EMBEDDED CODING ALGORITHMS APPLICABLE TO TIME VARIABLE CHANNELS

By:

Farshid Zolghadr, B.Sc., AMIEE

A thesis presented for the degree of Doctor of Philosophy in the Department of Engineering, University of Warwick

November 1989

To My Wife Shahnaz
My Daughter Ayeshah
and My Parents

TABLE OF CONTENTS

LIST OF TABLES AND ILLUSTRATIONS

ACKNOWLEDGMENTS

I would like to express my sincere gratitude to my supervisor, Dr. B. Honary, for his guidance, encouragement and valuable comments throughout this work. I would also like to thank Professor M. Darnell for his valuable comments during the course of this research.

I am also indebted to my colleagues in the Hull-Warwick Communications Research Group, without whose encouragements, helpful comments and friendship I would not have been able to complete this work.

Finally, I would like to take this opportunity to thank my parents and my wife for their support and continual encouragements.

DECLARATIONS

The following is a list of the materials which have either been published or submitted for publication, during the course of this research programme, along with the sections of the thesis to which they relate. These materials are the direct result of the work carried out by the author from the 1st of October 1985.

1) Zolghadr, F., Honary, B., Darnell, M., "Statistical Real Time Channel Evaluation (SRTCE) techniques using variable length T-codes", IEE Proc., Com., Speech and Vision, Vol.136, Pt.I, No.4, pp259-266, Aug. 1989.

Relevant thesis sections:- 3.1, 3.2, 3.3 and 3.4.

2) Zolghadr, F., "Preliminary Investigation of Embedded Array Encoding", Internal Report, Coventry Polytechnic, Jan. 1987

Relevant thesis sections:- 4.1, 4.2, 4.4 and Appendices A to D.

3) Darnell, M., Honary, B., Zolghadr, F.,: "Embedded coding techniques: principles and theoretical studies.", IEE Proc. -F, Comm. Radar and Signal Processing, Feb. 1988

Relevant thesis sections:- 4.1, 4.2, 4.4 and Appendices A to D.

4) Darnell, M., Honary, B., Zolghadr, F., "Embedded Array Coding For HF Channels, Theoretical & Practical Studies", Proc. of IMA Conf. on "Cryptography and Coding", Ed. (Beker, H. J.), Oxford Univ. Press, pp135-152, 1989.

Relevant thesis sections:- 4.3 and 4.4.

5) Zolghadr, F., Honary, B., Darnell, M.,: "Embedded Convolutional Coding.", Proceedings of the IERE, Fifth Int. Conf. on Digital Processing of Signals in Comm., Sept. 88.

Relevant thesis sections:- 5.1, 5.2, 5.3 and 5.4.

6) Zolghadr, F., "Reliable, Interference Resistance, Data Transmission Techniques for Multi-User HF Communications", Progress Report No.2 to Royal Aerospace Establishment (Farnborough) Contract No. 2119/037, Aug. 1988.

Relevant thesis sections:- 6.1, 6.2, 6.3 and 6.5.

7) Honary, B. Zolghadr, F., Darnell, M., "A Code-Assisted Bit Synchronisation (CABS) Scheme", Electronics Letters, Vol. 25, No. 23, 19th Jan. 1989.

Relevant thesis sections:- 6.3 and 6.5.

8) Zolghadr, F., "Reliable, Interference Resistance, Data Transmission Techniques for Multi-User HF Communications", Progress Report No.3 to Royal Aerospace Establishment (Farnborough) Contract No. 2119/037, Jan. 1989.

Relevant thesis sections:- 6.1, 6.3, 6.4, 6.5.

9) Honary, B., Zolghadr, F., Darnell, M., "Convolutional Encoding for Both Synchronisation and Protection Over HF Channels", IEE Colloquium on "Adaptive HF Management", London March 1989.

Relevant thesis sections:- 6.1, 6.3, 6.4, 6.5.

10) Zolghadr, F., "Reliable, Interference Resistance, Data Transmission Techniques for Multi-User HF Communications", Progress Report

No.4 to Royal Aerospace Establishment (Farnborough) Contract No. 2119/037, Apr. 1989.

Relevant thesis sections:-6.1, 6.2, 6.3, 6.4, 6.5, 6.6 and Appendices E and F.

11) Honary, B., Zolghadr, F., Darnell, M., Maundrell, M., "Multi-Functional MFSK Coding over Poor Narrow-Band HF Channels", IEE Proc. Part I, Submitted for publication Jun. 1989.

Relevant thesis sections:- 6.1, 6.2, 6.3, 6.4, 6.5, 6.6.

12) Zolghadr, F., "Reliable, Interference Resistance, Data Transmission Techniques for Multi-User HF Communications", Progress Report No.6 to Royal Aerospace Establishment (Farnborough) Contract No. 2119/037, September 1989.

Relevant thesis section:- 6.3.

13) Zolghadr, F., Honary, B., "Digital Matched Correlator Automatic Gain Control (Mc-AGC) Scheme Applicable to MFSK Modems", Submitted to IEE Proc. Part I, Oct. 1989.

Relevant thesis sections:- 6.3 and 7.2.

ABSTRACT

This thesis investigates new design and implementation techniques applicable to modern communication systems operating over time variable channels. Three areas of interest are investigated. These include, source coding in conjunction with real-time channel evaluation, channel coding and modem design.

An investigation of source coding methods has led to the development of a new embedded real time channel evaluation, based on statistical techniques. The performance of this technique is examined using simulation techniques for channels with and without memory

Existing channel coding schemes applicable to time variable channels have been examined. This led to the formulation of a new coding technique, termed embedded encoding. Two implementations of such codes, embedded array codes and embedded convolutional codes, were developed. The theoretical and practical performance of these codes has been investigated.

The final area of investigation has been the development of a 4-tone multi-frequency shift keying modem. In keeping with the intention of totally digital system design, the demodulator has been implemented on a single digital signal processing card. The demodulation method developed employs an embedded synchronisation technique, termed Code-Assisted Bit Synchronisation.

The demodulator performs symbol synchronisation by utilising the convolutional code used for the purpose of channel coding. It thus performs the combined functions of the demodulator, decoder and symbol timing recovery, which are normally found as separate subsystems. In combining these subsystems a more efficient modem has been developed.

CHAPTER 1

INTRODUCTION

Communication is the transfer of messages from a data source to a data sink. A message can take many physical forms e.g. sound, light, electrical, etc. In this thesis only electrical messages are considered. Electrical messages can be grouped into either analogue or discrete formats. With the recent advances in digital computers, the use of discrete messages have become increasingly more widespread. This increased demand, has resulted in extensive efforts being concentrated on the development of reliable and efficient communication systems.

The main attribute of many message streams is the inherent redundancy which exists within them. For efficient communication these messages should contain as little redundancy as possible. The removal of the message redundancy is referred to as source coding.

The transfer of messages involves the utilisation of various modules, i.e. modulator, demodulator and the communication medium. The modulator, which is employed at the transmitter, translates the digital message stream into a suitable format for transmission over the communication medium. There are many modulation techniques in use today. Basically, these techniques operate by varying some parameter of a constant amplitude sinusoidal signal (the carrier), in response to the message stream.

The demodulator, on the other hand, is situated at the receiver and performs the exact reverse operation to that of the modulator. The demodulation process involves the detection of the changes introduced by the modulator in the carrier signal and the subsequent mapping of these into the format of the original message stream. There are numerous methods of signal detection available, the choice being dictated by cost, complexity and the required performance.

The communication medium, usually referred to as the channel, is the means by which the message is conveyed to the receiver. In the context of this thesis, the channel, characterises a radio communication medium comprising, the radio frequency (RF) circuit, antennas and the ionosphere. In this medium, information is conveyed by the propagation of electromagnetic waves in space. The characteristics of the channel are controlled by the choice of the RF frequency, the property of the medium and the antenna. The term "a Time Variable Channel" describes a communication medium in which the effects of the channel vary with time.

The reliability and the efficiency of any communication system depends directly on the choice of the above system modules and the properties of the medium. The choice of the modulation and the demodulation methods have a direct influence on the resilience of the communication system towards the unwanted effects of the channel. These unwanted effects are the various distortions of the transmitted message, caused by the channel. These distortions manifest themselves in many physical forms, e.g. purely additive noise effects, phase and frequency instabilities, etc. The direct result of these distortions is the introduction of errors in the message stream delivered to the recipient of the information.

As a means to combat the effect of the errors caused by the channel, error control techniques are employed. Generally speaking, error control coding (channel coding) techniques are implemented by adding redundant information to the digital message before transmission. In practice, the digital message stream (i.e. the data) is divided into blocks on which a set of parity check calculations are performed. The result of this operation are the necessary redundant information digits which are added to each block of data.

A measure of the maximum amount of information that can be transmitted over a channel, is the channel capacity. The channel capacity is monotonic function of the received signal to noise ratio (SNR). Clearly, for time variable channels the channel capacity varies with time. Shannon has shown [Shannon 1948] that if the transmission rate does not exceed the channel capacity, it is possible to transmit coded information with an arbitrary small probability of error at the output of the decoder; however the block length must be allowed to increase without limit.

As a direct consequence of the Shannon's channel capacity theorem and the variable channel capacity experienced over time variable channels, enormous amount of effort has been spent on devising techniques for estimating the instantaneous channel status. These techniques are broadly referred to as Real-Time Channel Evaluation (RTCE) techniques. RTCE techniques allow appropriate action to be taken in response to the information obtained via continuous monitoring of the channel state.

This thesis concentrates on the design and implementation of new algorithms applicable to modern communication systems operating over time variable channels. The investigation considers three main areas of interest. These include, source coding in conjunction with RTCE, channel coding and modem design.

The second chapter of this thesis reviews some of the theoretical background relevant to this thesis. In the first section fundamentals of information theory are outlined. This is followed by the description of various forms of source coding techniques. Two of the most commonly used channel models are then discussed. The concept of channel coding in the context of block and convolutional codes are reviewed, and the principles of modem design for time variable channels are outlined.

In the third chapter the principle of a new RTCE technique is presented. This algorithm is an extension of source coding procedures widely used today. The technique known as statistical RTCE (SRTCE) [Zolghadr Honary Darnell 1989] is particularly applicable to channels were the effect of noise is severe. SRTCE is a unique form of RTCE, which can provide both channel error rate and error distribution characteristics.

Chapters four and five introduce a new concept in error control coding termed Embedded coding [Zolghadr 1987, Darnell Honary Zolghadr 1988, Zolghadr Honary Darnell 1988]. Chapter four concentrates on the theoretical and the simulation study of embedded block codes. Embedded block codes employ a combination of forward error correction and detection in an automatic repeat request (ARQ) environment. In chapter five, the concept of embedded coding is extended to convolutional codes. In embedded convolutional encoding, punctured convolutional codes are used for the purpose of error correction and code rate variation. Error detection is preformed via a new RTCE technique which is a by-product of the soft decision decoding of convolutional codes.

Chapter six of this thesis is devoted to the development of a new digital modem algorithm, which combines the functions of the demodulator, decoder and the symbol synchronisation subsystems. The developed algorithm is implemented using a digital signal processor (TMS320c25) to produce a complete modem operating over the High Frequency (HF) channel. The modem employs a novel digital processing procedure termed code-assisted bit synchronisation (CABS) [Zolghadr

1988, Zolghadr 1989a, Honary Zolghadr Darnell 1989, Honary Zolghadr Darnell Maundrell 1989]. Signal detection is achieved via a set of noncoherent correlators, while symbol synchronisation and error correction are performed by a soft-decision Viterbi decoder [Viterbi 1971].

Finally, chapter seven summarises and discusses the main results of the research, and suggests ideas for future investigation in this area.

The Appendices, A through to D, contain the main theoretical studies of the embedded coding, described in Chapter 4. The last two appendices present the assembly programs for the CABS modulator and demodulator. These have been enclosed since the actual implementation of the algorithm used is in itself a major task and would ease further research in this area.

CHAPTER 2

THEORETICAL BACKGROUND

2.1 Source Coding (theory)

In order to improve the efficiency of the communication process, source coding is applied to remove the redundancy of the message [Shannon 1948, Abramson 1963, Blahut 1987, Held 1987]. Source coding is achieved by examining the statistical nature of the symbols in the source alphabet. Generally speaking, the aim of source coding is to reduce the average number of symbols required to represent the source alphabet.

In order to gain a fuller understanding of the concepts of source coding, it is important to have some basic tools and principles for describing the source parameters. Consequently, in this Section the theoretical background of information theory, and the properties of some simple discrete sources are given. Finally, three examples of source coding are examined.

2.1.1 Discrete Sources

A discrete source generates a message sequence which consists of symbols from the source alphabet, i.e.

$$X_i \in \{ X_1, X_2, X_3, \ldots X_Q \} \qquad (2.1)$$

where X_i is a symbol and Q is the size of the alphabet set.

The efficiency of a discrete source is determined by the statistical properties of the message sequence, produced by it. Furthermore, these parameters classify the source into the following two groups:

a) memoryless sources [Abramson 1963 pp13, Gallager 1968 pp40];

b) sources with memory [Sklar 1988 pp599, Kanal Sastry 1978].

A memoryless source, outputs symbols which exhibit no intersymbol dependency. A good example of such a source is the random number generators, used for choosing the winning lottery ticket.

If the probability of the source symbols are known, a measure of information content can be assigned to each symbol:

$$I(X_i) = -\log_2 p_i \tag{2.2}$$

where $I(X_i)$ is the amount of information (in bits) associated with symbol X_i, with the probability of occurrence, p_i.

A measure of the average self information of a source, is the source entropy. The source entropy, H(X), can be viewed as the average amount of uncertainty associated with the output of a source. For a memoryless source, the source entropy is defined as:

$$H(X) = E\{I(X_i)\} = -\sum_{i=1}^{Q} p_i.\log_2 p_i \tag{2.3}$$

where E{X} is the expected value of X.
It is clear from (2.3) that a source has maximum entropy if all the source symbols have equal probability of occurrence. Moreover, for a memoryless source, the source entropy can bounded by [Blahut, 1987, pp56-57]:

$$0 \leq H(X) \leq \log_2 Q. \tag{2.4}$$

The source symbols probabilities of occurrence, could in theory be conditionally dependent on the other symbols. Such sources are known as sources with memory. An example of this type of source is a source producing a stream of English text, in which there is a strong intersymbol dependency. The presence of intersymbol dependency severely restricts the efficiency of the source. This can be best demonstrated by the occurrence of the letter 'q' in a message of the English text. In this example, the recipient of the message will (with a high probability) expect the next letter in the message to be 'u'. Consequently, the transmission of the letter 'u' in this instance does not convey very much information.

