom{10}{3}\binom{15}{2}^3\binom{7}{4}15^4 + \binom{10}{2}\binom{15}{2}^2\binom{8}{6}15^6$$

$$+ \binom{10}{1}\binom{15}{2}\binom{9}{8}15^8 + \binom{10}{10}15^{10} = 5.1853\text{x}10^{14}$$

or 44.34% of all error patterns of weight 10.

There are $\binom{150}{11} = 1.4885\text{x}10^{16}$ error patterns of weight 11.

This code will correct

$$\binom{10}{5}\binom{15}{2}^5\binom{5}{1}15 + \binom{10}{4}\binom{15}{2}^4\binom{6}{3}15^3 + \binom{10}{3}\binom{15}{2}^3\binom{7}{5}15^5 + \binom{10}{2}\binom{15}{2}^2\binom{8}{7}15^7$$

$$+ \binom{10}{1}\binom{15}{2}\binom{9}{9}15^9 = 4.89796\text{x}10^{15}$$

or 32.9% of all error patterns of weight 11.

There are $\binom{150}{12} = 1.86065\text{x}10^{17}$ error patterns of weight 12.

This code will correct

$$\binom{10}{6}\binom{15}{2}^6 + \binom{10}{5}\binom{15}{2}^5\binom{5}{2}15^2 + \binom{10}{4}\binom{15}{2}^4\binom{6}{4}15^4 + \binom{10}{3}\binom{15}{2}^3\binom{7}{6}15^6$$

$$+ \binom{10}{2}\binom{15}{2}^2\binom{8}{8}15^8 = 3.92493\text{x}10^{16}$$

or 21.1% of all error patterns of weight 12.

There are $\binom{150}{13} = 1.97516\text{x}10^{18}$ error patterns of weight 13.

This code will correct

$$\binom{10}{6}\binom{15}{2}^6\binom{4}{1}15 + \binom{10}{5}\binom{15}{2}^5\binom{5}{3}15^3 + \binom{10}{4}\binom{15}{2}^4\binom{6}{5}15^5 + \binom{10}{3}\binom{15}{2}^3\binom{7}{7}15^7$$

$$= 2.65469\text{x}10^{17}$$

or 13.44% of all error patterns of weight 13.

There are $\binom{150}{14} = 1.93283\text{x}10^{19}$ error patterns of weight 14.

This code will correct

$$\binom{10}{7}\binom{15}{2}^7 + \binom{10}{6}\binom{15}{2}^6\binom{4}{2}15^2 + \binom{10}{5}\binom{15}{2}^5\binom{5}{4}15^4 + \binom{10}{4}\binom{15}{2}^4\binom{6}{6}15^6$$

$$= 1.50166\text{x}10^{18}$$

or 7.77% of all error patterns of weight 14.

There are $\binom{150}{15} = 1.75243\text{x}10^{20}$ error patterns of weight 15.

This code will correct

$$\binom{10}{7}\binom{15}{2}^7\binom{3}{1}15 + \binom{10}{6}\binom{15}{2}^6\binom{4}{3}15^3 + \binom{10}{5}\binom{15}{2}^5\binom{5}{5}15^5 = 7.00133\text{x}10^{18}$$

or 4% of all error patterns of weight 15.

There are $\binom{150}{16} = 1.47862 \times 10^{21}$ error patterns of weight 16.

This code will correct

$$\binom{10}{8}\binom{15}{2}^8 + \binom{10}{7}\binom{15}{2}^7\binom{3}{2}15^2 + \binom{10}{6}\binom{15}{2}^6\binom{4}{4}15^4 = 2.63093 \times 10^{19}$$

or 1.78% of all error patterns of weight 16.

There are $\binom{150}{17} = 1.1655 \times 10^{22}$ error patterns of weight 17.

This code will correct

$$\binom{10}{8}\binom{15}{2}^8\binom{2}{1}15 + \binom{10}{7}\binom{15}{2}^7\binom{3}{3}15^3 = 7.69332 \times 10^{19}$$

or 0.66% of all error patterns of weight 17.

There are $\binom{150}{18} = 8.61173 \times 10^{22}$ error patterns of weight 18.

This code will correct

$$\binom{10}{9}\binom{15}{2}^9 + \binom{10}{8}\binom{15}{2}^8\binom{2}{2}15^2 = 1.65106 \times 10^{20}$$

or 0.19% of all error patterns of weight 18.

There are $\binom{150}{19} = 5.98289 \times 10^{23}$ error patterns of weight 19.

This code will correct

$$\binom{10}{9}\binom{15}{2}^9\binom{1}{1}15 = 2.32699 \times 10^{20}$$

or 0.039% of all error patterns of weight 19.

There are $\binom{150}{20} = 3.91879 \times 10^{24}$ error patterns of weight 20.

This code will correct

$$\binom{10}{10}\binom{15}{2}^{10} = 1.62889 \times 10^{20}$$

or 0.004% of all error patterns of weight 20.

Table 4.1 lists results obtained for various random error correcting block codes, when they are interleaved as in example 4.1. The results are for interleaving to degree three.

Table 4.1 Random Error Correcting Codes Interleaved to Degree $\mu=3$

Component Code, (n,k,d)	Interleaved Code, $(\mu n, \mu k, d)$	Max. Corr. Burst	I	Error Patterns of Weight I		
				Total Number	No. Corrected	% Corrected
(7,4,3)	(21,12,3)	3	1	21	21	100%
			2	210	147	70.0%
			3	1330	343	25.79%
(7,3,4)*	(21,9,4)	6	1	21	21	100%
			2	210	165	78.57%
			3	1330	595	44.74%
			4	5985	990	16.54%
			5	20,349	756	3.72%
			6	54,264	216	0.398%
(15,11,3)	(45,33,3)	3	1	45	45	100%
			2	990	675	68.18%
			3	14,190	3375	23.78%
(15,10,4)*	(45,30,4)	6	1	45	45	100%
			2	990	717	72.42%
			3	14,190	4635	32.66%
			4	148,995	10,038	6.74%
			5	1,221,759	8820	0.722%
			6	8,145,060	2744	0.034%

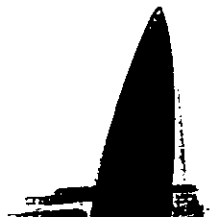

Table 4.1 (continued)

Component Code, (n,k,d)	Interleaved Code, (μn, μk, d)	Max. Corr. Burst	I	Error Patterns of Weight I		
				Total Number	No. Corrected	% Corrected
(15,7,5)	(45,21,5)	6	1	45	45	100%
			2	990	990	100%
			3	14,190	12,825	90.38%
			4	148,995	103,950	69.77%
			5	1,221,759	496,125	40.61%
			6	8,145,060	1,157,625	14.21%
(15,5,7)	(45,15,7)	9	1	45	45	100%
			2	990	990	100%
			3	14,190	14,190	100%
			4	148,995	144,900	97.25%
			5	1,221,759	1,089,900	89.21%
			6	8,145,060	6,078,450	74.63%
			7	45,379,620	24,365,250	53.69%
			8	215,553,195	65,212,875	30.25%
			9	886,163,135	94,196,375	10.63%
(31,26,3)	(93,78,3)	3	1	93	93	100%
			2	4278	2883	67.39%
			3	129,766	29,791	22.96%

Table 4.1 (continued)

Component Code, (n,k,d)	Interleaved Code, (μn, μk, d)	Max. Corr. Burst	I	Error Patterns of Weight I: Total Number	No. Corrected	% Corrected
(31,21,5)	(93,63,5)	6	1	93	93	100%
			2	4278	4278	100%
			3	129,766	116,281	89.61%
			4	2,919,735	1,989,270	68.13%
			5	51,971,283	20,108,925	38.69%
			6	762,245,484	100,544,625	13.19%
(31,16,7)	(93,48,7)	9	1	93	93	100%
			2	4278	4278	100%
			3	129,766	129,766	100%
			4	2,919,735	2,825,340	96.77%
			5	51,971,283	45,609,060	87.76%
			6	762,245,484	549,932,250	72.15%
			7	9,473,622,444	4,794,861,450	50.61%
			8	1.0184×10^{11}	2.8186×10^{10}	27.68%
			9	9.6184×10^{11}	9.0822×10^{10}	9.44%

* indicates codes formed from the previous code by multiplying its generator polynomial by (1+X).

It is known that codes formed in this way can correct bursts of length t+1.

Comparing the example of the (15,7) code interleaved to degree 10 with the table entry for the same code interleaved to degree three, we note that, as the degree of interleaving is increased, the percentage of lower weight error patterns that are correctable increases rapidly. This is as expected, since, as more and more component codewords are inserted, the probability of multiple errors in the same component word decreases. In the limit, as the degree of interleaving approaches infinity, all error patterns become correctable. However, the degree of interleaving is limited by the length of the codeword that can be stored at the receiver.

A useful feature of codes formed in this fashion is that, even if the total error pattern is uncorrectable, some parts of the pattern may be corrected. Any component word that has a correctable error pattern will correct it, thus reducing the number of errors in the decoder output. It is the independent decoding of the component words that makes this possible.

In general, interleaved codes formed by interleaving random error correcting codes are quite powerful for correcting burst, or burst and random error patterns. They are not efficient in the Reiger bound sense, but their ability to correct multiple bursts, and random errors, in addition to their basic burst correcting ability, that makes them attractive.

If the channel is such that only burst errors occur, then SBEC codes should be used, as they will correct the same length burst with fewer check bits than the codes just described. However, on the compound channel, as we will show in chapter five, there are at least some cases where interleaved random error correcting codes seem better than either product codes or concatenated codes for the correction of simultaneous burst and random errors.

4.2 Interleaving of Burst Error Correcting Codes

Long codes with good burst error correcting properties can be formed by interleaving short codes. In particular, if a short (n,k) code has $z = n-k-2b = 0$, then the $(\mu n, \mu k)$ code formed by interleaving to degree μ also has $z' = z = 0$. In general, the interleaved code has $z' = \mu z$. Thus, these interleaved codes can be optimal in the Reiger bound sense.

These interleaved codes can not only correct bursts of length b, where b is the burst error correcting ability of the component code, but also they can correct many patterns of multiple shorter bursts. As with codes formed by interleaving random error correcting codes, interleaved burst error correcting codes tend to reduce the number of errors in uncorrectable error patterns, since any component word which contains a correctable segment of the pattern will correct it.

Example 4.2 shows the extended burst correcting ability that can be achieved by interleaving short burst error correcting codes.

Example 4.2

Consider the $(15, 9, b=3)$ code [48] generated by

$$g(x) = 1 + x^3 + x^4 + x^5 + x^6.$$

If this code is interleaved to degree ten, the $(150, 90)$ code formed will correct all bursts of length 30 of less. The generator polynomial for this code is

$$g'(x) = g(x^{10}) = 1 + x^{30} + x^{40} + x^{50} + x^{60},$$

if the code is symbol interleaved. This code is optimal in the Reiger bound sense, since $z = 150-90-2(30) = 0$. Interleaved in this fashion, the code will correct any burst pattern that is confined to 30 consecutive symbols.

This same code can be formed by block interleaving, and will then be able to correct some multiple burst patterns. The block interleaving is performed by taking three symbols at a time from each component word. The component words must be independently

generated, and then combined to form the interleaved codeword.

The (150, 90) block interleaved code can then correct some patterns of 10 bursts of length three or less, five bursts of length six or less, three bursts of length nine or less plus one burst of length three, or two bursts of length 15 or less. Not all such bursts will be corrected, but any pattern which results in not more than three consecutive incorrect symbols in any component word will be corrected.

As with interleaved random error correcting codes, the performance of these codes improves as the degree of interleaving increases, because the probability of an uncorrectable error pattern in a component word decreases.

In table 4.2, we present some burst error correcting codes formed by the interleaving of some good, short, SBEC codes. The codes are interleaved to a degree that brings the burst error correcting ability up to approximately 3000 symbols. In this table $z = n-k-2b$, and $z' = \mu n - \mu k - 2\mu b = \mu z$.

It is evident from the table that the interleaved codes are efficient in the Reiger bound sense only if $z = 0$ for the component code, or the degree of interleaving is small, since $z' = \mu z$. It is also worthy of note that the codes with $z = z' = 0$ tend to have lower rates. Thus, efficiency in the Reiger bound sense does not imply efficiency in the information rate sense.

Using the component codes in table 4.2, or shortened versions of them, it is possible to develop interleaved codes for any desired burst correcting ability. The code designed can meet the Reiger bound, or have a high information rate but not both. The codes are attractive because they correct long bursts or multiple short bursts, and hence are efficient error correctors in a probabalistic sense.