A measure of information content of sources with memory can be obtained by generalising equation (2.2). As an example, for a one dimensional source (i.e. a source with one symbol dependency), (2.2) can be written as [Abramson, 1963, p22]:

$$I(X_i|X_j) = -\log_2 P(X_i|X_j) \qquad (2.5)$$

where $X_i|X_j$ implies the symbol X_i conditional on the symbol X_j and $P(x)$ is the probability of x.

The source entropy in this example can be obtained by extending equation (2.3), which yields [Abramson, 1963, p35]

$$H(X) = E\{I(X_i|X_j)\} = -\sum_{i=1}^{q} \sum_{j=1}^{q} P(X_i,X_j).\log_2 P(X_i|X_j) \qquad (2.6)$$

where $P(X_i,X_j)$ is the probability of symbol X_i following the symbol X_j.

An examination of equations (2.3) and (2.6) shows that the entropy of a source with memory is lower than that of a memoryless source with the same alphabet and symbol probability [Sklar, 1988, p598]. Clearly, a source with maximum entropy conveys maximum amount of information; in other words, unpredictability is the key to efficient data transfer. In the following Sections, three examples of source coding techniques are described.

2.1.2 Huffman Coding

Huffman codes are a form of variable-length codes which can achieve the shortest average code length for a given input alphabet [Huffman 1952]. In Huffman coding, each symbol in the source alphabet is assigned to a codeword (usually binary). The length of the codeword is chosen such that the information contents of the codeword and the corresponding symbol are equivalent.

The coding process [Held 1987 pp98-100] is of a tree structure which is formed by listing the source symbols, along with their probabilities, in a descending order of occurrence. These symbols form the root of the tree with their probabilities labelling the branches. The two entries with the lowest probability are merged to form a new node with their composite probability. The pair formed in this manner is moved up in the list, to ensure the descending probability structure is preserved. The new pair however, is given a higher priority in the reordering process and is placed above the entries with the same probability. The above process is repeated until the top of the tree structure is reached.

The two branches merging into a given node are arbitrary labelled with 0's and 1's. To find the codeword representing a symbol X_i, the path from the top of the tree is traced to the root of the tree marked by the symbol X_i. The binary digits which label the branches of the path, are then read out as the codeword for symbol X_i.

A measure of effectiveness of a source coding technique is the compression ratio. This is defined as the ratio of the average number of bits per symbol before and after coding [Sklar 1988 pp657]. When applying the Huffman coding scheme to memoryless sources, it is possible to achieve high compression ratios. In practice however, since some degree of conditionality exists between the source symbols, the level of compression achievable is reduced.

As mentioned in Section 2.2, the intersymbol dependency, reduces the source entropy. For a m-dimensional source, the effect of intersymbol dependency can be removed by providing the source with a new alphabet consisting of all the m-tuples of the original symbols. The Huffman coding process described above could then be applied to the m-tuple symbols in the new alphabet. This method of source coding is known as extended Huffman coding [Abramson 1963 pp86-87, Blahut 1987 pp69-71]. In this manner, higher compression ratio can be obtained.

2.1.3 Run-Length Codes

The output of many sources is a sequence of repeated character sequences, which result in a reduction in the source entropy. In order to increase the efficiency of the source, the effect of these long runs should be minimised. An example of this type of sources is the facsimile machines [Usubuchi et al. 1980]. In facsimile machines the scanned image is converted into a series of binary digits 0's and 1's (representing black or white levels). The final two-dimensional array formed in this manner, therefore, consists of many runs of repeated bits. The function of source coding is thus to compress the overall number of bits required to fully describe the scanned image.

Run-length codes are a class of source codes which are highly suited to this form of messages. These codes operate by making an appropriate substitution for lengthy runs of a given symbol. The substitution consists of a "start of compression indicator" followed by the number of times the character should be repeated and finally the repeated character itself. In this manner the need for transmission of all the repeated symbols in the run is eliminated. It is clear that, this technique can greatly improve the source entropy if long runs of repeated symbols are present.

2.1.4 T-codes: Methods of Code Construction

T-codes are variable-length codes which exhibit compression ratios comparable with Huffman codes. The synthesis of T-codes has been outlined extensively in [Titchener 1985, Titchener 1986]; and only a brief review is given here. There are two basic methods of code generation:

(a) via the augmentation algorithm,

(b) via the depletion algorithm.

In the augmentation algorithm the following steps are used:

(i) the complete initial character set is listed (set S^0), ie in the binary case the initial character set is (0,1);

(ii) this initial list is repeated;

(iii) a character from the first half of the list is allocated as a prefix to all the elements in the repeated second half of the list.

With the repeated application of the above procedure, Q times, an augmented set of degree Q is obtained (set S^Q). This is demonstrated in Table 2.1 using a binary base set for an augmentation of degree 3. It should be noted that an augmented binary set of Qth degree always contains (2^Q+1) characters.

S^0	S^1	S^2	S^3
0	~~0~~		
1	1	1	~~1~~
	00	~~00~~	
	01	01	01
		001	001
		0000	0000
		0001	0001
			11
			101
			1001
			10000
			10001

Table 2.1 T-code generation, for augmentation degree of 3 using method (a).

Although this method of code generation is very simple in principle, any practical generating process will suffer from the difficulty of having to store variable-length code words in fixed-length registers. It will therefore be necessary to use $(2^{Q+1}+2)$ memory locations for storing the (2^Q+1) code words and the (2^Q+1)-length variables.

In order to minimise the required memory locations during the code generation and encoding/decoding procedure, the depletion algorithm was proposed. This algorithm involves the following steps:

(i) a complete list of binary codes of length (Q+k) bits, is arranged in numerically ascending order, where Q is the degree of augmentation and k is the number of bits in the original base set -referred to as the "literal code";

(ii) the codes in this set are grouped into $(2^{r-1}.n)$ codes, where r corresponds to the rth iterative cycle and n is the number of elements in the base set (when grouping the codes, previously depleted entries are counted as if still present);

(iii) the set is depleted by a single deletion at an arbitrary, but corresponding, position in the first and each subsequent alternate group; the deleted code in the first group is referred to as a "prefix-depletion code".

The steps (ii) & (iii) are repeated for r=1,2 ..Q, the result being a set of $[2^{Q}(n-1)+1]$ binary depletion code words. Table 2.2 shows the process for a third-order depletion code with a single bit binary literal code; here Q=3, k=1 and n=2. These code words can be directly translated to the augmented code words using the algorithm detailed below.

S	S(0)	S(0,1)	S(0,1,2)
~~0000~~	-	-	-
0001	~~0001~~	-	-
0010	0010	~~0010~~	-
0011	0011	0011	0011
~~0100~~	-	-	-
0101	0101	0101	0101
0110	0110	0110	0110
0111	0111	0111	0111
~~1000~~	-	-	-
1001	~~1001~~	-	-
1010	1010	1010	1010
1011	1011	1011	1011
~~1100~~	-	-	-
1101	1101	1101	1101
1110	1110	1110	1110
1111	1111	1111	1111

Table 2.2 Binary depletion code generation, for augmentation degree of 3, using the method (b).

(a) The prefix-depletion codes are labelled depending on their association with the iterative cycle, i.e. the first prefix-depletion code corresponds to the first iterative cycle, etc.

(b) The prefix-depletion code words are examined individually; the first k bits corresponding to the literal code are left unchanged; the next Q bits following the literal code are referred to as the "prefix-selection code". In the process of translation of the prefix-depletion codes to prefixes, these bits will be replaced by the previous variable-length prefixes which, along with the literal code, will form the prefix. Each bit in the prefix-selection code points to a specific prefix. If the bit is equal to one, the corresponding prefix is included and, if the bit is equal to zero, it is not. It should be noted that, for each new prefix, only the previously determined prefixes are required.

(c) Once all the prefixes have been determined, the translation of the binary depletion codes into augmented variable-length T-codes can be carried out. This process is very similar to step (b) above. The binary depletion code words are examined individually and the first k bits (the literal code) are left unchanged. The remaining Q bits (the prefix selection code) are examined, starting with the bit immediately following the literal code. If this following bit is equal to 1, the first prefix (found in the previous step above) is included: otherwise it is not. This process is repeated for the remaining bits (and their corresponding prefixes). Tables 2.3 and 2.4 show the prefix generation and the T-code generation processes respectively for the example of Table 2.2, where S(i,..,j) correspond to the depletion code set with prefix depletion codes i,..,j.

Prefix depletion codes	Prefixes
0000	0
0001	1
0010	00

Table 2.3 Prefix depletion codes and their corresponding prefixes.

Binary depletion codes	**T-codes**
0011	01
0101	11
0110	100
0111	101
1010	0000
1011	0001
1101	0011
1110	00100
1111	00101

Table 2.4 Binary depletion codes and their corresponding T-codes.

Apart from the good compression capabilities of these codes, T-codes enable the decoder to synchronise automatically following any disturbance. This characteristic can best be illustrated by an example. Assume the sentence, "The city of Coventry", is to be transmitted using the T-code set {1,3,7,15,10,6,8} [Zolghadr Honary Darnell 1989]. The encoded data stream is shown in Fig.2.1, along with the transmitted text.

```
1111111111110 1111010 10 0 111111110 1111110 110 1111111110 0 11100
    T         h     e      c      i    t     y         o

1111101110 0 1110111111110 11100 11111011110 10 111100 110 1110110
    f           C       o       v      e   n   t     r

1111111110
    y
```

Fig.2.1 Example text and its equivalent encoded data stream.

The received data stream is shown in Fig.2.2, where a single digit error has corrupted the third bit in the second transmitted symbol -as indicated by the symbol '*'. Although this error causes a loss in synchronisation, the decoder resumes synchronisation after three symbols and, in fact, only one symbol is actually lost.

```
               *
111111111110 110 10 10 10 0 11111110 1111110 110 111111110 0 11100
     T       / t  e  e / e       c       i     t      y         o
              Loss of
               sync.

1111101110 0 1110111111110 11100 1111011110 10 111100 110 1110110
     f             C         o        v       e    n    t     r

1111111110
    y
```

Fig.2.2 Example of automatic synchronisation of T code, S(1,3,7,15,10,6,8).

2.2 Channel Models (theory)

The communication channel causes many forms of degradation of the transmitted signal, ie distortion, noise and interference. These effects are complex probabilistic processes, thus it is difficult, if not impossible, to predict the behaviour of these distortions. The combined effect of these degradations leads to the corruption of the transmitted signal, which in turn results in errors at the receiver.

One of the main aims of any communication system is to reduce the effect of these errors, by utilising reliable communication processes. The design of these schemes, however, relies on the availability of some knowledge of the channel. Consequently, it is highly desirable to have some method of analysing the structure of the error generating mechanism of channels.

The error generating process of a digital communication channel can be viewed as a discrete-time stochastic process [Kanal Sastry 1978]. The output of the process is a sequence of error patterns which are assumed to be statistically independent of the transmitted data sequence.

Broadly speaking, channels can be characterised into two main categories:

a) memoryless channels;

b) channels with memory.

Although real channels are a mixture of the above, generally, one dominates over the other. In the following two Sections these two classes are discussed.

2.2.1 Discrete Memoryless Channels

A discrete memoryless channel (DMC) is characterised by a discrete input alphabet, a discrete output alphabet and a set of conditional probabilities which govern the mapping of the channel input and output sequences. Furthermore, each output symbol of the channel depends only on the corresponding input.

For a binary DMC the input and output symbols of the channel are binary. The error patterns formed by the channel can be represented as a binary pattern, which are added to the channel input symbols under GF(2), where GF(X) is the Galois field with x elements. The main characteristic of the memoryless channels is that the error mechanism effects each channel input symbol independently. Moreover, the occurrence of errors is not dependent on the previous errors. In order to characterise this type of channels, a two state Markov process can be used (see Fig.2.3).

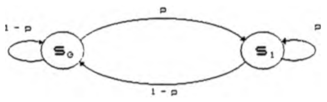


Fig.2.3 Two state model of a memoryless channel.

The model is composed of two states, S_0 and S_1. At each time interval, the model makes a transition from state Z_{n-1} to state Z_n, where n is the time reference. The actual transitions are governed by the state transition probabilities. The output of the model, Ω_n, is formed by mapping the current state of the model, Z_n, to a discrete output alphabet. For a binary DMC, the following mapping could be used:

$$\begin{aligned} Z_n &= S_0 \quad \longrightarrow \quad Q_n = 0 \\ Z_n &= S_1 \quad \longrightarrow \quad Q_n = 1 \end{aligned} \qquad (2.7)$$

where $Q_n = 1$ is a channel error event.

The error generation process of Fig.2.3 is completely described by the following model parameter [Zolghadr Honary Darnell 1989]:

$$P(Z_n = S_1) = p \qquad (2.8)$$

where P(x) is the probability of occurrence of the event, x.

It is evident that the probability of error for this channel is given by equation (2.8). Examples of actual channels which can be closely approximated by this model, include those which are predominately disturbed by Gaussian noise.

The capacity of a channel, C, is given as [see Kanal Sastry 1978]

$$C = 1 - H_e \quad \text{bits/symbol} \qquad (2.9)$$

where H_e is the average conditional entropy of a typical error sequence of the channel. For a DMC with error rate, p, H_e is the unconditional entropy and is given as [Kanal Sastry 1978 , (see also equation (2.3))]

$$H_e(\text{DMC}) = -p.\log_2 p - (1-p).\log_2 (1-p) \qquad (2.10)$$

As discussed in Section 2.1, the basic property of entropy is that it increases as the memory of the process decreases. Consequently, for a DMC with bit error rate (BER), p, the channel capacity is lower than that of a channel with memory with the same BER.

2.2.2 Channels with Memory

For a channel with memory, the occurrence of an error increases the probability of a further error event. In this manner bursts of errors occur which are separated by relatively error free intervals. An example of such channels is the High Frequency (HF) skywave channel.

A simple representation of channels with memory is the Gilbert model [Gilbert 1960], (see Fig.2.4).

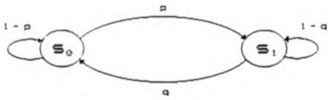


Fig.2.4 Gilbert's model

This model is very similar in structure to the model of a DMC (Fig.2.3). However, this model is now fully described by the following two parameters [Zolghadr Honary Darnell 1989, Kanal Sastry 1978]:

$$P(Z_n = S_0 | Z_{n-1} = S_1) = q \qquad (2.11)$$

$$P(Z_n = S_1 | Z_{n-1} = S_0) = p \qquad (2.12)$$

with

$$P(Z_n = S_1) = p/(p+q) \qquad (2.13)$$

$$P(Z_n = S_0) = q/(p+q) \qquad (2.14)$$

It is clear that for the model to be memoryless the equation pairs (2.11)&(2.14) and (2.12)&(2.13) must be equivalent (ie (p+q) must be equal to one).

The Gilbert model, like the DMC model, consists of a "good state", S_0 and a "bad state" S_1. No errors are produced in the good state; in the bad state bursts of errors are generated. The output of the model, Q_n, is formed by mapping the current state of the model, Z_n, to a discrete output alphabet. The mapping process is however, probabilistic in which the generation of errors in the bad state is governed by the parameter ,h, ie :

$$Z_n = S_0 \longrightarrow Q_n = 0 \qquad (2.15)$$

$$Z_n = S_1 \begin{cases} \longrightarrow Q_n = 1 & \text{probability } h \\ \longrightarrow Q_n = 0 & \text{probability } (1-h) \end{cases} \qquad (2.16)$$

where $Q_n = 1$ is a channel error event.

The BER for this channel, $P(Q_n = 1)$, can be evaluated from equations (2.13)&(2.16)

$$P(Q_n = 1) = h.p/(p+q) \qquad (2.17)$$

It is clear (see equation 2.9 and 2.6) that the capacity of the Gilbert channel is higher than that of the DMC. This increase in capacity, which is the characteristic of all channels with memory, is an important factor which motivates work in channel modelling.

2.3 Channel Coding (theory)

In this section, the error control techniques used for removing the channel errors are introduced. Shannon has shown that [Shannon 1948] it is theoretically possible to communicate with very low probabilities of error, even in the presence of noise. However, the practical achievement of this statement has proved rather difficult. Shannon's theorem implies that, if a source with entropy, H, is transmitting data over a channel with capacity C, the BER approaches zero as long as $H \le C$. In principle, channel coding is concerned with the controlled addition of redundant data or parity check digits into the transmitted digits in an attempt to satisfy this condition.

The effectiveness of a coding scheme is measured by how closely it matches the entropy of the coded data, to the channel capacity available. Two main classes of error control codes are briefly described here:

a) block codes;

b) convolutional codes;

2.3.1 Block Codes

Block codes are a class of error control codes, which consist of a closed set of codewords, where each codeword is of length n with k information digits (if linear). Block codes can be divided into two main groups, linear and nonlinear. Linear block codes, form a sub-space of the vector space V_n of all n-tuples, over a finite field GF(q), where q is a power of a prime number (q=2 for the binary case) [Peterson Weldon 1972]. Nonlinear block codes, on the other hand are generated by means of some nonlinear operators (e.g. NAND gates etc.). In this dissertation only linear codes are discussed.

In block coding, codewords are formed by the addition of, c, redundant digits to every, k, input data digits. In this manner, a code with block length, n (n = k + c), consists of a set of 2^k codewords. The set formed in this way is referred to as a code book. A code is said to be systematic if the k data digits appear explicitly in the codeword. A measure of the added redundancy of a code is the code rate, R, where R=k/n.

Linear codes can be characterised by a generator matrix, G. The generator matrix is formed by placing k basis vectors as rows of the k x n matrix. Basis vectors are a set k linearly independent vectors which span an n-dimensional vector space. For a given code, all the codewords can be generated by taking all the 2^k linear combinations of the basis vectors.

The generator matrix is normally used in the Standard Echelon (SE) form, defined by the linear combination of rows or by rows interchange [Peterson Weldon 1972]. The SE form of the generator matrix has the following structure:

$$[G] = [I : g] \tag{2.18}$$

where I is the k x k identity matrix and g is a k x (n-k) matrix. For systematic codes, the encoding procedure can be viewed as

$$[X] = [K][G] \tag{2.19}$$

where [X] is the resulting 1 x n code vector and [K] is the 1 x k data vector.

One of the important properties of linear block codes is that the addition (under appropriate GF) of two codewords results in a valid codeword. This implies that if a code is used for error detection, in order to detect the presence of errors, the error pattern must not be equivalent to a valid codeword. Consequently, this property sets the maximum number of errors that an error detection code can tolerate.

In general, the error control capability of a code is determined by the Hamming distance of the code. For a linear block code the Hamming distance, d, is defined as the smallest non-zero weight of a codeword in the code.

The error control capability of a code is directly determined by the Hamming distance of the code. The error correction capability of the code is given by

$$t \leq \left\lfloor \frac{d-1}{2} \right\rfloor \tag{2.20}$$

where t is the error correction capability of the code (in digits) and $\lfloor x \rfloor$ implies the integer part of x;

and the number of errors, e, the code can detect is given by

$$e \leq d - 1 \tag{2.21}$$

2.3.2 Decoding of Linear Block Codes

Most of the decoding methods are based on the calculation of the syndrome vector [MacWilliams Sloane 1977, Michelson Levesque 1985 pp63-69]. The evaluation of the syndrome vector relies on the availability of the parity check matrix. This matrix has dimensions (n-k)xn, with the property that when multiplied by a valid codeword vector, [X], gives:

$$[X]\ [H]^T = [0] \tag{2.22}$$

where $[H]^T$ is the transpose of [H] and [0] is the 1x(n-k) null vector.

The set of all possible codewords is called the null space of the matrix [H] [Peterson Weldon 1972]. [H] is therefore defined as the matrix which has the code book as its null space. In other words, a multiplication of a codeword with the transpose of the parity check matrix yields an all zero vector. This property of the [H] matrix is important for decoding block codes.

Stating equation (2.22) in a different way, it implies that a 1 x n vector [C] is a codeword if and only if

$$[C]\ [H]^T = [0] \tag{2.23}$$

Now, if [C] is the received version of a transmitted codeword [X], and the vector [E] represents the "error" vector, then

$$[C] = [X] + [E] \tag{2.24}$$

where the addition is performed under GF[2].

The decoding process would then involve the evaluation of equation (2.23). The result of this operation is referred to as the syndrome vector [S], where [S] is given by

$$[S] = [C]\ [H]^T = ([X] + [E])\ [H]^T \tag{2.25}$$

$$[S] = [X]\ [H]^T + [E]\ [H]^T \tag{2.26}$$

and combining equations (2.23) and (2.26) gives

$$[S] = [E]\ [H]^T \tag{2.27}$$

where, again all the operations are performed under GF[2].

From equation (2.27) it is evident that the actual value of the syndrome vector is only dependent on the error vector. This information is used as the sole means by which error detection is performed by a decoder. Once the syndrome vector has been evaluated, its value would indicate the validity of the received codeword. If [S] = [0], it is assumed that no errors have occurred. On the other hand a non-zero syndrome suggests that received codeword contains errors.

For error correction, however, the actual value of the non-zero syndrome is used to find the error vector which has corrupted the transmitted codeword. It is evident that a unique syndrome exists for every correctable error pattern. The set of all vectors (corrupted codewords) having the same syndrome is called a coset of the group of codewords [Berlekamp 1968]. Furthermore, the number of codewords in a code, is equal to the number of corrupted codewords in any coset.

In each coset, a word having the minimum weight is chosen as the coset leader [MacWilliams Sloane 1977]. The coset leader is thus the most likely error pattern to have corrupted the codewords (provided the channel is symmetric). In this manner, each syndrome is associated with a particular coset leader. A received corrupted codeword is decoded by computing the syndrome and finding the coset leader corresponding to the syndrome. Finally, the actual decoded codeword is formed by subtracting (addition in GF(2)) the coset leader from the received word. This is equivalent to finding the codeword nearest in distance to the received word.

2.3.3 Convolutional Codes

Convolutional codes [Elias 1955] are a class of error control codes in which information digits are transmitted as a continuous sequence with no block structure. In these coding scheme, the redundant digits check for errors over several consecutive digits. These codes are also referred to as tree codes since the code bits can be represented as a tree whose branches correspond to the coded segments. As with block codes, convolutional codes may be used for correcting random or bursty errors.

Unlike block codes, convolutional codes have check digits which are formed over v successive data digits, where v is the constraint length of the code. A convolutional encoder may be described as a linear-state machine consisting of a v-stage shift register and m linear algebraic function generators [Lin Costello 1983 pp288-290]. The input data, which is usually, though not necessarily, binary, is shifted along the register, b bits at a time. An example with v=3, m=2, b=1 is shown in Fig.2.5.

The encoder of Fig.2.5 can also be represented as a tree diagram of Fig.2.6. In this format, if the first input bit is a zero, the code symbols are those shown on the first upper branch, while if it is a one, the output code symbols are those shown in the first lower branch. This process is continued with every input bit, however the process moves to the next level.

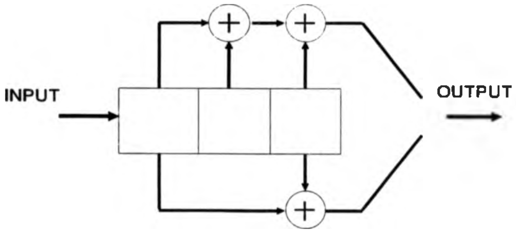


Fig.2.5 Convolutional Encoder for v=3, m=2, b=1.

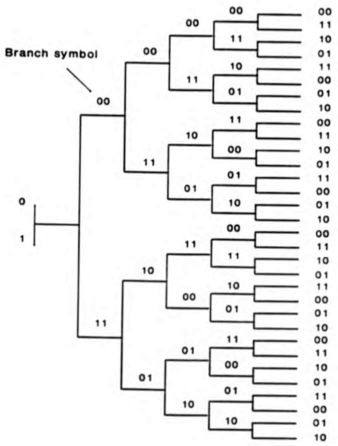


Fig.2.6 Tree-code representation of encoder of Fig.2.5

From Fig.2.6 it also becomes clear that after the first three branches the structure becomes repetitive. The reason for this is obvious from examination of the encoder. The encoder being a finite state machine, has a memory of two bits; it is therefore evident that with the fourth bit entering the shift register, the effect of the first bit is completely removed. Consequently, the data sequences, 10000.... and 00000... generate the same code symbols after the third branch.

A compact method of representing the tree structure of Fig.2.6, is the trellis diagram, shown in Fig.2.7, where the branches due to an input zero are printed in bold. The trellis structure of convolutional codes stems directly from the repetitive nature of the tree diagram. Trellis diagrams facilitates the implementation of maximum likelihood decoding of convolutional codes.

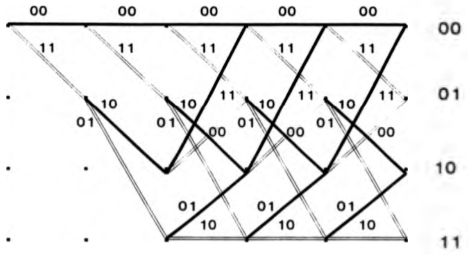


Fig.2.7 Trellis diagram of encoder of Fig.2.5

The error control capabilities of convolutional codes are measured in a similar way to that of block codes. As mentioned in Section 2.3.1, Hamming distance directly controls the error correction capability of a code. The determination of the Hamming distance of convolutional codes is however a difficult task, since the code no longer posses a finite block. The Hamming distance of a convolutional code can be determined by evaluating the Hamming distance between all the possible paths through the trellis and the all zeros path. The paths considered must all start from the state zero. The Hamming distance of the code is then the minimum Hamming distance value obtained via the search process. The error correction capability of the code is then given by equation (2.20). As an example, the Hamming distance of the code represented in Fig.2.7, is five, thus the code can correct double digit errors. In the next Section the decoding process of binary convolutional codes is discussed.

2.3.4 Decoding of Binary Convolutional Codes

As discussed in Section 2.2.1, on a symmetric channel, errors which transform a channel code digit 0 to 1 or 1 to 0 are assumed to occur independently from digit to digit with probability p. If all code sequences are equally likely, the function of the decoder can be described as finding the most probable transmitted sequence based on the received sequence. The criterion used is, broadly, based on two techniques, hard decision and soft decision. In any event, the decoding process involves searching amongst all the possible paths through the trellis and choosing a path which satisfies the decoding criterion.

In this manner, for every m received bits (ie the received code symbol), the decoder assigns a decoding metric to a single section of the trellis diagram. This trellis section which corresponds to b data bits, is appended to the end of the trellis diagram. The assigned decoding metric, is the decoding criterion measure of the received code symbol, with respect to all the branches of the trellis section. L such trellises are stored in the decoder buffer, where L is referred to as the search length of the decoder and is usually set to be five times the constraint length of the code, v [Viterbi 1967]. The decoder then searches through these, L, trellis sections to find a path that best satisfies the decoding criterion.

In hard decision decoding, the assigned metric is based on the Hamming distance. In other words, the received code symbol is compared with the m bits labelling the branches (ie the branch symbols) of the trellis section. The decoding process is then based on choosing a code sequence which differs from the received sequence in the least number of digits [Viterbi 1971]. This is principally based on three main methods: Viterbi algorithm [Viterbi 1967], sequential decoding [Reiffen 1962] and threshold decoding [Massey 1963].

The Viterbi algorithm is based on a complete search of all the possible paths in the trellis. This algorithm results in the minimum error probability, however, it is evident that as the constraint length grows the decoding complexity grows exponentially. The sequential decoding method is based on a limited search algorithm. Here, the distance growth along a given path is monitored; if the growth exceeds a predetermined value a new path is considered. Although this scheme suffers from a higher probability of error and decoder storage overflows, the decoding complexity is somewhat reduced. The threshold decoding technique is based on the majority of all the parity check equations (or a suitable sum or combination of them) related to a particular information digit failing. The check sum may be generated from the syndrome (H matrix).

2.3.5 Soft Decision Decoding

In the binary case of H.D decoding, the decoder is presented with two distinct outputs resulting from a single threshold at the demodulator. This form of decoding however, neglects the additional information which could be made available by the demodulation process. This additional information can be used to improve the decoding process by feed-forward from the demodulator, or feedback from the decoder [Thiede 1972] to adaptively adjust the demodulator threshold levels.