Table 4.2 Interleaving of Burst Error Correcting Codes (for μb approximately 3000)

Component Code, (n,k)	b	z= n-k-2b	μ	Code (μn, μk)	$b' = \mu b$	$z' = \mu n - \mu k - 2\mu b$	Information Rate, R
(7,3) [49]	2	0	1500	(10500,4500)	3000	0	.42857
(15,9) [49]	3	0	1000	(15000,9000)	3000	0	.600
(27,17) [49]	5	0	600	(16200,10200)	3000	0	.62963
(34,22) [49]	6	0	500	(17000,11000)	3000	0	.64706
(50,34) [49]	8	0	375	(18750,12750)	3000	0	.6800
(56,38) [26]	9	0	333	(18648,12654)	2997	0	.67857
(59,39) [26]	10	0	300	(17700,11700)	3000	0	.66102
(65,43) [27]	11	0	272	(17680,11696)	2992	0	.66154
(72,48) [27]	12	0	250	(18000,12000)	3000	0	.66667
(78,52) [27]	13	0	230	(17940,11960)	2990	0	.66667
(82,54) [27]	14	0	214	(17548,11556)	2996	0	.65854
(27,20) [25]	3	1	1000	(27000,20000)	3000	1000	.74074
(38,29) [25]	4	1	750	(28500,21750)	3000	750	.74359
(48,37) [25]	5	1	600	(28800,22200)	3000	600	.77083
(67,54) [25]	6	1	500	(33500,27000)	3000	500	.80597
(103,88) [25]	7	1	428	(44084,37664)	2996	428	.85437
(96,79) [25]	8	1	375	(36000,29625)	3000	375	.82292
(121,102) [27]	9	1	333	(40293,33966)	2997	333	.84298
(127,106) [27]	10	1	300	(38100,31800)	3000	300	.83465

Table 4.2 (continued)

Component Code, (n,k)	b	z= n-k-2b	μ	Code (μn, μk)	b'=μb	z'=μn-μk-2μb	Information Rate, R
(111,88) [27]	11	1	272	(30192,23936)	2992	272	.79279
(131,106) [27]	12	1	250	(32750,26500)	3000	250	.80916
(63,55) [49]	3	2	1000	(63000,55000)	3000	2000	.87302
(85,75) [49]	4	2	750	(63750,56250)	3000	1500	.88235
(131,119) [49]	5	2	600	(78600,71400)	3000	1200	.90840
(169,155) [49]	6	2	500	(84500,77500)	3000	1000	.91716
(121,112) [49]	3	3	1000	(121000,112000)	3000	3000	.92562
(290,277) [49]	5	3	600	(174000,166200)	3000	1800	.95517
(511,499) [49]	4	4	750	(383250,374250)	3000	3000	.97652

Chapter 5. INTERLEAVING OF RANDOM ERROR CORRECTING CODES WITH BURST ERROR CORRECTING CODES.

Thus far, we have discussed the interleaving of random error correcting codes for the correction of bursts and random errors, and the interleaving of burst error correcting for the correction of long bursts or multiple short bursts.

We now present a concept which is a logical extension of these ideas, but about which there seems to have been no literature published to date. This concept is the interleaving of random error correcting codes with burst error correcting codes.

Codes so formed exhibit an interesting and useful **probabilistic** error correcting capability, in that they can correct **bursts slightly** longer than the basic burst, or interleaved burst correcting code used and also many patterns of short bursts and random errors.

The method of forming the code is as follows. The component words are generated by independent generators, and then interleaved by a timing circuit. The burst correcting code is divided into blocks of length b and between each block we insert t or fewer symbols from the random error correcting code. The random error correcting code is chosen to match the burst correcting code in the following manner.

If, in the (n,k,b) burst correcting code, b divides n, then the (n',k',t) random error correcting code should be such that, if x is the number of symbols from the random error correcting code to be inserted between each block of the burst error correcting code, then $n'/x = n/b + 1$.

If, in the burst error correcting code, b does not divide n, then the random error correcting code should be chosen so that $n'/x = \lceil n/b \rceil$. This procedure is illustrated in the following examples.

Example 5.1

In the (15, 9, b 3) code, b divides n, and $n/b = 5$. The random code chosen should have $n/b + 1 = 5 + 1 = 6$ blocks of length t or less.

The (7, 4, 1) Hamming code can be shortened to (6, 3, 1) which is suitable for interleaving with the (15, 9, b=3) code. Letting X represent a symbol from the (15, 9, b=3) code and 0 represent a symbol from the (6, 3, 1) code, the interleaved codeword would have the form:

0XXX0XXX0XXX0XXX0XXX0

This is a (24, 12, b=4) code with $z = n-k-2b = 1$.

Example 5.2

In the (27, 17, b = 5) code, $b \nmid n$, and $\left\lceil \frac{n}{b} \right\rceil = \left\lceil \frac{27}{5} \right\rceil = 6$.

Therefore the same (6, 3, 1) code used in example 5.1 is acceptable. Using the same notation, the interleaved codeword would have the form:

0XXXXX0XXXXX0XXXXX0XXXXX0XXXXX0XX

This is a (33, 20, b = 6) code with z = 1.

It would appear, from example 5.2, that the timing circuit that performs the actual interleaving might be more complex in the case where $b \nmid n$, since there is a certain lack of symmetry. In practice however this makes little or no difference, as a counter is required to recognize the end of a codeword in any case. This unsymmetrical point in the codeword is an ideal place in which to insert synchronizing bits for framing purposes.

5.1 Comparison of Interleaved Random and Burst Error Correcting Codes with SBEC Codes

We will now illustrate with an example, the error correcting ability of codes formed in this fashion and compare it with the error correcting ability of a code with the same parameters that corrects only bursts and a code of the same length with optimum burst correcting ability.

Example 5.3

Consider the (21, 12, b = 4) code formed in example 4.1, from the (15, 9, b = 3) code and the (6, 3, 1) code. The (15, 9, b = 3) code corrects

$$E_1 = 15 + 14 + 13 \cdot 2 = 55 \text{ error patterns.}$$

The (6, 3, 1) code corrects

$$E_2 = \binom{6}{1} = 6 \text{ error patterns.}$$

The (21, 12, b = 4) code corrects all combinations of errors correctable in either or both component codes.

Therefore it corrects

$$E = E_1 + E_2 + E_1 E_2 = 55 + 6 + 55 \cdot 6 = 391$$

error patterns.

A (21, 12, b = 4) burst error correcting code and correct

$$E^1 = 21 + 20 + 19 \cdot 2 + 18 \cdot 4 = 151$$

error patterns all of which are contained in E.

The optimum code of this length would be a (21, 11, b = 5) code, which corrects

$$E^{11} = 21 + 20 + 19 \cdot 2 + 18 \cdot 4 + 17 \cdot 8 = 287 \text{ error patterns.}$$

We can see from this example, that if the number of error patterns that a code corrects is taken as a criterion of quality, then the interleaved code is superior to the other two codes mentioned. If the channel is such that random errors occur as well as short bursts, then the interleaved code will indeed outperform the other two. The only one of the two that can outperform the interleaved code is the optimum (21, 11, b = 5) code on a channel where errors are restricted to burst patterns and where burst patterns of length five have a fairly high probability.

Table 5.1 is a list of codes formed as in example 5.1 and 5.2 with the error analysis corresponding to example 5.3.

In the table, E is the number of error patterns correctible by the interleaved code, E' is the number of error patterns correctable by a code of the same length and burst correcting ability, and E'' is the number of error patterns correctable by a code of the same length but with optimum burst correcting ability. The component codes used in the table are SBEC codes with z = 1, 2 and random error correcting codes which are either the (7,4,1) Hamming code or its shortened version or the trivial (7,1,3) or (5,1,2) codes which have only two words.

Table 5.1 is divided into three sections. In section A the burst error correcting codes are optimum and the random error correcting code is the (7,4,1) Hamming code and its shortened versions. In section B, the burst error correcting codes are still optimum, but have been shortened so that $b \mid n$. The random error correcting codes in section B are the (7, 1, 3) and (5, 1, 2) codes. In section C, the same random error correcting codes are used with non-optimum burst correcting codes.

In all three sections the interleaved code corrects considerably more error patterns that either of the two codes with which it is compared. In section C the same burst correcting code has been interleaved with the (5, 1, 2) code and the (7, 1, 3) code. This shows that by increasing z from 3 to 5 and decreasing the rate a little, the number of correctible error patterns can be more than tripled.

If we consider, E, E', and E'' as sets of error patterns then we have that

$$E' \subset E \quad ; \quad E' \subset E''$$

and

$$[E - E'] \cap [E'' - E'] = \emptyset$$

If b is the burst correcting ability of the interleaved code, and e is an error pattern, then $e \in E'$ iff e is a burst of length b or less. Also $e \in [E - E']$ implies e is a burst of length less than b plus a random error, and $e \in [E'' - E']$ implies e is a burst error pattern of length b+1 for codes in section A, b+1 or b+2 for codes in section B and b+2 or b+3 for codes in

Table 5.1 Interleaved Codes Formed from one Burst and one Random Error Correcting Code.

Component Codes	Code	z	R	E	E'	E''
A						
(15,9,b=3);(6,3,1)	(21,12,b=4)	1	.5714	391	151	287
(19,11,b=4);(5,2,1)	(24,13,b=5)	1	.5417	815	275	639
(27,17,b=5);(6,3,1)	(33,20,b=6)	1	.6016	2687	927	1791
(34,22,b=6);(6,3,1)	(40,25,b=7)	1	.625	6719	2239	4351
(38,24,b=7);(6,3,1)	(44,27,b=8)	1	.6136	14,783	4863	9471
(50,34,b=8);(7,4,1)	(57,38,b=9)	1	.6667	45,055	12,799	25,087
(56,38,b=9);(7,4,1)	(63,42,b=10)	1	.6667	100,351	28,159	55,295
(59,39,b=10);(6,3,1)	(65,42,b=11)	1	.6462	182,783	57,343	112,639
B						
(15,9,b=3);(5,1,2)	(20,10,b=4)	2	.500	615	143	271
(16,8,b=4);(5,1,2)	(21,9,b=5)	2	.4286	1231	287	543
(25,15,b=5);(5,1,2)	(30,16,b=6)	2	.5333	3863	831	1599
(30,18,b=6);(5,1,2)	(35,19,b=7)	2	.5429	9151	1919	3647
(35,21,b=7);(5,1,2)	(40,22,b=8)	2	.550	21,119	4351	8447
(48,32,b=8);(5,1,2)	(53,33,b=9)	2	.6226	59,135	11,775	23,039
(54,36,b=9);(5,1,2)	(59,37,b=10)	2	.6271	132,351	26,111	51,199
(50,30,b=10);(5,1,2)	(55,31,b=11)	2	.5636	236,543	47,103	92,159
(48,32,b=8);(7,1,3)	(55,33,b=9)	4	.600	193,535	12,287	47,103
(54,36,b=9);(7,1,3)	(61,37,b=10)	4	.6066	433,151	27,135	104,447
C						
(12,7,b=2);(5,1,2)	(17,8,b=3)	3	.4706	263	63	223
(12,7,b=2);(7,1,3)	(19,8,b=3)	5	.4211	863	71	479

Table 5.1 (continued)

Component Codes	Code	z	R	E	E'	E''
(18,11,b=3);(5,1,2)	(23,12,b=4)	3	.5217	747	167	607
(18,11,b=3);(7,1,3)	(25,12,b=4)	5	.480	2447	183	1279
(24,15,b=4);(5,1,2)	(29,16,b=5)	3	.5517	1935	415	1535
(24,15,b=4);(7,1,3)	(31,16,b=5)	5	.5161	6335	447	3199
(30,19,b=5);(5,1,2)	(35,20,b=6)	3	.5714	4751	991	3711
(30,19,b=5);(7,1,3)	(37,20,b=6)	5	.5405	15,551	1055	7679
(36,23,b=6);(5,1,2)	(41,24,b=7)	3	.5854	11,263	2303	8703
(36,23,b=6);(7,1,3)	(43,24,b=7)	5	.5581	36,863	2431	17,919
(42,27,b=7);(5,1,2)	(47,28,b=8)	3	.5957	25,887	5247	19,967
(42,27,b=7);(7,1,3)	(49,28,b=8)	5	.5714	85,087	5503	40,959
(48,31,b=8);(5,1,2)	(53,32,b=9)	3	.6038	59,135	11,023	44,303
(48,31,b=8);(7,1,3)	(55,32,b=9)	5	.5818	193,535	12,287	92,159

section C. This specifies the types of errors correctable by the three codes under consideration. That is, all codes correct bursts of length b or less, the interleaved codes can correct shorter bursts plus random errors, and the optimum code of the same length and with the same number of, or one less, information symbols corrects slightly longer burst errors.

We will now consider the case where the random error correcting code has a generator polymomial which has $1+x$ as a factor. An (n, k, t) random error correcting code which has a generator polymomial which has $1+x$ as a factor can correct burst error patterns of length t+1. By taking advantage of this fact, it's possible to develop efficient interleaved codes.

The interleaving technique will again be illustrated by examples.