A decoding process which is based on the above criterion is referred to as soft decision (S.D) decoding [Chase 1972]. S.D decoding is a type of probabilistic decoding [Balser Silverman 1954, 1955], the simplest form of which is erasure or null-zone detection [Bloom et al 1957]. The received signal is quantised to three symbols: 1, 0 and null, where null represents a signal element close to the H.D threshold and therefore of doubtful value. In this manner the decoder has a prior knowledge of the digits which are likely to be in error, and can decode accordingly. Null-zone detection can be extended to double-null-zone detection with an improvement in performance [Cahn 1969]. Generally speaking, optimum decoding is obtained when the number of threshold levels tends to infinity. For the binary case however, 16 levels is thought to be adequate [Honary 1981 pp60-61] to obtain near optimum performance.

S.D decoding can be viewed as the assignment of metrics to the received coded digits, based on some likelihood function. That is to say, the assigned metrics are based on the probability of the received digit conditioned on all the possible code symbols having been transmitted. The function of the decoder is therefore to maximise the overall likelihood function for the codeword.

As an example, consider a binary PSK signal system operating over an additive white Gaussian Noise (AWGN) channel, with a transmitted symbol energy of E_s, and a one sided noise spectral density N_0. Let y_j be the j^{th} code digit of the received codeword; and x_{jk} be the j^{th} digit of codeword k, in the codebook. The conditional probability density function of y_j given x_{jk} is [Viterbi 1971]:

$$P(y_j|x_{jk}) = \frac{\exp[-(y_j - \sqrt{E_s}\, x_{jk})^2/N_0]}{\sqrt{(\pi N_0)}} \qquad (2.28)$$

Since each digit in the codeword is affected independently by the AWGN, the overall metric (confidence) for a given codeword is then the product of the equation (2.28) for all the digits in the codeword (ie for all j). In this manner, the k^th^ codeword metric M_k, would be evaluated using equation (2.28) as follows

$$M_k = \prod_{j=1}^{n} P(y_j|x_{jk}) \qquad (2.29)$$

where n is the number of digits in the codeword.

The final decision of the decoder is based on choosing the codeword with the highest metric. In some of the literature a logarithmic likelihood function is used [Viterbi 1971] instead of the actual likelihood function. This is based on taking the logarithm of the digit metric given by equation (2.28). The final log-likelihood for any codeword is then the sum of the log-likelihood functions for each of its digits. The advantage of this arrangement is that additions rather than multiplications, of the digit confidence metrics, need to be evaluated by the decoder. Again, as in the previous arrangement, the decoder would choose the codeword with the highest log-likelihood metric value. Clearly, the implementation of S.D decoding is more complex than for the H.D decoding, However coding gains in the order of 1.5-2dB can be achieved [Wozencraft Jacobs 1968 pp401].

2.4 Modems for Time Variable Channels

The main disturbances on time variable channels can be characterised into the following groups [Perl Shpigel Reichman 1987, Seymour 1987]:

- fading (flat or frequency selective)
- multipath spread;
- Doppler shift;
- various additive noise effects.

A well-designed modem should be only limited in performance by the additive effects. Such a modem therefore, should measure and cancel the effect of the above disturbances. The properties of a given modem are controlled directly by the modulation/demodulation process used. In the following sections the main classes of modulation/demodulation are briefly discussed.

2.4.1 Modulation Methods

In digital modulation schemes, information is conveyed by changing some parameters of the carrier signal. These parameters are: frequency, phase and amplitude. Although a wide variety of combinations of modulation schemes exist, here the following three main binary and multi-level modulation schemes [Clark 1977, Stremler 1982] are considered:

a) amplitude shift keying (ASK),
b) phase shift keying (PSK), and
c) frequency shift keying (FSK).

With ASK modulation the data signal modifies the amplitude of the carrier signal. In the binary case the carrier may be completely suppressed during the data '0', giving on-off keying (OOK). In the multi-level version of ASK, the total amplitude range of the carrier signal is divided into 2^k levels (ie constellation points). Every k data bits is then assigned to a particular level. Clearly this results in the constellation points being closer together and so immunity to noise is reduced.

One particular problem associated with ASK is that of fading. Since the amplitude of the signal is used to convey information, the decision boundaries must be constantly varied. If the decision boundaries cannot be accurately tracked, a poor performance results. This problem is worsened in the case of multi-level ASK, since more decision boundaries exist which must be tracked with greater accuracies. A further practical problem with ASK is that the transmitter is operated with a low peak-to-mean power ratio. Thus, a higher cost transmitter for the same transmitted signal power results.

PSK modulation technique [Clark 1983 pp255] is concerned with changing the phase of the carrier signal in accordance with the data. The simplest form of PSK is the binary PSK [Stremler 1982 pp582]. Here the data bits '0' and '1' are represented by the carrier phases of 0 and $-\pi$. As in the case of ASK, the number of signal constellation points can be extended such that each carrier phase represents more than one bit. This results in the distance between the signal constellation points to be reduced, which in turn causes an increase in system bit error rate (BER). One of the major problems associated with PSK systems is that high channel phase stability is required. Failing this, these phase instabilities must be accurately tracked.

For FSK and its recent derivatives, multiple frequency shift keying (MFSK) [Ralphs 1985], the frequency of the carrier signal is varied according to the data. As an example, for an m-tone MFSK modulation scheme, k data bits are used to select one of the m possible tones, where

$$k = \lfloor \log_2 (m) \rfloor \tag{2.30}$$

where $\lfloor x \rfloor$ is the greatest integer $\leq x$.

In this manner the full bandwidth of the system is divided equally into a sufficient number of frequency slots, where each slot is allocated to one of the transmitted tones. The choice of the actual transmitted tones is an important consideration [Honary Zolghadr Darnell 1989]. The transmitted tone frequencies are chosen such that, over a given symbol interval, the tones form an orthogonal signal set. This point is discussed further in Chapter 6. One advantage of the FSK systems is that increasing the number of tones (ie the signal constellation) does not effect the distance between the constellation points (assuming orthogonal tone spacing). This means that the error rate of the system does not degrade as rapidly as for other systems, however this is achieved at the expense of longer symbol time.

FSK systems operating over a time variable channel are in general more robust than the other methods described above. In most FSK systems the tone spacings are chosen such that the effect of Doppler shifts become insignificant. The effect of multipath spread is normally overcome by choosing the symbol duration to be significantly larger than the expected multipath spread. One major problem associated with FSK modulation systems is the effect of frequency selective fading. This can be overcome by in-band frequency diversity techniques [Proakis 1983 pp470].

2.4.2 Demodulation Methods

Demodulation is the process of converting the channel symbols to the actual received data symbols. In order to obtain the best overall performance, the process of conversion must take the communication channel into account. There are two general methods of demodulation schemes: coherent, which requires the acquisition of a local reference signal in phase coherence with the received signal; and non-coherent, which is phase insensitive.

Coherent detection methods are generally based on deriving the carrier phase and frequency from the received signal itself. For the case of ASK, the received signal (after appropriate filtering) is fed to a phase locked loop (PLL) [Stremler 1982 pp220].

Derivation of the carrier signal is somewhat more difficult for PSK modulation, since no distinct carrier component is present in the modulated waveform. A simple method of extracting the reference signal is to full-wave rectify the received signal and isolate the fundamental frequency component at twice the carrier frequency. This is fed to a frequency divider to give a carrier frequency component which has a phase of 0° or 180° with respect to the carrier in any received signal element. Alternative methods of carrier extraction are available [Lindsey Didday 1968], but they all result in the same uncertainty in the phase of the carrier. The effect of this phase uncertainty can be removed by applying differential PSK [Clark 1983 pp78]

The coherent detection of FSK signal involves considerable equipment complexity. This is because for the satisfactory operation of the FSK system, it is important that there are no phase discontinuities at the boundaries between adjacent signal elements. This results in the phase of the carrier over any signal element being a function of the previously transmitted element. Therefore, the reference carrier signals needed for the detection of the tones cannot be extracted from the received signal [Clark 1983 pp79]. Consequently, coherent detection of FSK signals is not normally used.

In noncoherent detection techniques, the receiver has no prior knowledge of the phase of the carrier signal. It follows that the operation of a noncoherent detector is not affected by the phase instabilities of the carrier signal.

Noncoherent detection of ASK signal is normally achieved by either the envelope detection methods [Clark 1983 pp84] or noncoherent correlation methods [Clark 1983 pp67-69].

In the envelope detection method, the received signal is squared (or half-wave rectified) and passed through a low-pass filter. The output of the filter would then correspond to one of the possible constellation voltage levels. In the case of noncoherent correlation method, the inphase and quadrature components of the received signal are evaluated, then squared and added. The signal formed in this manner is again a direct representative of the transmitted constellation voltage level.

Noncoherent detection of PSK modulation is normally referred to as differentially-coherent detection. Here differential coding must be used at the modulator. The detector therefore operates over the duration of two adjacent signal-elements [Stremler 1982 pp587-588].

The detection process in a noncoherent FSK system is based on a bank of noncoherent correlators. Each correlator evaluates the inphase and quadrature component of the received signal at one of the possible constellation frequencies for exactly one symbol interval. The correlator with the highest value is then chosen as the demodulators output. An alternative method of noncoherent detection of FSK signals is based on replacing the correlators with quenched resonators [Clark 1983 pp64-65]. In any event, the detection method utilises a bank of detectors each matched to one of the constellation frequencies.

The error performance of the above demodulation schemes (coherent and noncoherent) under AWGN is outlined extensively elsewhere [Arthurs Dym 1962, Edwards 1973]. Generally, the probability of error for a given signal to noise ratio is lower in the coherent systems [Edwards 1973] since the out of phase noise component of the signal is rejected. This assumes the availability of an exact replica of the carrier signal, exact even to the timing of RF phase. The performance of coherent detection methods is however severely degraded under conditions encountered over time variable channels.

This page has been intentionally left blank.

CHAPTER 3

REAL TIME CHANNEL EVALUATION

3.1 Introduction

The distortion imposed by the channel on the transmitted data stream in a digital communication system is normally observed in the form of errors at the receiver. The main objectives of any communication system are to minimise the number of these errors and to maximise the throughput of the system. In order to optimise the system performance adaptively in response to channel conditions, an estimate of the receiver's error rate is required to initiate control actions. Real time channel evaluation (RTCE) techniques [Betts Ellington Jones 1970, Ellington 1970, Darnell 1978, Darnell 1983a] are useful tools for obtaining on-line estimates of the channel state.

The most attractive form of RTCE is the one which utilises the normal operating signals of the communication system. Examples of this method of RTCE, include [Darnell 1983b]:

a) In-Band RTCE;

b) RTCE using soft decision information;

c) RTCE via repeat-request control signal, monitoring;

d) RTCE by traffic signal modification;

e) RTCE by assessment of synchronisation quality;

f) RTCE by pseudo-error counting.

In essence, all the above RTCE techniques attempt to predict the system performance by direct or indirect measurement of effective signal-to-noise ratio (SNR). In this manner, information relating to overall system error rate or predicted throughput (in the case of ARQ systems) can be supplied by the RTCE system. Although these informations are vital to successful system management, additional data regarding the distribution of errors would enable the system controller to choose the best form of error control technique.

In this Chapter the principles of a new real-time channel evaluation (RTCE) technique applicable to digital communication channels are described. This technique, which is known as statistical RTCE (SRTCE) [Zolghadr Honary Darnell 1989] is particularly applicable to channels were the effect of noise is severe. SRTCE is a unique form of RTCE, which can provide both channel error rate and error distribution characteristics. The technique is based on Markov processes [Kemeny Snell 1976 pp24-42] and relies on manipulating the data source's statistical properties so that they assume predetermined values, whilst at the same time preserving the stationary nature of the uncoded data. This is achieved through the use of a source coding process which serves two purposes:

a) it removes the inherent redundancy associated with the uncoded data stream (in this case, the English text), thus providing data compression;

b) it introduces selective redundancy into the data stream, as required by the RTCE system.

The technique then relies on monitoring the source encoded digital data stream at the receiver and determining its statistics. Any deviation of these statistics from those specified for the "pre-engineered" transmitted data then potentially provides an indication of channel state.

Channel errors also cause other problems for the communicator. One of the major difficulties encountered is the problem of transmitter-receiver synchronisation. Conventionally the source data is divided into segments, e.g. bytes or blocks, and these segments are then encoded in order to provide error control capabilities matched to the expected channel characteristics. However, most codes can give rise to a degree of ambiguity at the receiver under conditions of noise and distortion; clearly, this must be resolved by the decoder. It is therefore essential for the receiver to be able to detect the start and finish of each data segment so that error control decoding operations can be performed satisfactorily. There are a number of possible methods for dealing with this problem, including:

(i) initial synchronisation, followed by precise subsequent timing at the receiver (single-shot synchronisation) [Ralphs 1985 pp120-131];

(ii) initial synchronisation followed by segment synchronisation, through insertion of segmenting information (periodic synchronisation, e.g. RS232 serial links) [Ralphs 1985 pp120-131];

(iii) synchronisation using variable-length, uniquely-decodable codes (e.g. Huffman codes or T-codes) [Ferguson Rabinowitz 1984, Titchener 1987].

Although schemes (i) and (ii) above are normally sufficient for most practical purposes, they are unlikely to be adequate when the channel error rate is high. In such situations, the channel imposes severe ambiguities which cannot be resolved by timing relative to an initial reference or by the insertion of segmenting information. Scheme (iii), however, distributes the synchronisation information throughout the code words of the variable-length code, thus allowing the decoder to re-synchronise quickly following any disturbances. T-codes, in particular, have a rapid re-synchronisation performance.

In Section 3.2.2, the use of T-codes in a SRTCE system and the method of code set selection are examined. In Section 3.3 the results of the simulation studies of the SRTCE system are presented for the cases of an AWGN channel and a channel with memory.

3.2 Statistical Real Time Channel Evaluation (SRTCE)

The SRTCE technique [Darnell Honary Zolghadr 1989] is based upon the statistical analysis of both the transmitted data and the received data prior to decoding. This analysis enables the formulation of two basic models for the encoded data at the transmitter and the receiver. The technique is then used to derive a channel model based on finite-state Markov processes.