Example 5.4

A (27, 13, b=7) interleaved code can be formed by using as component codes the (20, 10, b=5) code and the (7, 3, 1) code. The (7, 3, 1) code corrects single random errors or bursts of length two since its generator polymomial has 1+x as a factor. The (20, 10, b=5) code divided into $\frac{20}{5} = 4$ blocks of length 5. In the space between blocks we insert 2 symbols from the (7, 3, 1) code. The seventh symbol is added at the end of the codeword. Using X for symbols from the (20, 10) code and 0 for symbols from the (7, 3, 1) code, a codeword in the interleaved code has the form:

XXXXX00XXXXX00XXXXX00XXXXX0.

Example 5.5

A (29, 16, b=6) code can be formed by interleaving a (19, 11, b=4) code and a (10, 5, 1) code. The (10, 5, 1) cpde is a shortened version of the (15, 10, 1) code which has (1+x) as a factor in its generator polymomial. Using the same notation as in example 4.5, the interleaved codeword has the form:

00XXXX00XXXX00XXXX00XXXX00XXX

To take advantage of the maximum burst correcting ability available, it is essentialthat nonadjacent symbols from the random error correcting component code be separated by b symbols from the burst correcting code. Note that, in example 5.5, if the two initial symbols had been placed in the final positions instead, the burst correcting ability of the interleaved code would have been five instead of six.

A list of codes formed by interleaving burst correcting codes with codes that correct single random errors or burst errors of length two is presented in table 5.2.

Table 5.2 Interleaved Codes Formed form an SBEC and a Code that Corrects Single Random or Double-Adjacent Errors

Component Codes	Code	z	R	E	E'	E''
A						
(12,6,b=3);(7,3,1)b=2	(19,9,b=5)	0	.4737	615	255	255
(16,8,b=4);(7,3,1)b=2	(23,11,b=6)	0	.4783	1567	607	607
(20,10,b=5);(7,3,1)b=2	(27,13,b=7)	0	.4815	3807	1407	1407
(24,12,b=6);(7,3,1)b=2	(31,15,b=8)	0	.4839	8959	3119	3119
(28,14,b=7);(7,3,1)b=2	(35,17,b=9)	0	.4857	20,607	7165	7165
(32,16,b=8);(7,3,1)b=2	(39,19,b=10)	0	.4872	46,631	15,871	15,871
(36,18,b=9);(7,3,1)b=2	(43,21,b=11)	0	.4884	103,935	34,815	34,815
(40,20,b=10);(7,3,1)b=2	(47,23,b=12)	0	.4894	229,735	75,775	75,775
B						
(7,3,b=2);(8,3,1)b=2	(15,6,b=4)	1	.400	223	103	191
(15,9,b=3);(12,7,1)b=2	(27,16,b=5)	1	.5926	1343	383	735
(19,11,b=4);(10,5,1)b=2	(29,16,b=6)	1	.5517	2415	799	1535
(27,17,b=5);(12,7,1)b=2	(39,24,b=7)	1	.6154	9215	2175	4223
(34,22,b=6);(12,7,1)b=2	(46,29,b=8)	1	.6304	23,039	5119	9983
(38,24,b=7);(12,7,1)b=2	(50,31,b=9)	1	.620	50,687	11,007	21,503
(50,34,b=8);(14,9,1)b=2	(64,43,b=10)	1	.6719	157,695	28,671	56,319
(56,38,b=9);(14,9,1)b=2	(70,47,b=11)	1	.6714	351,231	62,463	122,879
(59,39,b=10);(12,7,1)b=2	(71,46,b=12)	1	.6479	626,687	124,927	245,759
C						
(14,9,b=2);(15,10,1)b=2	(29,19,b=4)	2	.6552	839	215	415
(24,17,b=3);(15,10,1)b=2	(39,27,b=5)	2	.6923	2759	575	1119
(32,23,b=4);(15,10,1)b=2	(47,33,b=6)	2	.7021	7199	1375	2687

Table 5.2 (continued)

Component Codes	Code	z	R	E	E'	E''
(40,29,b=5);(15,10,1)b=2	(55,39,b=7)	2	.7091	17,759	3198	6270
(48,35,b=6);(15,10,1)b=2	(63,45,b=8)	2	.7143	42,239	7295	14,335
(56,41,b=7);(15,10,1)b=2	(71,51,b=9)	2	.7183	97,919	16,383	32,255
(64,47,b=8);(15,10,1)b=2	(79,57,b=10)	2	.7215	222,879	36,351	71,679
D (15,10,b=2);(16,10,1)b=2	(31,20,b=4)	3	.6452	959	231	863
(27,20,b=3);(20,14,1)b=2	(47,34,b=5)	3	.7234	4219	703	2687
(38,29,b=4);(20,14,1)b=2	(58,43,b=6)	3	.7414	11,519	1669	6655
(48,37,b=5);(20,14,1)b=2	(68,51,b=7)	3	.750	28,799	4031	15,615
(67,54,b=6);(24,18,1)b=2	(91,72,b=8)	3	.7912	96,334	10,079	42,495
(103,88,b=7);(30,24,1)b=2	(133,112,b=9)	3	.8421	376,319	32,255	126,975
(96,79,b=8);(26,20,1)b=2	(122,99,b=10)	3	.8115	599,039	58,367	229,375
E (31,25,b=2);(32,25,1)b=2	(63,50,b=4)	5	.7937	3967	487	3711
(63,55,b=3);(44,37,1)b=2	(107,92,b=5)	5	.8598	16,342	1663	12,927
(85,75,b=4);(44,37,1)b=2	(129,112,b=6)	5	.8682	58,431	3999	31,231
(131,119,b=5);(54,47,1)b=2	(185,166,b=7)	5	.8973	221,183	11,519	90,623
(169,155,b=6);(58,51,1)b=2	(227,206,b=8)	5	.9075	612,479	28,287	223,231

Again E is the number of error patterns corrected by the interleaved code, E' is the number of error patterns corrected by a code with th the same parameters, but no random error correcting ability, and E'' is the number of error patterns corrected by a code of the same length with optimum burst correcting ability equal to or greater than that of the interleaved code.

Table 5.2 is divided into five sections, according to the value of z for the interleaved code. In section A, the burst error correcting component codes used are shortened versions of burst correcting codes that are optimum. In this section we take advantage of the fact that when we add the factor (1+x) to the generator polymomial of the (7, 4, 1) Hamming code, the (7, 3, 1) b=2 code formed is also optimum as a burst correcting code. Thus the interleaved codes formed in section A are optimum in the sense that for the lengths given there exist no better burst correcting codes. In the worst case, the interleaved code corrects 2.41 times as many error patterns as a code with the same parameters but no random error correcting ability, while in the best case the improvement is 3.03.

In section B, full length optimum burst correcting codes are used with shortened versions of the (15, 10, 1) b=2 code. For the interleaved code z=1, since the (15, 10, 1) b=2 code has z=1 as a burst correcting code. In this section the number of error patterns corrected is increased by from 2.16 to 5.6 over the same code with no random error correcting ability, and by 1.17 to 2.86 over an optimum code of the same length, but with a slightly lower rate.

In section C, we use burst error correcting codes with z of 1 and the (15, 10, 1) b=2 code which also has a z of 1. Thus z=2 for the interleaved code. In this case the improvement in the number of corrected error patterns is from 3.9 to 6.13 when the interleaved code is compared to the same code with no random error correcting ability, and from 2.02 to 3.11 when the interleaved code is compared to an optimum burst correcting code of the same length and rate.

In section D, the burst error correcting code has z=1, and the random error correcting code is a shortened version of the (31, 25, 1) b=2 BCH code which has z=2 when used as a burst correcting code. Thus the interleaved code

has z=3. The improvement in the number of corrected error patterns over an SBEC with the same parameters varies from 4.15 to 10.26, and over an optimum SBEC of the same length but with k reduced by one symbol varies from 1.11 to 2.96.

In section E the SBECs used as component codes have z=2, while the random error correcting code is a shortened version of the (63, 56, 1) b=2) BCH code. The improvement in the number of corrected error patterns over an SBEC with the same parameters varies from 8.15 to 21.65, while over an optimum SBEC of the same length but with k reduced by one symbol varies from 1.07 to 2.74.

From this table it is clear that the interleaved codes outperform the SBECs if the criterion is the number of corrected error patterns. The interleaved codes correct all error patterns corrected by the SBECs of the same parameters and while they do not correct all the error patterns corrected by the optimum burst correcting codes (unless z=0 for the interleaved code), they do correct a greater total number of error patterns.

To this point we have considered the case of two component codes. As there is no reason for this restriction, we will now consider the case where an SBEC is interleaved with two random error correcting codes, both with double adjacent error correcting ability. For simplicity we will use the same random error correcting code twice.

If the SBEC code corrects E_1 error patterns and the random error correcting code corrects E_2 error patterns, then the number of error patterns, E_T, corrected by the interleaved code is given by

$$E_T = (E_1+1)(E_2+1)(E_2+1) - 1$$
$$= E_1 E_2^2 + 2E_1 E_2 + E_2^2 + E_1 + 2E_2$$

In table 5.3 we repeat section A to D of table 5.2, but with the random error correcting code used twice. In some cases shortening has been necessary to get the maximum burst correcting ability possible.

Table 5.3 SBEC Codes Interleaved with two Codes that Correct Single Random Errors or Double-Adjacent Errors.

Component Codes	Code	z	R	E	E'	E''
A						
(12,6,b=3); (7,3,1)b=2;(7,3,1)b=2	(26,12,b=7)	0	.4615	8623	1343	1343
(16,8,b=4); (7,3,1)b=2;(7,3,1)b=2	(30,14,b=8)	0	.4667	21,951	3071	3071
(20,10,b=5); (7,3,1)b=2;(7,3,1)b=2	(34,16,b=9)	0	.4706	53,311	6911	6911
(24,12,b=6) (7,3,1)b=2;(7,3,1)b=2	(38,18,b=10)	0	.4737	125,439	15,359	15,359
(28,14,b=7); (7,3,1)b=2;(7,3,1)b=2	(42,20,b=11)	0	.4762	288,511	33,791	33,791
(32,16,b=8); (7,3,1)b=2;(7,3,1)b=2	(46,22,b=12)	0	.4783	652,287	73,727	73,727
(36,18,b=9); (7,3,1)b=2;(7,3,1)b=2	(50,24,b=13)	0	.480	1,455,103	159,743	159,743
(40,20,b=10); (7,3,1)b=2;(7,3,1)b=2	(54,26,b=14)	0	.4815	3,211,263	344,063	344,063
B						
(7,3,b=2); (8,3,1)b=2;(8,3,1)b=2	(23,9,b=6)	2	.3913	3583	607	1151
(15,9,b=3); (12,7,1)b=2;(12,7,1)b=2	(39,23,b=7)	2	.5897	32,255	2175	4223
(19,11,b=4); (10,5,1)b=2;(10,5,1)b=2	(39,21,b=8)	2	.5385	54,399	4223	8191
(27,17,b=5); (12,7,1)b=2;(12,7,1)b=2	(51,31,b=9)	2	.6078	211,183	11,263	22,015
(34,22,b=6); (12,7,1)b=2;(12,7,1)b=2	(58,36,b=10)	2	.6207	552,959	25,599	50,175
(38,24,b=7); (12,7,1)b=2;(12,7,1)b=2	(62,38,b=11)	2	.6129	1,216,511	54,271	106,495

Table 5.3 (continued)

Component Codes	Code	z	R	E	E'	E''
(50,34,b=8); (14,9,1)b=2;(14,9,1)b=2	(78,52,b=12)	2	.6667	4,415,487	139,623	274,791
(56,38,b=9); (14,9,1)b=2;(14,9,1)b=2	(84,56,b=13)	2	.6667	9,834,495	299,007	589,823
(59,39,b=10); (12,7,1)b=2;(12,7,1)b=2	(83,53,b=14)	2	.6386	15,040,511	581,631	1,146,879
C						
(15,10,b=2); (14,9,1)b=2;(14,9,1)b=2	(43,28,b=6)	3	.6512	23,519	1247	4735
(24,17,b=3); (14,9,1)b=2;(14,9,1)b=2	(52,35,b=7)	3	.6731	72,127	3007	11,519
(32,23,b=4); (14,9,1)b=2;(14,9,1)b=2	(60,41,b=8)	3	.6833	188,159	6911	26,623
(40,29,b=5); (14,9,1)b=2;(14,9,1)b=2	(68,47,b=9)	3	.6912	464,127	15,615	60,415
(48,35,b=6); (14,9,1)b=2;(14,9,1)b=2	(76,53,b=10)	3	.6974	1,103,871	34,751	136,103
(56,41,b=7); (14,9,1)b=2;(14,9,1)b=2	(84,59,b=11)	3	.7024	2,558,975	76,799	299,007
(64,47,b=8); (14,9,1)b=2;(14,9,1)b=2	(92,65,b=12	3	.7065	5,820,415	167,935	655,359
D						
(15,10,b=2); (16,10,1)b=2;(16,10,1)b=2	(47,30,b=6)	5	.6383	30,628	1375	10,239
(27,20,b=3); (20,14,1)b=2;(20,14,1)b=2	(67,48,b=7)	5	.7164	166,399	3967	30,207
(38,29,b=4); (20,14,1)b=2;(20,14,1)b=2	(78,57,b=8)	5	.7308	460,799	9215	70,655
(48,37,b=5); (20,14,1)b=2;(20,14,1)b=2	(88,65,b=9)	5	.7386	1,151,999	20,735	159,743

Table 5.3 (continued)

Component Codes	Code	z	R	E	E'	E''
(67,54,b=6); (24,18,1)b=2;(24,18,1)b=2	(115,90,b=10)	5	.7826	4,644,863	54,783	425,983
(103,88,b=7); (30,24,1)b=2;(30,24,1)b=2	(163,136,b=11)	5	.8344	22,579,199	157,695	1,236,991
(96,79,b=8); (26,20,1)b=2;(26,20,1)b=2	(148,119,b=12)	5	.8041	15,748,095	282,623	2,221,839

In section A of table 5.3, the codes are optimum as in the same section of table 5.2, and the improvement factor on the same basis of comparison is now in the range from 6.42 to 9.33.

Comparing section B of table 5.2 and 5.3, we see that while z has increased from one to two, the improvement factor over an SBEC with the same parameters has increased to from 5.9 to 32.89, and over an optimum SBEC to from 3.11 to 13.11.

In section C, z has increased from two to three, but the comparable improvement factors are now in the ranges from 18.86 to 34.66 and from 4.97 to 8.88.

In section D, z has increased from three to five, and the comparable improvement factors have increased to from 22.27 to 143.18 and to from 2.99 to 18.25.

From this comparison we see that the improvement factor increases with the number of random error correcting codes used, at least in this case.

We will now prove that, under quite general conditions, the improvement factor increases as the number of random error correcting codes increases. The proofs of the following lemmas and theorem are given for the binary case only, but the extension to the q-ary case is easily obtained.

Lemma 5.1

If, in a block of n symbols, there are a possible B burst error patterns of length b or less, then for $b \geq 1$, there are exactly

$$B' = B - b2^{b-1} + \sum_{i=1}^{b-1} i2^{i-1}$$

burst error patterns of length b+1.

Proof:

For the binary case, B is given by:

$$B = n + (n-1) + 2(n-2) + 4(n-3) + \ldots + 2^{b-2}(n-b+1)$$

$$= n + \sum_{i=2}^{b} 2^{i-2}(n-i+1)$$

Expanding the summation, we obtain

$$B = n + (n-1)2^0 + (n-2)2^1 + (n-3)2^2 + \ldots + (n-b-1)2^{b-2}$$

$$= n + n(2^0 + 2^1 + 2^2 + \ldots + 2^{b-2}) - (1x2^0 + 2x2^1 + 3x2^2 + \ldots + (b-1)2^{b-2})$$

$$= n + n((2^{b-1}-1)/(2-1)) - \sum_{i=1}^{b-1} i2^{i-1}$$

$$= 2^{b-1}n - \sum_{i=1}^{b-1} i2^{i-1}$$

Also from [50], for $i \geq 2$, there are

$$2^{i-2}(n-i+1)$$

burst error patterns of length i. For $i = b+1$, there are

$$2^{(b+1)-2}(n-(b+1)+1) = 2^{b-1}(n-b)$$

burst error patterns.

But

$$B + \sum_{i=1}^{b-1} i2^{i-1} = 2^{b-1}n$$

If B' is the number of burst error patterns of length $b+1$, then:

$$B' = 2^{b-1}(n-b)$$

$$= B - b2^{b-1} + \sum_{i\ 1}^{b-1} i2^{i-1}$$

Q.E.D.

Corollary 5.1

If there are B burst error patterns of length b or less, then there are

$$B'' = 2B - b2^{b-1} + \sum_{i=1}^{b-1} i2^{i-1}$$

burst error patterns of length $b+1$ or less.

This follows directly from Lemma 5.1 and the fact that the number of bursts of length $b+1$ or less must be the sum of the number of bursts of length $b+1$ and those of length b or less.

Lemma 5.2

If there are B burst error patterns of length b or less, and B'' of length $b+1$ or less, then $B'' < 2B$.

Proof:

Since, by corollary 5.1,

$$B'' = 2B - b2^{b-1} + \sum_{i=1}^{b-1} i2^{i-1}$$

the lemma is proven if it can be shown that

$$\sum_{i=1}^{b-1} i2^{i-1} < b2^{b-1}$$

Now, a series of the form $(ix^{i-1}, i\ 0,1,2,\ldots)$ can be summed according to

$$\sum_{i=0}^{n} ix^{i-1} = \frac{d}{dx}\left(\frac{1 - x^{n\ 1}}{1-x}\right)$$

Substituting $x=2$, $n=b-1$, we obtain

$$\sum_{i=1}^{b-1} i2^{i-1} = b2^{b-1} + (1 - 2^b)$$

and, therefore, for $b \geq 1$, we have that

$$\sum_{i=1}^{b-1} i2^{i-1} < b2^{b-1}$$

and hence $B'' < 2B$.

Q. E. D.

Lemma 5.3

If $n' \leq 2n+1 - b$ and there are B_1 burst error patterns of length b or less in a block of n bits, and B_2 burst error patterns of length b or less in a block of n' bits, then $B_2 < 2B_1$ for $b \geq 1$.

Proof:

Since $B_2 = 2^{b-1}n' - \sum_{i=1}^{b-1} i2^{i-1}$ and $B_1 = 2^{b-1}n - \sum_{i=1}^{b-1} i2^{i-1}$,

we require the condition on n' such that

$$2^{b-1}n' - \sum_{i=1}^{b-1} i2^{i-1} < 2(2^{b-1}n - \sum_{i=1}^{b-1} i2^{i-1})$$

or $$2^{b-1}n' - \sum_{i=1}^{b-1} i2^{i-1} < 2^b n - \sum_{i=1}^{b-1} i2^i .$$

Now $$\sum_{i=1}^{b-1} i2^{i-1} = b2^{b-1} + (1 - 2^b)$$

and $$\sum_{i=1}^{b-1} i2^i = 2 - 2^b + b2^b - 2^b = 2 - 2x2^b + b2^b$$

$\therefore$ if $$2^{b-1}n' - \sum_{i=1}^{b-1} i2^{i-1} < 2^b n - \sum_{i=1}^{b-1} i2^i$$

then $2^{b-1}n' - b2^{b-1} - (1-2^b) < 2^b n - 2 + 2x2^b - b2^b$

or $n' - b - 1/2^{b-1} + 2 < 2n - 2/2^{b-1} + 4 - 2b$

or $n' < 2n + 2 - b - 1/2^{b-1}$

But, since n' must be an integer, it must be that

$n' \leq 2n + 1 - b$

$\therefore$ If $n' \leq 2n + 1 - b$, then $B_2 < 2B_1$

Q.E.D

The preceding lemmas will now be used to prove the following theorem.

Theorem 5.1

An (n,k,b) interleaved code is constructed according to the previously described method. If an (n',k',b') interleaved code is formed from the (n,k,b) code by the interleaving of an additional random error correcting code, and if $n' \leq 2n + 1 - b$, then the improvement factor increases if the random error correcting code can correct s error patterns, where

$$s \geq 2^{b'} - b + 1$$

Proof:

By lemma 5.3, increasing the length of the code from n to n' increases the number of burst patterns of length b or less by a factor smaller than 2. By lemma 5.2, an increase of one in the burst length increases the number of burst patterns by a factor of less than 2. Therefore, increasing the burst length from b to b' can increase the number of bursts by a factor which is less than $2^{b'-b}$. Therefore, the number of error patterns in the burst error correcting code used for comparison can increase by no more than $2^{b'-b+1}$.

In the interleaved code, if j component codes are used, the number of correctable error patterns is the sum of the products of the component code error correcting capabilities, taken one, then two, then three, etc, up to j at a time. Thus if the

(j+1) st code can correct a total of s error patterns, the total increase in correctible error patterns will be by a factor greater than s.

$\therefore$ If $s \geq 2^{b' - b+1}$, then the improvement factor increases.

Q.E.D.

According to the preceding lemmas and theorem, the improvement factor increases as the number of random error correcting component codes is increased as long as the new length, n', is related to the previous length, n by $n' \leq 2n+1-b$ and the added random error correcting code can correct at least $s=2^{b' - b+1}$ error patterns, where b' is the new burst length and b is the previous burst length. These conditions are met in every case that has been considered. Indeed it would be difficult to think of even a trivial case where they are not satisfied.

As the burst correcting component code is usually long compared to the random error correcting component code, the first condition is generally satisfied, if not for the first additional subcode, then for subsequent ones. If n is large compared to $\beta - 1$, and since n' is not much greater than n, then $n' \leq 2n+1-b$ is satisfied. The second condition, that $s \geq 2^{b' - b+1}$, is also quite generally satisfied except for trivial cases. For example, a (3,1,1) code corrects 3 error patterns and could concievably be used to increase b by one symbol; then $2^{b' - b+1} = 2^2 = 4$ and $s=3 < 4 = 2^{b'-b+1}$. If the (5,1,2) code is used, b can be increased by two symbols and $s=\binom{5}{2}=10 > 2^{b'-b+1}=8$. From this we see that the improvement factor does increase with additional random error correcting codes in almost every case. It should be pointed out at this time that the conditions derived are sufficient but not necessary.

Thus for we have only considered, as random error correcting component codes, those codes which can correct single random errors or double adjacent errors. Table 5.4 shows an expansion to the case where the random error

correcting code can correct two random errors, and two random errors or triple adjacent errors. The SBEC component codes used in this table are the optimum codes, while the random error correcting codes are shortened versions of the (15, 7, 2), the (15, 6, 2) b=3, the (31, 21, 2) and the (31, 20, 2) b=3 BCH codes. Entries in the table are arranged in pairs, with each optimum SBEC code being interleaved first with a double random error correcting code and then with a double random error, triple adjacent error correcting code. The quantities E, E' and E" are as defined in previous tables and z is again the quantity n - k-2b which is a measure of how good a code is in the Reiger bound sense.

The improvement factor of the interleaved code over an SBEC code with the same parameters, defined as the ratio E/E' , ranges from 9.53 to 21.29 with the double-error correcting code, and from 9.82 to 22.95 with the double-error, triple adjacent-error correcting code. When the interleaved code is compared with an optimum SBEC of the same length (with possibly a smaller k) then the improvement factor, defined as the ratio E/E", ranges from 2.58 to 5.50 with the double error correcting code and from 1.58 to 3.00 with the double-error, triple adjacent-error correcting code.

Again we see that the interleaved codes outperform the SBEC codes in every case. It can also be noted that, when the random error correcting code has (1+x) as a factor in its generator polymomial, the improvement factor E/E' is greater while E/E" is smaller. This is due to the fact that adding the factor (1+x) decreases k for the random error correcting code by one symbol. This adds one symbol to the burst length that the comparable optimum SBEC can correct. Hence E" increases more than E does when the factor (1+x) is added.

It may be argued that the comparison in this section is not valid since the codes compared are SBEC codes and codes that correct the simultaneous occurrance of bursts and random errors. However, even if the interleaved

Table 5.4 Interleaved Codes Formed from Optimal SBEC Codes and Multiple Random Error Correcting Codes.

Component Codes	Code	z	R	E	E'	E''
(15,9,b=3);(12,4,2)	(27,13,b=5	4	.4815	4423	383	1,407
(15,9,b=3);(18,7,2)b=3	(33,16,b=6)	5	.4848	10,527	927	6655
(19,11,b=4);(10,2,2)	(29,13,b=6)	4	.4483	7615	799	2943
(19,11,b=4);(15,6,2)b=3	(34,17,b=7)	3	.5000	18,223	1855	6911
(27,17,b=5);(12,4,2)	(39,21,b=7)	4	.5385	30,335	2175	8191
(27,17,b=5);(18,7,2)b=3	(45,24,b=8)	5	.5333	72,191	4991	36,863
(34,22,b=6);(12,4,2)	(46,26,b=8)	4	.5652	75,839	5119	19,455
(34,22,b=6);(18,7,2)b=3	(52,29,b=9)	5	.5577	180,479	11,519	86,015
(38,24,b=7);(12,4,2)	(50,28,b=9)	4	.560	166,847	11,007	41,983
(38,24,b=7);(18,7,2)b=3	(56,31,b=10)	5	.5536	397,055	24,575	184,319
(50,34,b=8);(14,6,2)	(64,40,b=10)	4	.6250	596,991	28,671	110,591
(50,34,b=8);(21,10,2)b=3	(71,44,b=11)	5	.6197	1,413,631	63,487	483,327
(56,38,b=9);(14,6,2)	(70,44,b=11)	4	.6286	1,329,663	62,463	241,663
(56,38,b=9);(21,10,2)b=3	(77,48,b=12)	5	.6234	3,148,543	137,215	1,048,575
(59,39,b=10);(12,4,2)	(71,43,b=12)	4	.6056	2,062,847	124,927	483,327
(59,39,b=10);(18,7,2)b=3	(77,46,b=13)	5	.5974	4,909,055	270,335	2,064,383

codes are considered only for their burst correcting ability, they are superior to SBEC codes since they can correct some multiple burst patterns which the SBEC codes cannot. In addition, it will be shown in chapter 6, that the interleaved codes, at least in some cases, can be decoded in less time than SBEC codes, through the use of parallel decoding techniques. Thus if decoding speed is important, there is a case for the use of interleaved codes to replace SBEC codes.