3.2.1 The SRTCE System

The SRTCE system [Zolghadr Honary Darnell 1989] exploits the redundancy in the transmitted data, in order to evaluate the channel state. The evaluation requires the removal of the "natural" source redundancy and the introduction of deterministic redundancy in such a way that the natural probabilities of the source states are modified in the transmitted data stream. This requirement is achieved by utilising a source coding stage. Analysis of this latter stream yields a statistical model, M1, for the modified source. The received bit stream is then monitored and its statistical properties are determined in order to formulate a model, M2. The SRTCE technique is then used to derive a channel model, M3, based on finite state Markov processes.

Clearly, if the channel is perfect (i.e. with no noise or distortion), the received data stream would have the same statistical structure as the transmitted data, i.e. M1 and M2 would be identical. However, under normal operating conditions of channels with noise and distortion, any variation in the received data statistical structure can potentially provide information on channel state.

In the following discussion, a binary source will be assumed. The binary bit stream from the output of the source-coding stage, which includes the deterministic redundancy, is modelled using a 2-state Markov chain [Kemeny Snell 1976], as shown in Fig.3.1. The model is a finite stochastic process whose output is purely dependent on the immediate past [Kanal Sastry 1978]. At each state, prior to a transition, the process outputs a single binary digit.

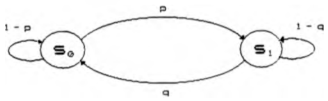


Fig.3.1 2-state model, M1, of a binary source

The states are mapped to the output binary sequence using the following rule:

$$\begin{array}{lcl} Z_n = S_0 & : & Q_n = 0 \\ Z_n = S_1 & : & Q_n = 1 \end{array} \qquad (3.1)$$

where Q_n is the output and Z_n is the state at step n. The basic structure of Fig.3.1 has the following model parameters:

$$\begin{array}{l} p(Z_n=S_0 \mid Z_{n-1}=S_1) = q \\ p(Z_n=S_1 \mid Z_{n-1}=S_0) = p \end{array} \qquad (3.2)$$

where P(x | y) indicates "the conditional probability of x given y". The transition matrix, G, for the source is

$$G = \begin{bmatrix} 1-p & p \\ q & 1-q \end{bmatrix} \qquad (3.3)$$

The above transition matrix is regular [Kemeny 1976 pp69], i.e. the state probabilities assume constant values as the transmission interval tends to infinity, regardless of the initial starting conditions (note, in some literature, the term ergodic is used instead of regular [Massey 1971 pp94-100]). In the steady-state, the state probabilities are given by

$$\Gamma(t+1) = \Gamma(t) = \Gamma(t).G \qquad (3.4)$$

where $$\Gamma(t) = [p(S_0) \quad p(S_1)] \qquad (3.5)$$

is the state probability vector as time $t \to \infty$, $p(S_1)$ is the probability of the process being in state S_1 and $p(S_0)$ is the probability of the process being in state S_0.

Solving equation (3.4) yields

$$p(S_0)/p(S_1) = q/p \qquad (3.6)$$

However $$p(S_0) + p(S_1) = 1 \qquad (3.7)$$

and therefore substituting from equation (3.7) in equation (3.6) for $p(S_0)$ and $p(S_1)$ gives

$$p(Z_n=S_1) = p/(p+q) \qquad (3.8)$$

$$p(Z_n=S_0) = q/(p+q) \qquad (3.9)$$

The encoded data, prior to transmission, is assumed to be statistically stationary. Therefore, the M1 model can be determined off-line by examining an arbitrary sample of the encoded data, of specified length, in order to evaluate the state transition probabilities. The M1 model can therefore be completely described by the parameters p and q, where p and q are given by equation (3.2). It must be noted that any drastic long term deviations of the M1 model parameters would cause an incorrect prediction of the channel model parameters; therefore, if the M1 model parameters are variable (i.e. the source is not stationary), for accurate SRTCE these changes must be signalled to the receiver.

It is clear that, if p+q =1, the model becomes memoryless [Kanal, Sastry 1978]. In the following discussion, a more general case of models with one-bit dependency is considered, since these models collapse to a memoryless case if appropriate additional constraints are imposed upon them.

The M2 model which is of similar structure to the M1 model, is dependent both upon the form of the transmitted data and the nature of the channel. Since the channel state is continually varying, this model must be evaluated on-line.

The channel model, M3, is then derived from models M1 and M2. The probabilities of receiving possible pairs of binary digits at the receiver are:

$$\left.\begin{aligned}
p_r(11)&=p_e(00).p_d(11)+p_e(01).p_d(10)+p_e(10).p_d(01)+p_e(11).p_d(00)\\
p_r(10)&=p_e(00).p_d(10)+p_e(01).p_d(11)+p_e(10).p_d(00)+p_e(11).p_d(01)\\
p_r(01)&=p_e(00).p_d(01)+p_e(01).p_d(00)+p_e(10).p_d(11)+p_e(11).p_d(10)\\
p_r(00)&=p_e(00).p_d(00)+p_e(01).p_d(01)+p_e(10).p_d(10)+p_e(11).p_d(11)
\end{aligned}\right| \qquad (3.10)$$

where $p_r(XY)$, $p_e(XY)$ and $p_d(XY)$ are the probabilities of obtaining the digit X followed by the digit Y at the receiver input, channel error pattern generator and the transmitter input, respectively. It is evident that

$$p(01)=p(10) \qquad (3.11)$$

and therefore the above set of simultaneous equations can be represented in matrix form as

$$\underline{R} = \underline{D} \cdot \underline{C} \tag{3.12}$$

where the above matrices in transpose form are:

$$\left.\begin{array}{l} \underline{R}^T = [p_r(11) \quad p_r(10) \quad p_r(00)] \\ \underline{C}^T = [p_e(00) \quad p_e(10) \quad p_e(11)] \end{array}\right| \tag{3.13}$$

and

$$\underline{D} = \begin{vmatrix} p_s(11) & 2 \times p_s(10) & p_s(00) \\ p_s(10) & p_s(11)+p_s(00) & p_s(10) \\ p_s(00) & 2 \times p_s(10) & p_s(11) \end{vmatrix} \tag{3.14}$$

From (3.12), $\underline{C}$ can be evaluated by:

$$\underline{C} = \underline{D}^{-1} \cdot \underline{R} \tag{3.15}$$

where the channel bit error rate (BER) (i.e. $p_e(1)$) is given by

$$p_e(1) = p_e(11) + p_e(10) \tag{3.16}$$

In order to obtain the matrix $\underline{C}$, the matrix $\underline{D}$ must be non-singular [Stroud 1982 pp155]. It can be seen that if the probabilities, $p_s(1)$ and $p_s(0)$ are equal, i.e. the binary state probabilities of the source are even, then the rank of the matrix $\underline{D}$ will be less than three and it will be singular. For this reason, the main requirement of the SRTCE system is that the source symbol probabilities should be unequal, i.e. $p_s(1) \neq p_s(0)$. Preliminary investigations [Darnell, Honary, Zolghadr 1989] have suggested that the probabilities of the binary source states less than 0.3 is acceptable.

Once $p_e(00)$, $p_e(10)$ and $p_e(11)$ have been evaluated, the channel model parameters can be obtained. Thus $p_c(11)$, $p_c(10)$, $p_c(01)$ and $p_c(00)$ are given by:

$$\left.\begin{aligned} p_a(11) &= p_a(1).p_a(1/1) \\ p_a(10) &= p_a(1).p_a(0/1) \\ p_a(01) &= p_a(0).p_a(1/0) \\ p_a(00) &= p_a(0).p_a(0/0) \end{aligned}\right| \qquad (3.17)$$

substituting from equation (3.2) for the conditional probabilities $p_a(1|1)$ and $p_a(0|1)$ yields

$$\left.\begin{aligned} p_a(11) &= p_a(1).(1-q_a) \\ p_a(10) &= p_a(1).q_a \\ p_a(01) &= p_a(0).p_a \\ p_a(00) &= p_a(0).(1-p_a) \end{aligned}\right| \qquad (3.18)$$

where p_a and q_a are the channel model parameters given by (3.2). Solving equations (3.18) for q_a and p_a yields

$$q_a = \frac{p_a(10)}{p_a(11) + p_a(10)} \qquad (3.19)$$

$$p_a = \frac{p_a(10)}{p_a(00) + p_a(10)} \qquad (3.20)$$

As with any statistical technique, only "long-term" effects can be satisfactorily characterised. It is therefore essential that a past record of the received bit stream should be stored in a buffer; thus, ideally an infinite buffer is required. However, since the channel state is constantly varying, a limited buffer size is appropriate. For this reason, the SRTCE system employs a finite-length buffer of length N bits which is updated continuously with blocks of K successive bits of the received data. The SRTCE analysis is then carried out on the stored N bits after every new block of K bits is entered, the "oldest" K-bit block in the buffer then being discarded. The choice of the buffer size, N, depends on the required accuracy and the time variability of the channel. This design consideration will be discussed further in Section 3.3.

3.2.2 Use of T-codes in an SRTCE System

A useful feature of T-codes is that, for a given augmentation degree, binary code sets of varying 1:0 ratio can be constructed (for construction methods of T-codes, the reader is referred to Chapter 2 of this thesis). It has been shown [Titchener 1987] that the ASCII character set can be represented more efficiently using a special class of T-codes; it is also shown that, by using rationalised character assignment (i.e. most probable source symbols assigned to the shortest codewords), the coding costs for the ASCII character set approach 4.5 bits/character, compared with as much as 12 bits/character using an RS232 serial link format.

It has been shown that [Darnell, Honary, Zolghadr 1989] for reliable SRTCE, the probability of transmitting a binary 1 or 0, whichever is more appropriate, must be less than 0.3. It is therefore desirable that the data encoded using the T-code should have this structure. In order to take full advantage of the variable-length nature of T-codes, rationalised-character-to-codeword assignment should be used. For this purpose, the frequencies of occurrence were evaluated for the main elements of the ASCII code set, including all the lower and upper case letters, space and carriage-return for a sample of the English text; the source analysed was 2 chapters of Jane Austin's novel "The Watsons", comprising approximately 90k characters. The shortest-length codewords were then assigned to the most frequent characters, as shown in Table 3.1. A computer search procedure was used to find an applicable code set; this involved the development of a general-purpose computer program for the construction of the complete family of T-codes of augmentation degree 7 with a single bit literal code, i.e. a total of 129 code words. In general, the total number of T-code sets of this family is given by [Titchener 1986]

$$\prod_{i=0}^{Q-1} (2^i(n-1)+1) \qquad (3.21)$$

where Q is the degree of augmentation and n is the number of bits in the original base set. Therefore, there are approximately 1.3E9 possible code sets in the family corresponding to Q=7 and n=2. It is consequently unrealistic to attempt to find a suitable T-code set by a complete search method. It has been shown [Titchener 1985] that the choice of group offsets of maximum allowable weight leads to an uneven T-code set, where the degree of probability imbalance is dependent upon the weight of the group offsets. A limited search was therefore initiated, starting with the T-code set S(1,3,7,15,10,6,0). In total, 40 code sets were examined and an encoder constructed and tested for each with the sample of the English text mentioned previously. Simulation techniques were used to determine the $p_d(1)$ of the encoded data and the coding costs (ie the number of bits per character). It was found that the probability imbalance was inversely related to the efficiency of the code. A code set (1,3,7,15,10,6,8) was found with $p_d(1)=0.737$ and coding costs of 6.0 bits/character compared with 4.5 bits/character for Titchener's ASCII T-codes [Titchener 1987]; this satisfies the requirements of the RTCE technique. Fig.3.2 shows the length distribution for this code. In Table 3.1, the ASCII characters and their corresponding T-code representations are given.

ASCII	Probability	T-code
sp	0.168284	0
e	0.102470	10
t	0.068360	110
o	0.065045	11100
a	0.061839	111010
n	0.055264	111100
i	0.051722	1111110
s	0.050552	1111100
r	0.049762	1110110
h	0.047974	1111010
d	0.036590	11111110
l	0.031098	11111010
u	0.024881	11101110
m	0.022444	11110110
c	0.018793	111111110
cr	0.018588	111110110
w	0.017829	111011110
y	0.017320	1111111110
f	0.016334	1111101110
g	0.014125	1111011110
p	0.013659	1111011100
b	0.010832	11111111110
v	0.008752	11111011110
k	0.004669	11101111110
I	0.004062	11101111100
E	0.001841	11110111010
H	0.001820	111111111110
M	0.001636	111110111110
W	0.001408	111011111110
x	0.001365	111011111010
L	0.001300	111101111110
q	0.001267	111101111100
S	0.001159	111101110110
T	0.001029	1111111111110
O	0.000921	1111101111110
C	0.000834	1110111111110
G	0.000715	1110111110110
j	0.000704	1111011111110
A	0.000607	1111011111010
B	0.000368	1111011101110
N	0.000347	11111111111110
Y	0.000325	11111011111110
z	0.000271	11101111111110
F	0.000260	11101111101110
P	0.000184	11110111111110
D	0.000141	11110111110110
V	0.000097	11110111011110
R	0.000076	111111111111110
J	0.000043	111110111111110
U	0.000032	111011111111110
K	0.000000	111011111011110
X	0.000000	111101111111110
Q	0.000000	111101111101110
Z	0.000000	1111111111111110

Table 3.1. Ordered ASCII characters and the corresponding T-codes.

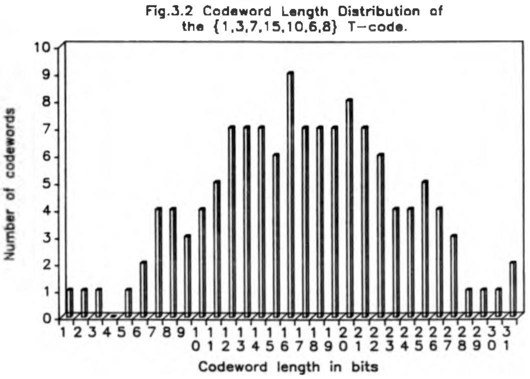
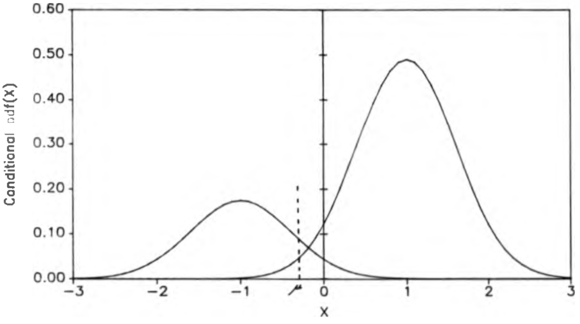


Fig.3.2 T-code length distribution

3.2.3 SRTCE in the Presence of AWGN

A further requirement of the binary form of SRTCE system considered here, is that the channel should be a binary symmetric channel (BSC) - a requirement implied by the form of the expression (3.10); consider Fig.3.3 which shows the conditional probability density function (pdf) of the received (baseband) signal in the presence of AWGN with standard deviation of 0.6. The decision threshold is set at an arbitrary value μ.