In the following section, the error correcting ability of the interleaved codes will be compared with that of some of the other codes mentioned in section 3.1.3 that correct the simultaneous occurrence of burst and random errors.

5.2 Comparison of Interleaved Random and Burst Error Correcting Codes with Product codes, Concatenated Codes, and Reed-Solomon Codes.

In this section, some of these three types of codes will be generated and the number of error patterns that they can correct will be computed. We will then compare this number with that for interleaved codes similar to those in tables 5.1 to 5.4, having similar parameters. The comparison will include the information rate and the maximum length of correctable bursts.

5.2.1 Product Codes [29,30,51].

In section 3.1.3, it was shown that an (n_1, k_1, t_1) code with $t_1 = (d_1-1)/2$ and an (n_2, k_2, t_2) code with $t_2 = (d_2-1)/2$ can form a product code with parameters (n_1n_2, k_1k_2, t), where $t = (d_1d_2-1)/2$, and that this code can correct all burst error patterns of length $b = \max(n_1t_2, n_2t_1)$ or less.

We will now show, by means of three examples, the comparison between the number of error patterns guaranteed to be correctable by interleaved codes and by product codes of similar rates.

Example 5.6

Consider the (105,28) product code formed from the (15,7,2) and the (7,4,1) codes.

We have $d_1 = 5$, $d_2 = 3$, and hence, for the product code $t=(5x3-1)/2 = 7$, and the burst correcting ability of the code is $b = \max(15x1, 7x2) = 15$.

The number of random errors, E_R, correctable with this code is given by

$$E_R = \sum_{i=1}^{7} \binom{105}{i} = 2.44716017x10^{10}$$

The number of burst error patterns, E_B, is given by [50]

$$E_B = 2^{b-1}(n-b+1)+1 - 1$$
$$= 2^{14}(105-15+2)-1$$
$$= 2^{14}(92)-1$$
$$= 1,507,327$$

The total number of error patterns, E_{TP}, that the product code is guaranteed to correct, is then given by

$$E_{TP} = (E_R + 1)(E_B + 1)-1$$
$$= 3.688673045x10^{16}$$

We will now compute the number of error patterns correctable by some interleaved codes.

The (15,5,3) BCH code can be interleaved with itself seven times to produce a (105,35) code, which has a slightly higher rate than the (105,28) product code. The (15,5,3) BCH code can correct a total of

$$E_1 = \sum_{i=1}^{3} \binom{15}{i} = 575 \quad \text{error patterns.}$$

Thus the (105,35) can correct a total of

$$E_a = (E_1 + 1)^7 - 1 = 2.103572012x10^{19} \quad \text{error patterns.}$$

The improvement factor is then $E_a/E_{TP} = 570.28$.

Now consider a (108, 28) interleaved code formed by using the (30, 10, b = 10) code twice and the (12, 2, 3) code four times.

The (30, 10, b = 10) code corrects

$$E_2 = 2^9((30-10+1)+1)-1 = 11,263 \text{ error patterns.}$$

The (12, 2, 3) code corrects

$$E_3 = \sum_{i=1}^{3}\binom{12}{i} = 298 \text{ error patterns.}$$

Thus the (108, 28) interleaved code corrects

$$E_b = (E_2+1)^2(E_3+1)^4-1$$

$$= 1.014074908\text{x}10^{18} \text{ error patterns, for an improvement factor of } E_b/E_{TP} = 27.49.$$

Example 5.7

The direct product of the (15, 7, 2) code with itself is a (225, 49) product code. We have $d_1 = d_2 = 5$ and hence $t = (25-1)/2 = 12$ and $b = 30$.

The code guarantees correction of

$$\sum_{i=1}^{12}\binom{225}{i} = 2.761610207\text{x}10^{19} \text{ random error patterns}$$

and

$$2^{29}(197)-1 = 1.057635697\text{x}10^{11} \text{ burst error patterns.}$$

Thus the total number of guaranteed correctable error patterns is $2.920777535\text{x}10^{30}$.

If we interleave the (15, 7, 2) code with itself 15 times, we obtain a (225, 105) code which corrects

$$121^{15}-1 = 1.744940227\text{x}10^{31} \text{ error patterns.}$$

Thus the improvement factor is 5.97, and the interleaved code has a rate, k/n, which is more than twice that of the product code. Both codes correct bursts of length 30 or less.

If we interleave the (15,5,3) code with itself 15 times, the resultant code is a (225,75) code which corrects

$$576^{15}-1 = 2.548808762\text{x}10^{41} \quad \text{error patterns.}$$

In this case, the improvement factor is $8.73\text{x}10^{10}$, and the interleaved code has a rate more than 50% higher than the product code. The interleaved code corrects bursts of length 45, which is also 50% longer than the product code is guaranteed to correct.

If we interleave the (50,30,b = 10) code with itself twice and with the (15,5,3) code eight times, the resultant is a (220,100) code which corrects

$$(2^9(50-10+2))^2(576)^8-1 = 5.602970942\text{x}10^{30} \quad \text{error}$$

patterns. The improvement factor is 1.92, and the rate is more than twice that of the product code.

One might think that this code is inferior to the (15,7,2) code used 15 times as the rates are about the same and the improvement factor is smaller. However, the (15,7,2) code, interleaved 15 times, can correct bursts of length 30 or less, while the code just discussed can handle bursts up to length 44. Thus the choice of code would depend on the channel characteristics.

Example 5.8

Consider the product code formed by the (7,4,1) code used twice. The result is a (49,16) code which corrects four or fewer random errors and bursts of length seven or less. The total number of guaranteed correctable error patterns is given by

$$E_{TP} \quad (2^6(44))\left(\sum_{i=1}^{4}\binom{49}{i}\right)-1 = 651,977,215.$$

The (45,15,) code formed by interleaving the (15,5,3) code three times corrects

$$576^3 - 1 = 191,102,975 \text{ error patterns,}$$

and the improvement factor is 0.29, and no longer represents an improvement.

If we interleave the (12,2,3) code four times, we obtain a (48,8) code which corrects $299^4 - 1 = 7,992,538,800$ error patterns, for an improvement factor of 12.26. However the interleaved code has a rate that is only about half that of the product code. The interleaved code corrects bursts of length 12, compared to only seven for the product code.

These examples show that, if the product code is long enough that a similar length code can be constructed by interleaving many short codes, then the interleaved code can exhibit a large improvement factor, even if the interleaved code has a considerably higher rate than the product code.

The last example shows that, if the product code is fairly short, and hence only a few codes can be used in interleaving a code of similar length, then either the improvement factor tends to be less than one, or the interleaved code tends to have a lower rate.

It must be pointed out here that the decoding process that enables a product code to correct up to its guaranteed ability is complex. What is usually used is a probabilistic decoding scheme that will not correct all error patterns of weight $(d_1 d_2 - 1)/2$, but will correct many patterns of greater weight.

The actual number of error patterns that can be corrected by a product code depends to a large degree on the decoding technique used, and, in fact, will probably be higher than the figures computed in the examples.

We cannot, therefore, conclude from this study that interleaved codes are better than product codes. In order to make this conclusion it would be necessary to calculate the actual number of error patterns correctable by a product code for the best decoding scheme available, and then see how the interleaved codes compare.

Since the examples indicate that the improvement factor is favorable only for comparitively long codes, the calculation would be extremely tedious and would consume much computer time.

However, we can say, as far as guaranteed error correction is concerned, that long product codes do not seem to be as good as interleaved codes. We also point out, from [29], that for a BSC, if one code has error probability $f_1(p)$ and another $f_2(p)$, then their product is capable of decoding with probability of error no greater than $f_2(f_1(p))$.

For the same channel, an interleaved code would have error probability $f(p)=f_1(p)+f_2(p)$. In general, $f_2(f_1(p)) < f_1(p)+f_2(p)$ and thus the product code has a lower error probability than the interleaved code, provided that both codes are constructed from the same component codes. However, in such a case, the product code would have a rate much lower than the interleaved code, so that we cannot say that the product code is superior.

5.2.2 Concatenated Codes

In section 3.1.3, concatenated codes were introduced. Now we will treat these codes as we did the product codes; that is, we will form some concatenated codes of lengths ands rates comparable to some interleaved codes, and then compare the total number of correctable error patterns for both types of codes.

Following Forney [32], we will use Reed-Solomon codes for the outer codes and binary BCH codes for the inner codes.

In example 3.4, we introduced a (105, 28) concatenated code, which used the (7,4,1) BCH code as an inner code and the (15,7,4) R.-S. code over GF(2^4) as an outer code. A codeword of the (105, 28) concatenated code thus formed consists of, in binary form, 15 7-tuples, each of which are codewords in the (7,4,1) code. When these 7-tuples are decoded into 4-tuples, which are then symbols from GF(2^4), the resultant is a codeword in the (15,7,4) R.-S. code.

A concatenated code can correct error patterns that are distributed in such a way that, after the inner decoder has operated on the codeword, the remaining errors are confined to t_1 or fewer symbols, where t is the number of errors correctable by the outer code.

We will now investigate the actual number of error patterns that such a code can correct. We will derive a general formula for the number of error patterns guaranteed correctable by a concatenated code and then apply the formula to some specific examples for comparison with interleaved codes.

Consider a concatenated code formed from an (N, K, t_1) outer code over GF(2^k) and an (n, k, t_2) inner code. The outer code corrects t_1 errors; therefore the inner code must correct all the errors in $(N-t_1)$ of the n-tuples making up the codeword. The remaining t_1 n-tuples can have up to n errors as these will be seen as single errors by the outer code. A single codeword of the inner code can correct e_1 error patterns, where e_1 is given by:

$$e_1 = \sum_{i=1}^{t} \binom{n}{i}$$

and $N-t_1$ of the n-tuples must contain e_1 or fewer errors.

The total number of correctable error patterns is given by:

$$E_{TOT} = \binom{N}{t_1}\left(\sum_{i=0}^{n} \binom{n}{i}^{t_2}\right)\left(\sum_{i=1}^{N-t_1} \binom{N-t_1}{i}(e_1+1)^i\right) - 1$$

Example 5.9

Consider the (105, 52) code formed by the concatenation of the (15, 13, 1) Reed-Solomon code with the (7, 4, 1) BCH code. For this code

$$E_{TC} = (\binom{15}{1}\sum_{i=0}^{7}\binom{7}{i})(\sum_{i=1}^{14}\binom{14}{i}8^i) - 1$$

$$= 4.392344151 \times 10^{16}$$

The (105, 35) code formed by interleaving the (15, 5, 3) code with itself 7 times corrects $2.078140532 \times 10^{19}$ error patterns for an improvement factor of 473.13. In addition the concatenated code, while it corrects some bursts of length 8, is only guaranteed to correct bursts of length 3, while the interleaved code handles bursts of length 21 or less.

The (105, 35) code formed by interleaving the (30, 10, b = 10) code twice with itself and three times with the (15, 5, 3) code corrects $2.424670539 \times 10^{16}$ error patterns. The improvement factor is less than unity in this case, but the interleaved code will correct bursts of up to length 29.

Example 5.10

Consider the (105, 44) code formed by concatenating the (15, 11, 2) R.-S. code with the (7, 4, 1) BCH code. For this code

$$E_{TC} = (\binom{15}{2}(\sum_{i=0}^{7}\binom{7}{i})^2)(\sum_{i=1}^{13}\binom{13}{i}8^i) - 1$$

$$= 4.372822622 \times 10^{18}$$

We compare this with the (135, 45) code formed by interleaving the (30, 10, b=10) with itself twice and with the (15, 5, 3) code five times. This gives almost the same number of information symbols, but at a lower rate. This codes corrects

$(11,264)^2\ (576)^5\ -1 = 8.044474896 \times 10^{21}$ error patterns for an improvement factor of 1839.65. If the (15, 5, 3) code is used 4 times instead of 5, the interleaved code corrects $1.396610225 \times 10^{19}$ error patterns for an improvement factor of 3.19.