Fig.3.3 The Conditional pdf of Baseband Signal (+1, −1) in Presence of AWGN with Standard Deviation of 0.6



The probability of error for this arrangement is given by

$$p(\text{error}) = p_a(0)\int_{(\mu-a0)/\sigma}^{\infty}\frac{1}{\sqrt{(2\pi)}}\,e^{-y^2/2}\,dy + p_a(1)\int_{-\infty}^{(\mu-a1)/\sigma}\frac{1}{\sqrt{(2\pi)}}\,e^{-x^2/2}\,dx \quad (3.22)$$

$$= p_a(0)\,\mathrm{Erfc}[(\mu-a0)/\sigma] + p_a(1)\,(1-\mathrm{Erfc}[(\mu-a1)/\sigma]) \quad (3.23)$$

where σ is the standard deviation of the noise (equal to 0.6), Erfc()

is the complementary error function [Schwartz 1982 pp318-327] and a0 and a1 are the transmitted voltages representing binary digits 0 and 1 respectively.

The optimum decision threshold level is obtained when the following condition is satisfied [Schwartz 1982 pp318-327]

$$p_a(0)\ f(\mu - a0) = p_a(1)\ f(\mu - a1) \tag{3.24}$$

where f(x) is defined as

$$f(x) = \frac{1}{\sqrt{(2\pi)}\sigma}\, e^{-x^2/2\sigma^2} \tag{3.25}$$

Solving equation (3.23) for μ, yields a non-BSC i.e.

$$p(\text{error}|0) \neq p(\text{error}|1) \tag{3.26}$$

In order to obtain a BSC, equation (3.23) must be solved with the constraint that $p_a(1)$ and $p_a(0)$ are identical. This yields a decision threshold level equal to the average of the two transmitted voltage levels.

The AWGN channel is simply modelled using the 2-state Markov process described in Section 3.2.1. In this case, the resulting channel is memoryless: however, the derived equations still apply. In Section 3.3.1, the results of simulation studies for this type of channel are discussed.

3.2.4 SRTCE for the Gilbert-Elliot Channel

The Gilbert-Elliot model [Elliot 1963, Kanal Sastry 1978] is a statistical channel model based on Markov processes. The state diagram of this model is shown in Fig.3.4.

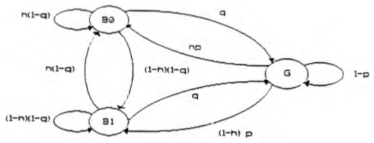


Fig.3.4 The Gilbert-Elliot channel model.

The model comprises three states, one good state (G) and two bad states (B0, B1). No errors are produced in state B0; in the state G, random errors are produced with probability f; in state B1, errors occur with probability one. It is possible to visualize this channel as making transitions between long periods in the good state with small background probability of error, f, and periods in a burst state with relatively larger probability of error. The BER for this channel is therefore [Kanal, Sastry 1978],

$$p_e(1) = \frac{f.q + h.p}{p+q} \qquad (3.27)$$

where q = p(B-->G), p = p(G-->B), h = p(error|B) and f = p(error|G), with p(X-->Y) denoting the probability of transition from state X to state Y.

It is clear that the above channel is binary-symmetric, since the error patterns generated by this model are added (modulo-2) to the encoded data, and are therefore not conditioned by the data symbols. It can be shown that the error patterns produced have one-bit dependency [Elliot 1963]; therefore, equation (3.10) is valid for this model. The performance of the SRTCE system operating over a channel characterised by this model is discussed in Section 3.3.2.

3.3 Performance Study of SRTCE System

In this section, the results of simulation studies are presented. The analysis concentrates on two channels, the AWGN channel and a channel with memory. As mentioned previously, the encoded data statistical parameters must be first determined off line. As part of this process, the effect of buffer size on the encoded data parameters was investigated. The sample of the English text was encoded using the T-code set S{1,3,7,15,10,6,8} and analysed by the scheme described in Section 3.2 for various fixed buffer sizes. Table 3.2 shows the encoded data averaged statistical parameters for a range of buffer sizes and their respective standard deviations. It is clear that effect of varying buffer sizes on the mean value of these parameters is negligible (maximum of 1%); however, the standard deviation of these parameters is inversely related to the buffer size. Since it is desirable to minimise the standard deviation of the statistical parameters, large buffer sizes must be employed in the SRTCE system.

3.3.1 AWGN Channel

Using simulation techniques, the effect of buffer size and the actual channel BER on the performance of the SRTCE technique were investigated under AWGN conditions. For this analysis, buffer sizes ranging from 1000 to 15000 bits were examined. For each buffer size, the corresponding data model parameters previously determined (see Table 3.2) were used; while keeping the channel signal to noise ratio constant, the BER for every block was evaluated and then averaged over the total number of blocks.

Buffer size	$p_e(00)$	$p_e(01)$	$p_e(10)$	$p_e(11)$
15000	0.057904	0.206214	0.206214	0.529669
14000	0.057935	0.206246	0.206246	0.529571
13000	0.057971	0.206291	0.206291	0.529448
12000	0.058003	0.206337	0.206337	0.529323
11000	0.058035	0.206382	0.206382	0.529200
10000	0.058072	0.206420	0.206420	0.529088
9000	0.058108	0.206447	0.206447	0.528998
8000	0.058123	0.206471	0.206471	0.528936
7000	0.058136	0.206482	0.206482	0.528900
6000	0.058144	0.206493	0.206493	0.528871
5000	0.058145	0.206493	0.206493	0.528870
4000	0.058130	0.206501	0.206501	0.528868
3000	0.058109	0.206538	0.206538	0.528817
2000	0.058112	0.206579	0.206579	0.528731
1000	0.058137	0.206575	0.206575	0.528715

Buffer size	σ of $p_e(00)$	σ of $p_e(01)$	σ of $p_e(10)$	σ of $p_e(11)$
15000	0.001558	0.002300	0.002300	0.005567
14000	0.001591	0.002412	0.002412	0.005805
13000	0.001641	0.002559	0.002559	0.006103
12000	0.001699	0.002728	0.002728	0.006430
11000	0.001771	0.002906	0.002906	0.006797
10000	0.001875	0.003090	0.003090	0.007179
9000	0.001981	0.003296	0.003296	0.007616
8000	0.002090	0.003506	0.003506	0.008051
7000	0.002264	0.003725	0.003725	0.008509
6000	0.002497	0.003979	0.003979	0.009056
5000	0.002680	0.004269	0.004269	0.009686
4000	0.002953	0.004673	0.004673	0.010585
3000	0.003503	0.005331	0.005331	0.012103
2000	0.004396	0.006360	0.006360	0.014402
1000	0.006368	0.008332	0.008332	0.018586

Table 3.2 Averaged encoded data model parameters and their standard deviations for varying buffer size.

Fig.3.5 shows the effect of buffer size N and the actual channel BER on the rms error in the BER estimated (ie the standard deviation of the estimated channel BER with respect to the actual channel BER evaluated for every block) via the SRTCE procedure.

From Fig.3.5, it is apparent that the received data must be monitored over at least 3000 bits in order to provide accuracies in the range of 90%. This implies that a buffer must be employed at the receiver with a size greater than 3000 bits. At a transmission rate of say 9600 bits/s, the delay caused by this buffer is of the order of 1/3 second.

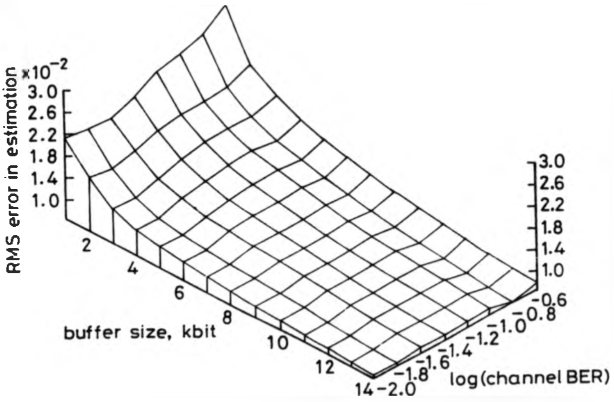


Fig.3.5 Effect of buffer size N and the actual channel BER on the rms error in the BER estimated, for AWGN channel

It is evident that the size of the buffer required will be inversely related to the channel BER, i.e. at a low channel BER the received bit stream must be monitored for a longer period to achieve a specified level of SRTCE precision.

3.3.2 The Gilbert-Elliot Channel

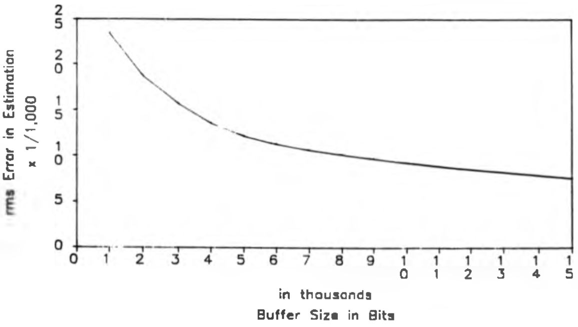
In order to investigate the performance of the SRTCE system operating over channels with memory, the Gilbert-Elliot model was used as a channel error source. Due to the large number of variables involved, the initial investigation was limited to the effect of buffer size on the performance of the SRTCE system for one channel BER. For this analysis the following channel model parameters were used:

$$\left.\begin{array}{ll} p = 0.04 & q = 0.76 \\ h = 0.2 & f = 0.01 \end{array}\right| \qquad (3.28)$$

For this choice of model parameters, equation (3.27) gives the channel BER as 0.0495. This agrees closely with the channel BER obtained through simulation (ie 0.0508). Fig.3.6 shows the effect of buffer size on the rms errors in estimation. It can be seen that, as for the case of the AWGN channel, the rms errors increase with the reduction in buffer size.

It is apparent from Fig.3.6 that reasonably accurate estimates of the channel state can be obtained using the SRTCE system. Although the long term error in the SRTCE technique is zero, a buffer of at least 4000 bits is required to maintain the rms error in estimation below 0.015.

Fig.3.6 rms Error in Estimation vs Buffer Size, Gilbert Elliot Channel, Channel BER = 0.0508



3.4 Discussions

The SRTCE system based on Markov processes enables channel characteristics to be predicted with reasonable accuracy. In this scheme, statistical analysis techniques are used to determine an equivalent channel model in real-time. The main requirement of the technique is that of the selective introduction of redundancy into the transmitted data. This redundancy is incorporated via a source coding process employing variable-length codes. The specific source coding method discussed in this study employs T-codes, which not only provide data compression, but also have good self-synchronisation characteristics.

The SRTCE system has a number of practical applications: for example, it can be applied to line communications involving say information transfer between a mainframe computer and its distributed terminals. In such a system, line coding schemes such as Manchester encoding [Stremler 1982 pp535] could be used in order to eliminate the DC component of the source-coded signal; the SRTCE delay caused by the receiver buffer for such a system would typically be a fraction of a second. In that it is capable simultaneously of providing source coding, synchronisation, error control and channel state information, the SRTCE technique introduced in this study represents one form of multi-functional coding [Darnell, Honary 1986].

This page has been intentionally left blank.

CHAPTER 4

EMBEDDED BLOCK ENCODING

4.1 Introduction

In this chapter, the principles of embedded block codes, which employ a combination of forward error correction and detection (FEC/FED) for error control in an ARQ environment are introduced. In this scheme, a concatenated code [Forney 1966] with an inner code for error correction and an outer code for error detection is used. A retransmission of the erroneous information packets is requested if the outer code decoder detects the presence of any errors after the initial correction has been performed by the inner decoder.

In Section 4.2, the performance of the system is analysed in terms of reliability and throughput efficiency [Lin Costello 1983]. The code structure and the generalised theoretical expressions for the reliability and throughput efficiency are derived. These are followed by a performance comparison between embedded array and conventional array codes.

The retransmission strategy determines the system throughput; three basic methods exist [Lin Costello 1984] i.e.

(i) stop-and-wait;
(ii) go-back-N;
(iii) selective-repeat.

For the sake of simplicity, the throughput efficiency of the proposed error control scheme incorporated with a selective-repeat ARQ is analysed assuming an infinite receiver buffer [Lin Costello

1983]. The results given in Section 4.2.3, show that the embedded scheme yields high reliability over a wide range of input bit error rates (BER). Finally, the modified embedded array codes which employ new sub-blocks in each stream, are described in Section 4.3.

4.2 Embedded Array Codes

The embedded encoding approach [Zolghadr 1987, Darnell Honary Zolghadr 1988] is one in which block-formatted data is transmitted at several different rates simultaneously in an automatic repeat request (ARQ) type of communication system. After each block, the receiver indicates to the transmitter the highest rate at which it has been able to decode the data in the previous block interval, as shown in Fig.4.1.

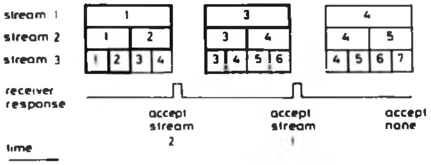


Fig.4.1 ARQ Protocol of the Embedded Encoding Scheme.

This technique attempts to match the instantaneous reception rate more accurately to the channel capacity available at any time; it thus represents a generalisation of the ARQ principle, which itself has long been recognised as an effective error control technique for diffuse channels.

The code is comprised of two main codes, D1 and D2. D2 the outer code has a detectability of up to t errors, while D1 is a combination of three inner codes, C1, C2 and C3, capable of correcting F_1, F_2 and F_3 or fewer errors respectively, such that $F_1 > F_2 > F_3$ and

$$F_i = \left\lfloor (d_i - 1)/2 \right\rfloor \qquad (4.1)$$

where i is an integer ($1 \le i \le 3$), d_i is the Hamming distance of code Ci and $\lfloor x \rfloor$ denotes the integer part of x.