Example 5.11

Consider the (279, 115) code formed by concatenating the (31, 23, 4) R.-S. code over $GF(2^5)$ with the (9, 5, 1) BCH code which is a shortened version of the (15, 11, 1) code. For this code

$$E_{TC} = (\binom{31}{4}(\sum_{i=0}^{9}\binom{9}{i})^4)\,(\sum_{i=1}^{27}\binom{27}{i}10^i) - 1$$

$$= 2.919 \times 10^{43}$$

The interleaved code formed by using the (15, 5, 3) code 19 times is a (285, 95) code which corrects bursts of length 57 or less. It corrects a total of 2.80561×10^{52} error patterns for an improvement factor of 9.6×10^8. The interleaved code formed by using the (30, 10, b=10) code three times and the (15, 5, 3) code 13 times is a (285, 95) code which corrects bursts of length 69 or less and a total of 1.0979188×10^{48} error patterns for an improvement factor of 3.76×10^4.

The interleaved code formed by using the (28, 14, b=7) code three times and the (15, 5, 3) code 13 times is a (279, 107) code which corrects bursts of length 60 or less and a total of 2.4452896×10^{45} error patterns for an improvement factor of 83.77. The concatenated code can correct some bursts of length 37, but is only guaranteed to correct bursts of length 30 or less.

The number of correctable error patterns for concatenated codes seems to increase slowly with the value of t for the outer code and rapidly as the length of the code increases; that is, as the order of the Galois field increases.

The long code can be compared to an interleaved code formed from many short component codes. As we have noted in the preceding section such an interleaved code is capable of correcting a great many error patterns.

The preceding examples indicate that, at least in some case, interleaved codes can have a greater guaranteed error correcting ability than concatenated codes.

In the examples we have considered strictly algebraic decoding. Another decoding technique, called generalized minimum distance (GMD) decoding has been devised [52] which combines some of the advantages of algebraic decoding with the extended power of probabilistic decoding.

With this technique, each received symbol , r_i, is assigned a value (0 or 1 in the binary case) and a reliability class, C_i, which indicates how sure the receiver is that the assigned value is correct. With each class one associated two parameters B_{ci} and B_{ei} such that $0 \leq B_{ci} \leq B_{ei} \leq 1$, and the weight of class C_i is given by $\alpha_i = B_{ei} - B_{ci}$.

Then the generalized distance, $D_g(r, f)$, between a received word r and a codeword f is defined as

$$D_g(r, f) \quad \sum_{i=1}^{n} d_g(r_i, f_i)$$

where

$$d_g(r_i, f_i) = \begin{cases} B_{ci}, & r_i = f_i \text{ and } r_i \text{ in } C_i \\ B_{ei}, & r_i \neq f_i \text{ and } r_i \text{ in } C_i \end{cases}$$

If a word f from a code with minimum distance d is transmitted and the number of symbols received correctly ($r_i = f_i$) and put in class C_i is n_{ci}, and the number received incorrectly ($r_i \neq f_i$) and put in class C_i is n_{ei}, then, if n_{ci} and n_{ei} are such that

$$\sum_i ((1 - \alpha_i)n_{ci} + (1+\alpha_i)n_{ei}) < d \qquad (*)$$

then $D_g(r, f) < D_g(r, g)$ for all codewords $g \neq f$.

The decoding procedure is to consider some of the least reliable classes as completely unreliable and then use standard errors and erasures decoding, considering the unreliable symbols as erasures. After decoding, equation (*) is checked. If it is satisfied, then the correct codeword has been found. Otherwise, the choice of which of the C_i's to consider as completely unreliable is changed. With this method, if there is a word such that (*) is satisfied it will be found in (d 1)/2 fewer trials.

Forney [52, 32] has shown that the performance of concatenated codes can be improved by letting the output of inner decoder be a set of GMD values, that is values plus weights, and then letting the outer decoder perform GMD decoding. While this procedure is complex, it can be implemented, and in such a case, the comparison between concatenated codes and interleaved codes may no longer be valid. However we may still say in this case that the interleaved codes can correct longer bursts and are more easily decoded.

5.2.3 Reed-Solomon Codes

The product codes and concatenated codes considered thus far have either low rates or are very long. We will now consider using Reed-Solomon codes for the correction of burst and random errors.

It is known that a t-error correcting R.-S. code over $GF(q^m)$ can correct all bursts of length $mt - m+1$ when regarded as a code over GF(q). Since every such burst has weight less than or equal to t, over $GF(q^m)$, the cosets containing bursts, over GF(q) are disjoint from the cosets containing error patterns of weight t or less over GF(q).

These codes are not designed to correct burst and random errors simultaneously. That is, if a burst of length mt-m+1 occurs, then any additional random errors make the pattern uncorrectable. However if a shorter burst occurs, then some random errors may be corrected also.

For example, a (127,119,4) R.-S. code over $GF(2^7)$ can be considered as an (889,833,4) code over GF(2). This code over GF(2) is guaranteed to correct four or fewer random errors, or bursts of length 22 or less. However, it will correct some bursts of length 28, some bursts of length 21 plus a random error, some bursts of length 14 plus two random errors or two short bursts, and many other patterns as well.

Reed-Solomon codes over $GF(q^m)$ have parameters $n = q^m-1$, $k = q^m-1-2t$, or $n = 2^m-1$, $k = 2^m-1-2t$ when $q=2$. When considered as codes over GF(2), the parameters are $(n=(2^m-1)m,\ k=(2^m-1-2t)m,\ t)$. They can be shortened to any desired length over $GF(2^m)$, but over GF(2), the length must remain a multiple of m.

If we consider an (N, N-2t, t) Reed-Solomon code over $GF(2^m)$ as an (mN, m(N-2t), t) code over GF(2), then we can provide a general formula for the number of correctable error patterns. We can have up to m errors in up to t of the N m-tuples that make up the codeword over GF(2). Thus, the total number of correctable error patterns is given by:

$$E_{TRS} = \sum_{j=1}^{t} \binom{N}{j} \left(\sum_{i=0}^{m} \binom{m}{i} \right)^j - 1$$

We will now work out some examples to see how Reed-Solomon codes compare with interleaved codes.

Example 5.12

The (15,11,2) Reed-Solomon code over $GF(2^4)$ can be considered as a (60,44) binary code over GF(2). It corrects bursts of length mt-m+1 = 5. It corrects a total of

$$E_{TRS} = \sum_{j=1}^{2} \binom{15}{j} \left(\sum_{i=0}^{4} \binom{4}{i}\right)^j - 1$$

$$= 15(16) + 105(16)^2 - 1 = 27,119 \text{ error patterns.}$$

The (15,11,1) BCH code, interleaved four times, forms a (60,44) code that corrects all bursts of length four or less. It corrects a total of $16^4 - 1 = 65,535$ error patterns for an improvement factor of 2.42.

The (31,21,2) BCH code, interleaved twice, forms a (62,42) code that corrects all bursts of length four or less and a total of

$$\left(1 + \binom{31}{1} + \binom{31}{2}\right)^2 - 1 = 247,008 \text{ error patterns}$$

for an improvement factor of 9.11.

The (62,40) code formed by interleaving the (48,32,b = 8) code with the (7,4,1) BCH code twice, corrects bursts of length 10 or less and a total of 344,063 error patterns for an improvement factor of 12.69.

Example 5.13

Consider the Reed-Solomon codes of length 31, over $GF(2^5)$, and having t = 1,2,3. The (31,29,1) R.-S. code corresponds to a (155,145) code over GF(2). It corrects 991 error patterns and gives no guaranteed burst error correction. There is a (255,239,2) BCH code which can be shortened to a (155,139,2) code which corrects 12,090 error patterns for an improvement factor of 12.2. This code corrects every double adjacent error pattern while the R.-S. code will not correct double adjacent error patterns

that overlap symbols.

The (63, 56, 1)b=2 code, interleaved twice, forms a (126, 112) code which corrects bursts of length four or less. This code corrects a total of $(63+62)^2-1 = 15,624$ error patterns for an improvement factor of 15.77. This interleaved code is considerably shorter than the (155, 145) code but has almost the same rate.

The (31, 27, 2) R.-S. code over $GF(2^5)$ can be considered as a (155, 135) code over GF(2). It corrects bursts of length six or less and a total of 477,151 error patterns. The (31, 26, 1) code interleaved four times produces a (124, 104) code which corrects 923,520 error patterns for an improvement factor of 1.94. If the (31, 26, 1) code is interleaved five times, the resultant is a (155, 130) code which corrects 28,629,150 error patterns for an improvement factor of 60. Both of these interleaved codes have a slightly lower rate than the R.-S. code.

The (31, 25, 3) R.-S. code over $GF(2^5)$ is a (155, 125) code over GF(2). It corrects a total of 1.477631×10^8 error patterns. The (15, 11, 1) BCH code interleaved 10 times produces a (150, 110) code which corrects $5.766503906 \times 10^{11}$ error patterns for an improvement factor of 3903, although the interleaved code has a lower rate. In this case, the R.-S. code corrects bursts up to length 11, while the interleaved code corrects bursts of length 10 or less.

The (50, 30, b=10) code can be interleaved with the (15, 9, b=3) code seven times to yield a (155, 93) code which corrects bursts of length 31 or less and a total of $3.713944764 \times 10^{16}$ error patterns for an improvement factor of 2.5×10^8. If, inplace of the (15, 9, b=3) code, we use the (15, 6, 2)b=3 code seven times, the resultant (155, 72) code corrects a total of $1.668218142 \times 10^{19}$

error patterns for an improvement factor of 1.13×10^{11}.

If the $(31, 20, 2)b=3$ code is interleaved five times, the resultant $(155, 100)$ code corrects bursts of length 15 or less and a total of $4.026509432 \times 10^{13}$ error patterns for an improvement factor of 2.72×10^{5}. These last three interleaved codes have rates considerably lower than the corresponding Reed-Solomon code.

These examples indicate that for Reed-Solomon codes with small t, it is possible to find interleaved codes that guarantee correction of more error patterns, although the rate of the interleaved code will tend to be less than that of the R.-S. code.

We emphasize that this comparison is made on the basis of strictly algebraic decoding. It is probable that the results may undergo large scale changes if a probabilistic scheme such as GMD decoding is applied to the Reed-Solomon code. GMD decoding could also be applied to the interleaved code but the complexity would be greater since decoders for different codes would have to be included.

For higher values of t, it is expected that the R.-S. codes will outperform the interleaved codes, since even for $t = 3$, $m = 5$, it was impossible to find better interleaved codes without reducing the rate of the interleaved code considerably.

The Reed-Solomon codes are the most powerful known class of multiple burst correcting codes and as SBEC codes, they are asymptotically optimal when interleaved. However, the decoding algorithm is much more complex than that of SBEC codes.

5.3 Biased Interleaving

It has long been recognized that a code which considers all symbols to be equally important is not always efficient for encoding data where the digits of the information sequence have exponentially weighted values. For example, consider the transmission of the

binary representation of an integer, M, whose value may vary from 0 to 2^a-1. Then

$$M = m_{a-1}2^{a-1} + m_{a-2}2^{a-2} + \ldots + m_1 2 + m_0$$

where $m_i = 0$ or 1. M can be represented by the coefficients, m_i, as

$$M = m_{a-1}m_{a-2}\ldots m_1 m_0$$

Clearly, an error in the position a-1 is far more crucial than an error in the position 0, and a code which ignores this fact may not be the best code to use.

Codes which give more protection to some symbols than others have been developed and discussed in [53-57] . These codes are generally called Unequal-Error-Protection (UEP) codes.

It is shown in [53,54] that, while it is not possible for a systematic cyclic code to be a UEP code, it is possible for non-systematic codes. In [55] , it is shown that if a UEP code is designed to give extra protection to one symbol only, then a BCH code which protects all symbols as much as the UEP code protects the most significant symbol will have a lower rate than the UEP code. For codes of the same rate, the UEP code provides more protection for the one significant bit but less for the ramaining bits than does the BCH code. In [57] , it is shown that, on a sampled-data transmission system, such a UEP code can give a smaller mean-square-error than a BCH code with the same rate.

This technique does not seem applicable to the encoding of messages where certain sections of the message are more important than others, since it does not seem possible to give extra protection to more than a few symbols per block.

However, there seems to be a definite need for a coding scheme that performs this function, since, in most messages, there are certain sections that are more critical than the rest. Of particular importance is the address, which may, in a typical military message,

consist of only two or three letters and a number. An error in any of these positions will result in incorrect delivery of the message.

Another important part of any message sent to a teleprinter is the carriage return-line feed function. A typical teletype message consists of blocks of about 60 alphanumeric characters, followed by a carriage return and a line feed. If an error occurs in the carriage return function, the next 60 characters will be lost, and, if the error occurs in the line feed function, the next 60 characters will print on top of the previous 60, resulting in a net loss of 120 characters.

For these reasons, it makes sense to give extra protection to these parts of the message.

One way of giving this added protection is through the technique we have called biased interleaving. In this technique, a random error correcting code is interleaved with a burst error correcting code in such a way that the bits of the random error correcting code are concentrated in the section of the message where errors will cause the most harm. This allows the correction of longer bursts in crucial areas, and also the correction of some burst and random error patterns in these areas, or some random errors there and burst errors of a shorter length elsewhere in the word.

A comparable technique has been developed by the University of Hawaii, and is used in their Aloha System [58]. Interleaving is not used in this system, but rather a double encoding technique is used, for error detection only, at present, although the system is designed so that error correction can be implemented at a later date.

In the Aloha System, information is transmitted in packets of 640 bits, or half-packets of 320 bits. Each packet or half-packet is preceded by a header of 32 identification and control bits. The header is encoded using a (48,32) code, and then the packet or

half-packet plus the encoded header is encoded with either a (704,688) code or a (384,368) code in systematic form. In this way, the header can be decoded first and, if errors are detected, a retransmission is initiated.

This encoding technique allows the detection of bursts of length 16 or less, or up to 5 random errors, in the header, and four random errors in the rest of the packet. Thus, the header has been given a great deal of added protection.

We will now show that similar protection can be given by using the technique of biased interleaving.

Example 5.14

Consider a channel where typically, in 400 bits, a burst of length 15 or less occurs, but where occasionally bursts of length 15 to 20 occur. The channel may also introduce occasional random errors.

Let us interleave the (131,119,b=5) code three times to form a (393,357,b=15) code. Then interleave two words from the (15,5,3) code in such a way that the symbols in positions (a+2lb; a=1,2,3; b=0,1,2,3,4) belong to one word and the symbols in positions (c+2ld; c=4,5,6; d=0,1,2,3,4) belong to the other word. The first 120 symbols of the interleaved word will look like this:

aaabbbccccdddddeeeeeaaabbbccccdddddeeeeeaaabbb

ccccddddeeeeeaaabbbccccddddeeeeeaaabbbccccc

dddddeeeeecccccddddeeeee

where a and b represent symbols from the two words of the (15,5,3) code and c,d, and e represent symbols from the three words of the (131,119,b=5) code.

This interleaved code can correct bursts of length 21 or less, that occur in the first 105 symbols, and bursts of length 15 or less that occur elsewhere in the codeword.

Given that a burst of length 15 occurred, commencing after position 90, this code will correct 30 of 90 or 33.3% of all single error patterns in the first 90 bits and 435 of 4005 or 10.86% of all double error patterns and 4060 of 117,480 or 3.46% of all triple error patterns that occur in the first 90 bits.

Note that the rate of the code is lowered from 0.908 to 0.868, or by only 0.04, by the addition of the random error correcting codes.

We see from this example, that considerable added protection may be obtained at the price of a small reduction in rate. Only a moderate increase in circuit complexity is required, in the form of a timing circuit, separate encoders and a multiplexer.

If a long message were to be sent, the first codeword would contain the random error correcting codes to protect the header, containing the address and perhaps some identification and control bits, while the remainder of the message would be encoded only with the burst correcting code. Thus the overall information rate of the message would asymptotically approach that of the fundamental code as the message length increased.

Now we will investigate the application of biased interleaving to solving the problem of carriage return and line feed errors in a standard teletype circuit.

A line of printing from a teletype consists of approximately 60 alphanumeric characters. Each character is transmitted, using the 5-level Murray code, as a sequence of five bits, preceded by a start pulse and followed by a stop pulse. These start and stop pulses are for synchronization only and need not be transmitted if synchronization can be maintained in some other way. Thus a line of printing

can be transmitted as $5 \times 60 = 300$ bits.

Example 5.15

Consider a teletype channel on which bursts of length five or less are typical, and on which random errors are frequently present. Random errors in the text of a message are relatively unimportant, as the reader can correct most of them himself. However, a random error in the carriage-return or line feed functions is a much more serious error.

The 300 bits of the line of printing could be encoded using the $(112, 100, b=5)$ code without interleaving. Thus, three code-words would be sent for each line of printing. Then the ten bits containing the carriage return and line feed commands could be encoded using the $(15, 5, 3)$ code interleaved to degree two. This would allow correction of bursts of length six or less, or any single, double or triple random error patterns, as well as many error patterns of weight four, five or six in this crucial area.

The entire line of printing, plus the control bits, (310 bits in all) would be transmitted as 376 bits, which gives an information rate of 0.83. Had no extra protection been given to the control bits, the information rate would have been 0.895. This rate would have to be lowered by some synchronization technique while the code with biased interleaving can be synchronized by searching for the words of the $(15, 5, 3)$ code

The techniques of these examples can be combined to give protection to both the header and the carriage return and line feed functions. The reduction in information rate would be small for any but very short messages.

We will now consider the transmission of the binary representation of integers. As has already been pointed out, the weights of the binary digits decrease exponentially, so that the higher order digits

are far more important than the lower order digits. This problem can be solved by using UEP codes, although the code will tend to have a low rate if extra protection is desired for several digits.

For example, a code [53] with nine information digits must have at least seven check bits if it is desired to four bits against two errors and the other five against one. The best known such code has nine check bits, so its rate is 0.5.

In the following example, we present an interleaving scheme which gives extra protection to 10 of 30 bits at a fairly high rate. We point out that the first 10 bits of a 30 bit binary number represent more than 99.9% of the value of the number.

Example 5.16

Consider a system which transmits integer numbers from 0 to $2^{30}-1$ (0 to approximately 1 billion). The binary representation of such numbers consists of 30 bits. Instead of encoding each number individually, we take the numbers 10 at a time and symbol interleave them. This gives us blocks of 300 bits, the first ten of which are twice as important as the second 10, and the second 10 twice as important as the third 10, etc.

We encode the first 100 bits of the block of 300 using the (15,10,1)b=2 BCH code interleaved 10 times, and the remaining 200 bits using the (112,100,b=5) code. The net rate of such a system is 0.802. With this encoding scheme, we can correct bursts of length 20 or less in the first 100 bits, plus all single random errors, and many other error patterns involving up to 10 random errors or 10 short bursts in this area, and bursts of length five or less in the remaining 200 bits.

The preceding has been solely to illustrate what can be done with biased interleaving. The actual form that would be used to combat

errors in a real situation would depend entirely on the situation and could only be obtained after careful analysis.

The codes so formed are not self-synchronizing, in that they will not correct synchronization errors. However, they can detect when synchronization is lost, by observing the number of corrections made. If this number is consistantly high, there is a high probability that synchronization has been lost. Then the decoder will start shifting the received sequence until it finds a position where it can decode the random error correcting code with few or no errors. The probability is high that at this point, synchronization has been recovered.

So, with biased interleaving, we can obtain an aid to synchronization as well as added protection for important sections of a message, with little decrease in rate and a small increase in complexity.

Chapter 6. DECODING OF INTERLEAVED CODES

The previous chapters have dealt with the construction of interleaved codes and their error correcting capabilities.

This chapter will deal with the special decoding techniques that are necessary when the codeword to be decoded is made up of more than one component word. Emphasis will be placed on the simplicity with which the codeword can be broken down into its component codewords, rather than on the decoding of the component words themselves.

6.1 Decoding of Interleaved Random Error Correcting Codes.

Codes formed by interleaving random error correcting codes can be decoded quite simply by using a timing circuit to separate the various words and then having a set of small decoders working in parallel to decode them. The separation can be done as the word is received, so that no time need be lost in this operation. Then the total time for decoding is the time taken to decode one of the component words.

A block diagram of the decoder for a $(\mu n, \mu k)$ code is shown in figure 6.1.

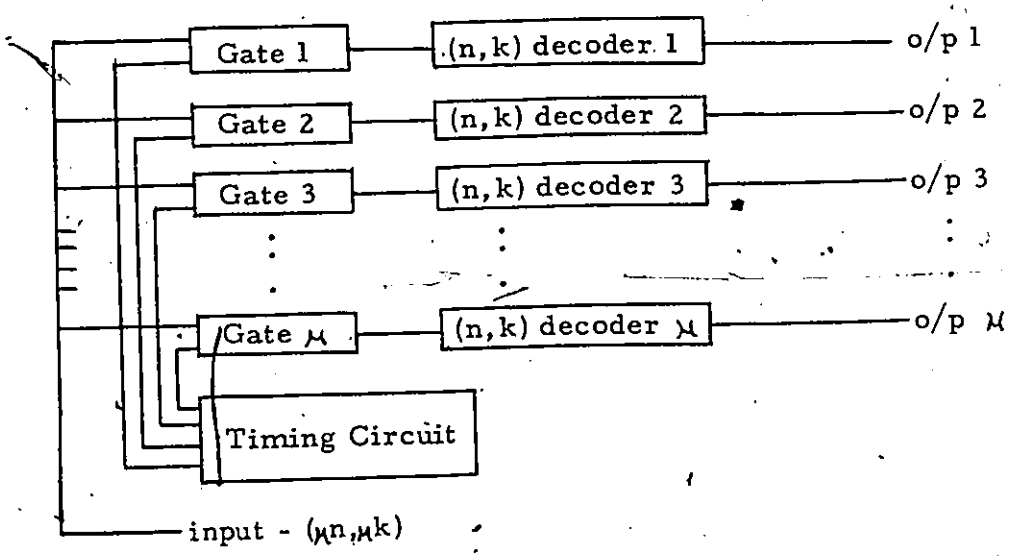

Figure 6.1 Decoder for a $(\mu n, \mu k)$ interleaved code.

Actually, this decoder is applicable to any interleaved code. The timing circuit and the individual decoders will vary as the structure of the interleaved code varies.

For the case of interleaved random error correcting codes, which are symbol interleaved, the timing circuit could consist simply of a μ-stage shift register, with its output connected to its input, and containing a single "one", being clocked at the bit rate. As the "one" is cycled around the shift register, it would enable the gates in succession, so that the $(i+j\mu)$th bit, $i = 1, 2, \ldots, n$; $j = 0, 1, \ldots, \mu-1$, would pass through the i'th gate and into the i'th decoder.

If the random error correcting component code is one-step majority logic decodable, the last information bit would leave the decoder k time units after the reception of the last bit of the word.

If, instead of an interleaved code, an (N,K,b) burst error correcting code had been used, with $N = \mu n$, $K = \mu k$, and $b = \mu t$, and the decoding procedure of error trapping described in [59] was used, the final information bit would not leave the decoder until K units of time after the reception of the last bit of the word.

We note that the saving in time achieved by the parallel decoding of the interleaved code is an advantage only if the information can be used in a parallel form. Otherwise the information must be re-serialized, and the saving is lost.

6.2 Decoding of Interleaved Burst Error Correcting Codes.

Interleaved burst error correcting codes can be separated into their component codes in exactly the same fashion by the circuit of figure 6.1. The (n,k) decoders would then be error trapping decoders for short burst correcting codes, and the timing circuit would be exactly the same if the code was symbol interleaved. If the code was block interleaved, the timing circuit would be a shift register of length μb, where b is the burst correcting ability of the component code.

Again the shift register would contain a single "one", but this time the input gates would be enabled by the system shown in figure 6.2.

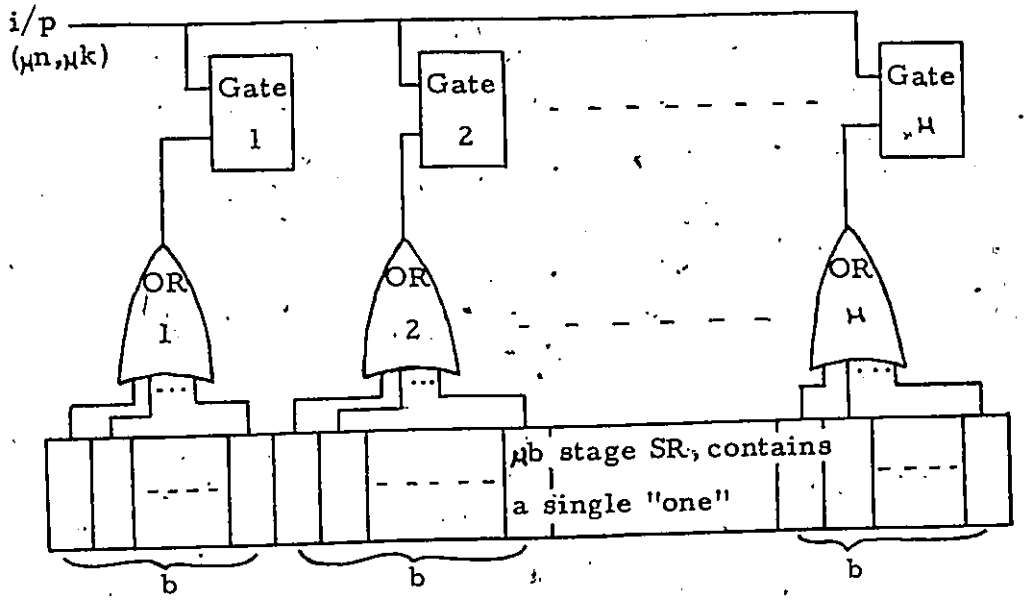

Figure 6.2 Timing and gate enabling circuit for block interleaved burst error correcting codes.