The following is the only constraint on the size of the inner codes C1, C2 and C3:

$$J_1.N_1 = J_2.N_2 = J_3.N_3 \qquad (4.5)$$

where J_i is a positive integer with $J_1 < J_2 < J_3$ and, N_i is the code length of Ci. In the analysis that follows $J_1=3$, $J_2=6$ and $J_3=12$.

The encoding is carried out in three stages:

(i) a message of K information digits is divided into four information sub-blocks or information packets, {k1,k2,k3,k4};

(ii) each sub-block is then encoded in a manner depending on its position in the main block, using the inner codes C1, C2 and C3 as shown in Fig.4.2;

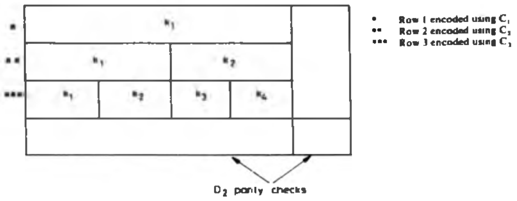

*
**
***
k_1
k_1
k_2
k_1
k_2
k_3
k_4
* Row 1 encoded using C_1
** Row 2 encoded using C_2
*** Row 3 encoded using C_3
D_2 parity checks

Fig.4.2 Embedded Encoding Block Structure.

(iii) the outer code D2 is then applied to the inner codes information block K to produce an encoded block for transmission.

The decoding involves error correction within each of the sub-block codes, followed by error detection by the outer decoder. When a block is received, it is decoded by the inner code decoders; if F_i, or fewer, errors are present within a sub-block, these can be successfully corrected, and the corrected information sub-block is stored in the outer decoder's buffer. Clearly, when the received sub-block contains F_i or fewer errors, the initial correction of the sub-block leads to error-free digits in the outer decoder's buffer; however, if the sub-block contains (F_i+1) or more digits in error, the inner decoder will fail to decode successfully, and the output sub-block will be in error.

The outer decoder performs error detection on the sub-blocks in the buffer, starting at the 1st sub-block and proceeding to the 4th; if the number of digits in error at this stage is $\leq t_i$ (where t_i is the error detection capability of the code i), the outer decoder will detect the presence of these errors and will accept the error-free sub-blocks, whilst requesting the retransmission of those in error. When the number of digits in error in the outer decoder input buffer is $>t_i$, the outer decoder may be unable to detect their presence and the incorrect sub-blocks may be output.

The error control technique described above can be viewed as an adaptive hybrid ARQ scheme [Farrell Honary Bate 1988], in which the information packets (sub-blocks) are encoded, in the same block, using codes of decreasing error correcting capability. Thus, at times when the bit error probability (BER) is relatively high, only the sub-block encoded with the most powerful code may be accepted at the expense of low overall code rate and a subsequent reduction in system throughput efficiency. However, when the channel signal-to-noise ratio (SNR) is comparatively high, the sub-blocks encoded using the less powerful code are also accepted, resulting in an increase in the system throughput efficiency.

4.2.1 Code Structure

Consider the case when four sub-blocks of information are to be transmitted, each comprising three information digits, a_i, b_i and c_i, i.e.

$$\begin{aligned} k1 &= a_1 \quad b_1 \quad c_1 \\ k2 &= a_2 \quad b_2 \quad c_2 \\ k3 &= a_3 \quad b_3 \quad c_3 \\ k4 &= a_4 \quad b_4 \quad c_4 \end{aligned} \qquad (4.3)$$

The sub-blocks are encoded using the inner codes C1, C2 and C3 as shown in Fig.4.2; the resulting block is then encoded using D2 (the outer code), to form the main block for transmission as illustrated in Fig.4.3

$$\begin{array}{l|l} a_{11} \; a_{12} \; a_{13} \cdots a_{1N_1} \; b_{11} \; b_{12} \; b_{13} \cdots b_{1N_1} \; c_{11} \; c_{12} \; c_{13} \cdots c_{1N_1} & P_1 \\ a_{11} \cdots a_{1N_2} \; b_{11} \cdots b_{1N_2} \; c_{11} \cdots c_{1N_2} \; a_{21} \cdots a_{2N_2} \; b_{21} \cdots b_{2N_2} \; c_{21} \cdots c_{2N_2} & P_2 \\ a_{11} \cdots a_{1N_3} \cdots c_{11} \cdots c_{1N_3} \cdots a_{41} \cdots a_{4N_3} \cdots c_{41} \cdots c_{4N_3} & P_3 \\ P_{12N_3+4} \cdots P_5 & P_4 \end{array}$$

Fig.4.3 Generalised Embedded Array Code Block Structure.

The outer code D2 is a shortened array code in which the information digits are arranged as shown in Fig.4.4, where P_i is the outer code's parity checks. This arrangement has three advantages:

a) it enables the outer decoder to detect errors within a given sub-block;

b) since each sub-block is checked individually, an erroneous sub-block at a higher level will not result in an error in lower-level sub-blocks;

c) the number of detectable errors associated with each sub-block is increased, thus resulting in a more powerful code.

a_1	b_1	c_1	0	0	0	0	0	0	0	0	0	P_1
0	0	0	a_2	b_2	c_2	0	0	0	0	0	0	P_2
0	0	0	0	0	0	a_3	b_3	c_3	a_4	b_4	c_4	P_3
P_{16}	P_{15}	P_{14}	P_{13}	P_{12}	P_{11}	P_{10}	P_9	P_8	P_7	P_6	P_5	P_4

Fig.4.4 Shortened (52,12) Array Code.

4.2.2 Performance analysis

In this section, the performance of the code is analysed in terms of the system throughput efficiency and reliability [Lin Costello 1984]. The throughput of an ARQ system is defined as the ratio of the number of correct blocks delivered to the user to the total number transmitted, scaled by the rate of the code used. In the theoretical analysis of throughput efficiency, as in other works, the probability of decoding error is assumed to be negligible [Lin Costello 1983 pp462]. Thus the derived expressions are only an upper bound on the system throughput efficiency.

The reliability of an ARQ system, on the other hand, is defined as the ratio of the number of erroneous blocks delivered to the user to the total number delivered. It is worth noting that in many applications the number of detectable error patterns is much greater than that given by the Hamming distance of the code. Consequently, only an upper bound can be theoretically derived.

The theoretical derivations of throughput efficiency and reliability involve the evaluation of expressions for the probability of decoding error P_{de}, the probability of repeat requests P_{rr}, and the probability of correct data P_{cd}, at the decoder output. By inspection of the code structure of Fig.4.3, it can be deduced that each information digit in sub-blocks k1, k2 and the combined sub-blocks k3 and k4, i.e. k3&4, are repeated (N1+N2+N3), (N2+N3) and N3 times respectively. Therefore, if Fi represents the number of correctable bits in the repetition of one bit of ki at the output of the inner decoder, then

$$\begin{aligned} F1 &= \lfloor (N1+N2+N3-1)/2 \rfloor \\ F2 &= \lfloor (N2+N3-1)/2 \rfloor \\ F3\&4 &= \lfloor (N3-1)/2 \rfloor \end{aligned} \qquad (4.4)$$

Let PR_{ei} denote the probability of the ith sub-block being in error at the output of the inner decoder then:

$$PR_{e1} = 1-[\sum_{j=0}^{F1} \binom{N1+N2+N3}{j} P^j (1-P)^{N1+N2+N3-j}]^2 \qquad (4.5)$$

$$PR_{e2} = 1-[\sum_{j=0}^{F2} \binom{N2+N3}{j} P^j (1-P)^{N2+N3-j}]^2 \qquad (4.6)$$

$$PR_{e3\&4} = 1-[\sum_{j=0}^{F3\&4} \binom{N3}{j} P^j (1-P)^{N3-j}]^4 \qquad (4.7)$$

where P is the channel's bit transition probability

and $\binom{i}{j}$ is the jth binomial coefficient= $i!/[j!\ (i-j)!]$

Once the inner decoder has decoded the sub-blocks, the information digits of each sub-block are applied to the outer decoder's buffer as shown in Fig.4.5.

In order to analyse the outer decoder's performance, the output bit error probability of the inner decoder must be determined.

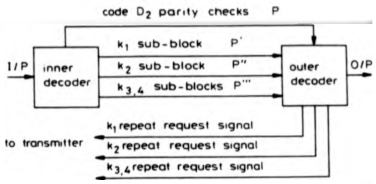

code D_2 parity checks P
I/P
inner decoder
k_1 sub-block P'
k_2 sub-block P''
$k_{3,4}$ sub-blocks P'''
outer decoder
O/P
k_1 repeat request signal
to transmitter
k_2 repeat request signal
$k_{3,4}$ repeat request signal

Fig.4.5 Embedded Array Code, Decoder structure.

Assume now that the inner decoder can be combined with the channel to form a "super channel", as indicated in Fig.4.6. This super channel has four distinct output bit error probabilities P, P', P''and P''', for code D2 parity checks and k1, k2 and k3&4 sub-blocks respectively, as shown in Fig.4.5.

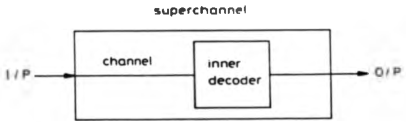


Fig.4.6 Block Diagram of 'Super channel'.

It is clear that the probability of k1 sub-block being correct at the output of the inner decoder, PR_{c1}, is

$$PR_{c1} = 1 - PR_{e1} \qquad (4.8)$$

However, for the "super channel" with BER P', the probability that the sub-block1 is correct must be equal to PR_{c1}, therefore

$$PR_{c1} = (1-P')^{3} \tag{4.9}$$

Using similar reasoning to that employed for k1 sub-block, P'' and P''' can be determined for k2 and k3&4 as follows

$$P' = 1-\sum_{j=0}^{F1} \binom{N1+N2+N3}{j} P^{j} (1-P)^{N1+N2+N3-j} \tag{4.10}$$

$$P'' = 1-\sum_{j=0}^{F2} \binom{N2+N3}{j} P^{j} (1-P)^{N2+N3-j} \tag{4.11}$$

$$P'''=1-\sum_{j=0}^{F3\&4} \binom{N3}{j} P^{j} (1-P)^{N3-j} \tag{4.12}$$

The probabilities of repeat request P_{r1}, undetectable errors P_{e1} and correct data at the output P_{c1}, which have been derived in Appendix C, are as follows:

$$P_{r1}=\sum_{i=0}^{H} \binom{3}{i} P'^{i} (1-P')^{3-i} \left[\sum_{j=0}^{N3+1-i} \binom{1+3 \times N3}{j} P^{j} (1-P)^{1+3 \times N3-j} \right]$$
$$- (1-P')^{3} (1-P)^{1+3 \times N3} \tag{4.13}$$

$$P_{r2} =\sum_{i=0}^{H} \binom{3}{i} P''^{i} (1-P'')^{3-i} \left[\sum_{j=0}^{N3+1-i} \binom{1+3 \times N3}{j} P^{j} (1-P)^{1+3 \times N3-j} \right]$$
$$- (1-P'')^{3} (1-P)^{1+3 \times N3} \tag{4.14}$$

$$P_{r3\&4} =\sum_{i=0}^{3} \binom{6}{i} P'''^{i} (1-P''')^{6-i} \left[\sum_{j=0}^{3-i} \binom{1+6 \times N3}{j} P^{j} (1-P)^{6 \times N3+1-j} \right]$$
$$- (1-P''')^{6} (1-P)^{6 \times N3+1} \tag{4.15}$$

where $H=3$ for $(N3+1) \geq 3$

$H=(N3+1)$ for $(N3+1) < 3$ (4.16)

and

$$P_{c1} = (1-P')^{3}\ (1-P)^{1+3m*3} \qquad (4.17)$$

$$P_{c2} = (1-P'')^{3}\ (1-P)^{1+3m*3} \qquad (4.18)$$

$$P_{c34} = (1-P''')^{6}\ (1-P)^{4m*3-1} \qquad (4.19)$$

and

$$P_{e1} = 1-P_{d1}-P_{c1} \qquad (4.20)$$

$$P_{e2} = 1-P_{d2}-P_{c2} \qquad (4.21)$$

$$P_{e34} = 1-P_{d34}-P_{c34} \qquad (4.22)$$

Fig.4.7 shows the possible decoding paths (or decoding tree) for the acceptance of one block (i.e. four sub-blocks) using the embedded array code.

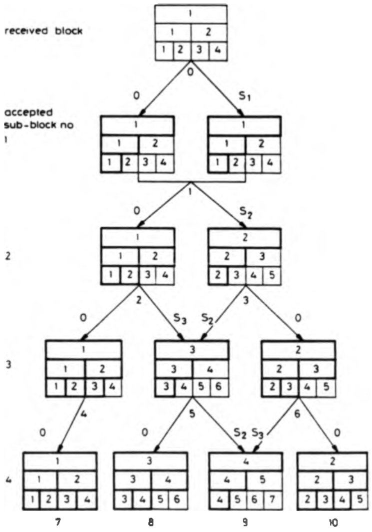


Fig.4.7 Possible Decoding Paths.

The tree consists of four main levels (excluding the first, which is the initial starting level), at each of which, one sub-block is accepted. In order to outline the accepted sub-blocks at each level, these are shown in bold type set.

It is clear from Fig.4.7 that in order to proceed from say, level i to level i+1 ($0 \le i < 4$), sub-block i+1 must be first decoded and subsequently accepted. When decoding this sub-block, the decoder could either accept the sub-block or ask for a retransmission in the manner demonstrated in Fig.4.1. The number of retransmissions required at each level depends on the position of the sub-block in the overall block. This is shown in Fig.4.8, where Si represents the number of repeat requests required for the acceptance of a sub-block in position i. It is apparent that each path probability and the average number of retransmissions needed for successful acceptance of one information sub-block, must be evaluated in order to obtain the average total number of retransmissions.