Using this system, gate 1 is enabled for b successive bits, then gate 2, and so on until gate μ has been enabled for b successive bits. Then the "one" is shifted to the first stage of the register, and gate 1 is enabled again. In this way, the component codewords are separated as they are received and can be decoded independently and in parallel. Thus the total decoding time will be μ times less than that required for an SBEC code with the same parameters. Again, the saving in time is an advantage only if the information can be used in parallel.

An advantage common to both interleaved random and burst error correcting codes is that the total time spent in the decoder is reduced. This means that the processing speed does not have to greatly exceed the transmission speed, or that only a small buffer storage need be provided.

6.3 Decoding of Codes Formed by Interleaving Random Error Correcting Codes with SBEC Codes.

Interleaved codes which have both random and burst error correcting component codes can be decoded by the circuit of figure 6.1, with the timing circuit of figure 6.2. The only difference would be that the OR gates would not all have the same number of inputs, if the random error correcting code was block interleaved, or that some of the OR gates could be left out if it was symbol interleaved.

The time taken for decoding would be the maximum of the times for decoding the various component codes. The decoding of the interleaved code will be faster than the decoding of an equivalent SBEC code only if all the component codes can be decoded in less time than that required for the equivalent SBEC code.

Certainly a short SBEC code can be decoded faster than a long SBEC code. Hence, the only question to be answered is this. Can a short random error correcting code be decoded faster than a long SBEC code?

The answer to this question depends on the relative lengths of the codes, the number of errors that the random error correcting code can correct, and the decoding technique applied to the random error correcting code.

A very short random error correcting code can be decoded in fewer steps than a very long SBEC code regardless of the decoding technique used. Also, a random error correcting code that is majority logic decodable can be decoded in fewer steps than an SBEC code as long as the random error correcting code contains fewer information symbols than the SBEC code.

If some algebraic decoding procedure has to be applied to the random error correcting code then the decoding may require a large number of steps, and any SBEC code except a very long one might be decoded in fewer steps.

Bearing in mind the characteristics of the interleaved codes formed in section 5.1 and the preceding remarks, we can conclude that in cases where a simple decoding technique, such as majority logic decoding, can be applied to the random error correcting component code, or where the degree of interleaving is fairly high, so that the component word length is substantially less than that of the interleaved word, the interleaved code can be decoded in fewer steps than an SBEC code with the same parameters.

This same procedure can be applied to codes formed by biased interleaving, with slight modifications of the timing circuit. In general, where codes formed by biased interleaving are used, the random error correcting code used will be very much shorter than the interleaved codeword, and hence the decoding time will not be affected by the interleaving; or, as in example 5.14, if a code is formed by biased interleaving a random error correcting code with an interleaved burst error correcting code, then the decoding time will be reduced, compared to that of an SBEC code with the same parameters.

Chapter 7. CONCLUDING REMARKS.

In this thesis, we have discussed the problem of errors on a digital communication system. We modelled the communication channel in four different ways, and showed where each model is useful. The Gilbert model for the compound channel was the best model for a typical communication channel.

We then discussed the various types of error correcting codes which are available for dealing with channel errors and pointed out that there exists no well understood class of codes that performs well on the compound channel.

As a solution to this problem, we investigated the technique of interleaving. We studied interleaved random error correcting codes and interleaved burst error correcting codes, and then considered interleaving these two types of codes with each other to form codes that would correct burst and random errors simultaneously.

The resulting codes perform very well when compared to SBEC codes. They seem to be superior to product codes, concatenated codes and Reed-Solomon codes, in terms of the number of correctable error patterns, although a more detailed comparison is required before definite results can be stated. In particular, the comparison was not performed with probabilistic decoding techniques applied to the non-interleaved codes.

The concept of biased interleaving was introduced, and it was shown that considerable added protection could be given to any desired section of a message by concentrating the bits of a random error correcting code in that section. This technique can be used as an aid to synchronization as well.

We then showed that the interleaved codes can be separated into component codes and decoded in parallel, and that a net

reduction in decoding time should result.

Further work in this area should include a more detailed comparison of interleaved codes with product codes, concatenated codes, and especially with Reed-Solomon codes, as these seem to be the only class of reasonable rate codes that can compete with the interleaved codes. Also, more consideration should be given to the synchronization problem, and the amount of assistance in synchronization that can be obtained by the interleaving of different types of codes.

REFERENCES

[1] R.W.Lucky, J. Salz and E.J. Weldon, Jr., 'Principles of Data Communications', McGraw-Hill, New York, 1968, p. 1.

[2] M. Eleccion, 'A/D and D/A Convertors', IEEE Spectrum, vol. 9, No. 7, July 1972, p. 63.

[3] Lucky, Salz and Weldon, ibid, pp. 15-17.

[4] W.W. Peterson and E.J. Weldon, Jr., 'Error Correcting Codes', MIT Press, Cambridge, Mass., 1972, p. 357.

[5] Lucky, Salz and Weldon, ibid, p. 15.

[6] A. Papoulis, 'Probability, Random Variables, and Stochastic Processes', McGraw-Hill, New York, 1965, p. 104.

[7] E.N. Gilbert, 'Capacity of a Burst-Noise Channel', BSTJ, vol. 39, 1960, p. 1253.

[8] G.D. Forney, Jr., 'Coding and its Application in Space Communications', IEEE Spectrum, June 1970, pp. 47-58.

[9] W.W. Wu, 'Applications of Error-Correcting Techniques to Satellite Communications', COMSAT Technical Review, vol. 1, No. 1, Fall, 1971, pp. 183-219.

[10] K. Brayer, 'Error-Correction Code Performance on HF, Troposcatter and Satellite Channels', IEEE Trans. Commun. Technol. vol. COM-19, Oct. 1971, pp. 781-789.

[11] E.R. Cacciamani, 'The SPADE System as Applied to Data Communication and Small Earth Station Operation', COMSAT Tech. Review, vol. 1, No. 1, Fall, 1971, pp. 171-181.

[12] Peterson and Weldon, ibid, pp. 5-15.

[13] Peterson and Weldon, ibid, chapters 5 and 8-11.

[14] A. Hocquenghem, 'Codes Correcteurs d'Erreurs', Chiffres(Paris), vol. 2, 1959, pp. 147-156.

[15] R.C. Bose and D.K. Ray-Chaudhuri, 'On a Class of Error Correcting Binary Group Codes', Inf. and Control, vol. 3, 1960, pp. 68-79.

[16] R.C. Bose and D.K. Ray-Chaudhuri, 'Furthur Results on Error Correcting Binary Group Codes', Inf. and Control, vol. 3, 1960, pp. 279-290.

[17] Peterson and Weldon, ibid, pp. 283-304.

[18] Peterson and Weldon, ibid, pp. 310-355.

[19] S.H. Reiger, 'Codes for the Correction of 'Clustered' Errors', IRE Trans., vol. IT-6, 1960, pp. 16-21.

[20] J.J. Stone, 'Multiple Burst Error Correction', Inf. and Control, vol. 4, 1961, pp. 324-331.

[21] S.E. Tavares and S.G.S. Shiva, 'Detecting and Correcting Multiple Bursts for Binary Cyclic Codes', IEEE Trans. Inform. Theory, vol. IT-16, 1970, pp. 643-644.

[22] P. Fire, 'A Class of Multiple-Error-Correcting Binary Codes for Non-Independent Errors', Sylvania Report RSL-E-2, Sylvania Reconnaissance Systems Laboratory, Mountain View, Cal.

[23] H.O. Burton, 'A Class of Asymptotically Optimal Burst-Correcting Block Codes', Presented at the International Communications Conference, Boulder, Col., 1969.

[24] Lucky, Salz and Weldon, ibid, p. 373.

[25] S. Lin, 'An Introduction to Error Correcting Codes', Prentice-Hall, Inc., Englewood Cliffs, N.J., 1970, p. 193.

[26] T. Kasami, 'Optimum Shortened Cyclic Codes for Burst-Error-Correction', IEEE Trans. Inform. Theory, vol. IT-9, April 1963, pp. 105-109.

[27] T. Kasami and S. Matoba, 'Some Efficient Shortened Cyclic Codes for Burst-Error-Correction', IEEE Trans. Inform. Theory, vol. IT-10, July 1964, pp. 252-253.

[28] A.J. Gross, 'Augmented Bose-Chaudhuri Codes Which Correct Single Bursts of Errors', IEEE Trans. Inform. Theory, vol. IT-9, 1963, p. 122.

[29] Peterson and Weldon, ibid, pp. 131-136

[30] E.R. Berlekamp, 'Algebraic Coding Theory', McGraw-Hill, New York, 1968, pp. 338-343.

[31] H.O. Burton and E.J. Weldon, Jr., 'Cyclic Product Codes', IEEE Trans. Inform. Theory, vol. IT-11, 1965, pp. 433-439.

[32] G.D. Forney, Jr., 'Concatenated Codes', Research Monograph No. 37, MIT Press, Cambridge, Mass., 1966.

[33] Peterson and Weldon, ibid, p. 371.

[34] H.T. Hsu, T. Kasami and R.T. Chien, 'Error-Correcting Codes for a Compound Channel', IEEE Trans. Inform. Theory, vol. IT-14, Jan. 1968, pp. 135-138.

[35] Peterson and Weldon, ibid, chapters 13-14.

[36] Peterson and Weldon, ibid, p. 401.

[37] Lucky, Salz and Weldon, ibid, pp. 382-392.

[38] S. Lin and H. Lyne, 'Some Results in Convolutional Code Generators', IEEE Trans. Inform. Theory, vol. IT-13, Jan. 1967, pp. 134-139.

[39] G.D. Forney, Jr., 'Review of Random Tree Codes', Codex Corporation Report, 1968.

[40] Peterson and Weldon, ibid, p. 427.

[41] E.R. Berlekamp, 'Note on Recurrent Codes', IEEE Trans. Inform. Theory, vol. IT-10, 1964, pp. 257-259.

[42] F.P. Preparata, 'Systematic Construction of Optimal Linear Recurrent Codes for Burst Error Correction', Calcolo, vol. 2, 1964, pp. 1-7.

[43] J.L. Massey, 'Implementation of Burst-Correcting Convolutional Codes', IEEE Trans. Inform. Theory, vol. IT-11, 1965, pp. 416-422.

[44] Y. Iwadare, 'Burst and Random Error Correction by Means of Threshold Decoding', Nippon Electric Co., Technical Report, 1968

[45] Lucky, Salz and Weldon, ibid, p. 405.

[46] A. Kohlenberg and G.D. Forney, Jr., 'Convolutional Coding for Channels with Memory', IEEE Trans. Inform. Theory, vol. IT-14, May 1968, pp. 618-626.

[47] Peterson and Weldon, ibid, p. 358.

[48] N. Abramson, 'Error Correcting Codes from Linear Sequential Networks', Presented at the Fourth London Symposium on Information Theory, Aug. 1960.

[49] Peterson and Weldon, ibid, p. 364.

[50] Peterson and Weldon, ibid, p. 111.

[51] S.M. Reddy and J.P. Robinson, 'Random Error and Burst Correction by Iterated Codes', IEEE Trans. Inform. Theory, vol. IT-18, Jan. 1972, pp. 182-185.

[52] G.D. Forney, Jr., 'Generalized Minimum Distance Decoding', IEEE Trans. Inform. Theory, vol. IT-12, Apr. 1966, pp. 125-131.

[53] B. Masnick and J. Wolf, 'On Linear Unequal Error Protection Codes', IEEE Trans. Inform. Theory, vol. IT-13, Oct. 1967, pp. 600-607.

[54] W.C. Gore and C.C. Kilgus, 'Cyclic Codes with Unequal Error Protection', IEEE Trans. Inform. Theory, vol. IT-17, Mar. 1971, pp. 214-215.

[55] C.C. Kilgus and W.C. Gore, 'A Class of Cyclic Unequal-Error-Protection Codes', IEEE Trans. Inform. Theory, vol. IT-18, Sept. 1972, pp. 687-690.

[56] D. Mandelbaum, 'Unequal Error Protection Codes Derived from Difference Sets', IEEE Trans. Inform. Theory, vol. IT-18, Sept. 1972, pp. 686-687.

[57] C.C. Kilgus and W.C. Gore, 'Root-Mean-Square Error in Encoded Digital Telemetry', IEEE Trans. Comm., vol. COM-20, pt. 1, June 1972, pp. 315-320.

[58] N. Abramson, 'The Aloha System', in 'Computer Communication Networks', N. Abramson and F. Kuo, eds., Prentice-Hall Inc., Englewood Cliffs, N.J., 1973, pp. 501-517.

[59] Peterson and Weldon, ibid, pp. 364-370.