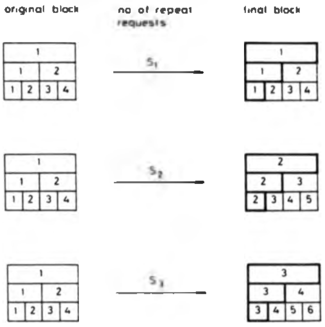


Fig.4.8 Required Number of Repeat Requests for Decoding a Given Sub-Block

Fig.4.9 shows the path nodes and their corresponding probabilities of selection. The probability of choosing these paths is given in Table 4.1 where P_{c1}, P_{c2} and $P_{c3\&4}$ are the probabilities of correctly accepting k1, k2 and k3&4 respectively. The number of retransmissions required for completion of each path must now be evaluated. It is clear that the average total number of retransmissions, Mav is:

$$Mav = PP1 \times M1 + PP2 \times M2 + PP3 \times M3 + \ldots + PP14 \times M14 \qquad (4.23)$$

where PPi is the path probability of path i given in Table 4.1 and Mi is the average required number of retransmissions for the completion of path i, given in Table 4.2.

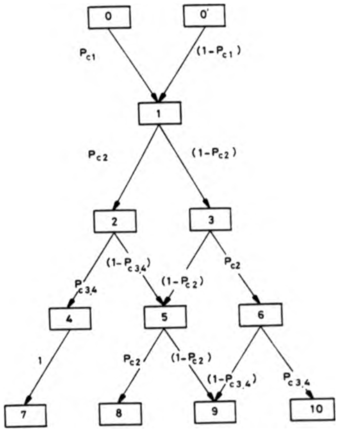


Fig.4.9 Decoding Paths and Their Probabilities.

Possible paths	Path No.	Path Probabilities (PPi)
0 1 2 4 7	1	$P_{e1}.P_{e2}.P_{e3\&4}.1$
0 1 2 5 8	2	$P_{e1}.P_{e2}.(1-P_{e3\&4}).P_{e2}$
0 1 2 5 9	3	$P_{e1}.P_{e2}.(1-P_{e3\&4}).(1-P_{e2})$
0 1 3 5 8	4	$P_{e1}.(1-P_{e2}).(1-P_{e2}).P_{e2}$
0 1 3 5 9	5	$P_{e1}.(1-P_{e2}).(1-P_{e2}).(1-P_{e2})$
0 1 3 6 9	6	$P_{e1}.(1-P_{e2}).P_{e2}.(1-P_{e3\&4})$
0 1 3 6 10	7	$P_{e1}.(1-P_{e2}).P_{e2}.P_{e3\&4}$
0'1 2 4 7	8	$(1-P_{e1}).P_{e2}.P_{e3\&4}.1$
0'1 2 5 8	9	$(1-P_{e1}).P_{e2}.(1-P_{e3\&4}).P_{e2}$
0'1 2 5 9	10	$(1-P_{e1}).P_{e2}.(1-P_{e3\&4}).(1-P_{e2})$
0'1 3 5 8	11	$(1-P_{e1}).(1-P_{e2}).(1-P_{e2}).P_{e2}$
0'1 3 5 9	12	$(1-P_{e1}).(1-P_{e2}).(1-P_{e2}).(1-P_{e2})$
0'1 3 6 9	13	$(1-P_{e1}).(1-P_{e2}).P_{e2}.(1-P_{e3\&4})$
0'1 3 6 10	14	$(1-P_{e1}).(1-P_{e2}).P_{e2}.P_{e3\&4}$

Table 4.1 Path Probabilities.

$M1 = 1 + 0 + 0 = 1$
$M2 = 1 + 0 + S_3 + 0 = 1 + S_3$
$M3 = 1 + 0 + S_3 + S_2 = 1 + S_2 + S_3$
$M4 = 1 + S_2 + S_2 + 0 = 1 + 2xS_2$
$M5 = 1 + S_2 + S_2 + S_2 = 1 + 3xS_2$
$M6 = 1 + S_2 + 0 + S_3 = 1 + S_2 + S_3$
$M7 = 1 + S_2 + 0 + 0 = 1 + S_2$
$M8 = 1 + S_1 + 0 + 0 = 1 + S_1$
$M9 = 1 + S_1 + 0 + S_3 + 0 = 1 + S_1 + S_3$
$M10 = 1 + S_1 + 0 + S_3 + S_2 = 1 + S_1 + S_2 + S_3$
$M11 = 1 + S_1 + S_2 + S_2 + 0 = 1 + S_1 + 2xS_2$
$M12 = 1 + S_1 + S_2 + S_2 + S_2 = 1 + S_1 + 3xS_2$
$M13 = 1 + S_1 + S_2 + 0 + S_3 = 1 + S_1 + S_2 + S_3$
$M14 = 1 + S_1 + S_2 + 0 + 0 = 1 + S_1 + S_2$

Table 4.2 Number of Required Retransmissions for Given Paths.

Referring to Figs.4.7 and 4.9: it is apparent that Mi is the sum of the number of blocks needed in order to proceed to the next node in the decoding tree, i.e.

$$Mi = 1 + \sum_{k=1}^{2} \sum_{j=1}^{3} A_{kj} \times S_j \qquad \text{for } 1 \le i \le 14 \tag{4.24}$$

where $A_{kj}=0$ or 1 (determined by the path) and S_j is the number of blocks required in order to accept data in ki position in the block, as shown in Fig.4.8. Then the number of retransmissions, S_j, is obtained as follows [Yu Lin 1981]:

$$S_1 = 1/P_{c1} - 1 \tag{4.25}$$

$$S_2 = P_{c2} - 1 + 2(1-P_{c2})P_{c1} + 3(1-P_{c2})(1-P_{c1})P_{c1} + \ldots\ldots$$
$$\ldots + (m+2)(1-P_{c2})(1-P_{c1})^{m}P_{c2} + \ldots \qquad m \rightarrow \infty \tag{4.26}$$

$$S_2 = P_{c2} - 1 + (1-P_{c2})(1+P_{c1})/P_{c1} \tag{4.27}$$

$$S_3 = P_{c3} - 1 + 2(1-P_{c3})P_{c1} + 3(1-P_{c3})(1-P_{c1})P_{c1} + \ldots\ldots$$
$$\ldots + (m+2)(1-P_{c3})(1-P_{c1})^{m}P_{c3} + \ldots \qquad m \rightarrow \infty \tag{4.28}$$

$$S_3 = P_{c3} - 1 + (1-P_{c3})(1+P_{c1})/P_{c1} \tag{4.29}$$

Note that the above number of retransmissions do not include the first transmissions.

From Appendix D equation (D.6), with M=Mav, the throughput efficiency is:

$$\text{Efficiency}_{s} = R \times 1/Mav \tag{4.30}$$

where R is the code rate of the embedded array code.

In order to evaluate the reliability of the embedded array code, a probability tree of the possible decoding paths is constructed as indicated in Fig.4.10. It is shown in Appendix B that the error probability of this probability tree, P(E), is given by

$$P(E)= P_{e1}/(1-P_{d1}) + P_{c1}P(H)/(1-P_{d1}) \qquad (4.31)$$

where P(H) is derived in Appendix B.

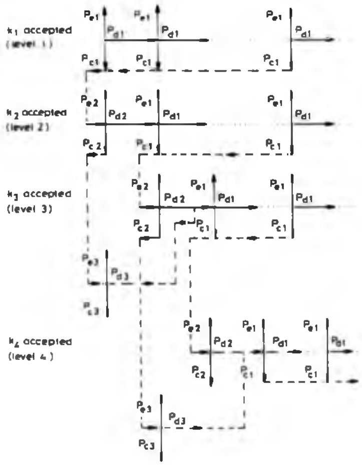


Fig. 4.10 Probability Tree for Embedded Array Code.

It is shown in Appendix D that the throughput efficiency and reliability of an array code of dimensions (n,k,4) is:

$$\text{Efficiency}_A = R_a \times Pc \tag{4.32}$$

and

$$P(e) = 1-(1-P)^n/[1-\sum_{i=1}^{3} \binom{n}{i} P^i (1-P)^{n-i}] \tag{4.33}$$

where R_a is the code rate (k/n)

and Pc is given in Appendix D by expression (D.1).

4.2.3 Results

In this section, a specific example of an embedded array code is considered. The inner codes C1, C2 and C3 are repetition codes of dimensions, (4,1), (2,1) and (1,1) respectively; the outer code is a array code of dimensions (52,36), (ie N1=4, N2=2, N3=1, N=52 and K=36). This choice of codes results in a shortened embedded array code of dimensions (52,12), as shown in Fig.4.11.

a_1	a_1	a_1	a_1	b_1	b_1	b_1	b_1	c_1	c_1	c_1	c_1	P_1
a_1	a_1	b_1	b_1	c_1	c_1	a_2	a_2	b_2	b_2	c_2	c_2	P_2
a_1	b_1	c_1	a_2	b_2	c_2	a_3	b_3	c_3	a_4	b_4	c_4	P_3
P_{16}	P_{15}	P_{14}	P_{13}	P_{12}	P_{11}	P_{10}	P_9	P_8	P_7	P_6	P_5	P_4

Fig.4.11 (52,12) Embedded Array Code Block Structure, with P_i's constructed as in Fig.4.4.

In the case of the embedded array code the reliability and throughput are calculated by the method described below.

Using the procedure explained in Section 4.2.2, the probability of error detection at the outer decoder can be obtained as:

$$P_{d1} = \sum_{i=0}^{2} \binom{3}{i} P'^{i} (1-P')^{3-i} \cdot [\sum_{j=0}^{2-i} \binom{4}{j} P^{j} (1-P)^{4-j}] - (1-P')^{3} (1-P)^{4}] \quad (4.34)$$

$$P_{d2} = \sum_{i=0}^{2} \binom{3}{i} P''^{i} (1-P'')^{3-i} \cdot [\sum_{j=0}^{2-i} \binom{4}{j} P^{j} (1-P)^{4-j}] - (1-P'')^{3} (1-P)^{4}] \quad (4.35)$$

$$P_{d3\&4} = \sum_{i=0}^{3} \binom{6}{i} P'''^{i} (1-P''')^{6-i} \cdot [\sum_{j=0}^{3-i} \binom{7}{j} P^{j} (1-P)^{7-j}] \quad (4.36)$$

$$- (1-P''')^{6} (1-P)^{7}$$

and the probability of correct data at the output of the outer decoder is:

$$P_{c1} = (1-P')^{3} (1-P)^{4} \quad (4.37)$$

$$P_{c2} = (1-P'')^{3} (1-P)^{4} \quad (4.38)$$

$$P_{c3\&4} = (1-P''')^{6} (1-P)^{7} \quad (4.39)$$

A computer program was written to evaluate thc throughput efficiency and the reliability using equations (4.30), (4.31) and Tables 4.1 and 4.2.

In order to compare the performance of the (52,12) embedded code with a conventional array code, the throughput efficiency of a (52,36) array code can be obtained using the techniques given in Appendix D, as outlined below.

$$\text{Efficiency}_{A} = 36/52 \times Pc \quad (4.40)$$

where Pc is the probability of error-free blocks given in Appendix D by equation (D.1).

The code reliability P(e) can be obtained by

$$P(e) = 1 - (1-P)^{n^2} / \left[1 - \sum_{i=1}^{3} \binom{n^2}{i} P^i (1-P)^{n^2-i} \right] \qquad (4.41)$$

The theoretical variations of throughput and reliability with P for both codes are plotted in Figs.4.12 and 4.13 respectively. The results of the simulation study of these codes, under AWGN channel, are also presented in Figs.4.14 and 4.15. Clearly, the theoretical bounds obtained agree closely with the actual results obtained under simulated conditions (with exception of the error patterns that can be detected).

The comparison of the embedded and the array codes shows that the embedded encoding method yields a high reliability at very high bit error rates (e.g. 0.1). Although the throughput efficiency of the embedded technique is greatly reduced due to the added redundancy of the inner codes, for high channel bit error rates (e.g. 0.05-0.1) the embedded system results in an improved throughput efficiency.

In the above discussions, fixed transmission rates (Baud rates) for the both systems have been assumed. It is clear that if the transmission rate of the array code is reduced so that the error free throughputs of the two codes are the same, the reliability of the array code would be superior to that of the embedded code. However the array code would cease to operate effectively at channel BER greater than 10^{-3}, while the embedded code would continue to perform successfully under such conditions.

Fig.4.12 Theoretical Throughput of Array and Embedded Array Codes Under AWGN Channel Conditions.

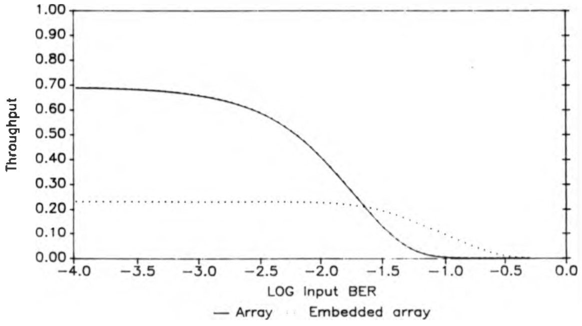


Fig.4.13 Theoretical Reliability of Array and Embedded Array Codes Under AWGN Channel Conditions.

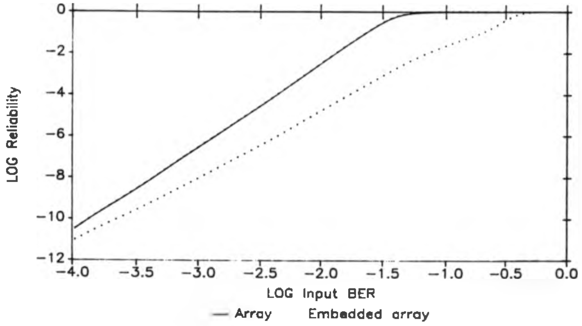


Fig.4.14 Simulated Throughput of Array and Embedded Array Codes under AWGN Channel Conditions

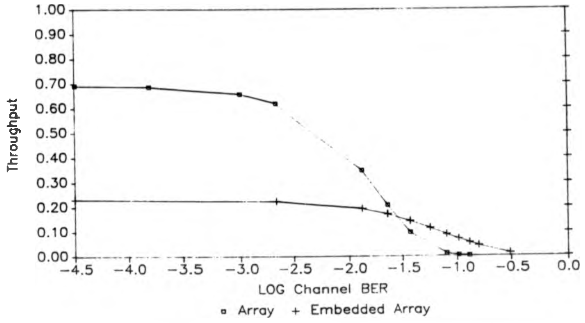
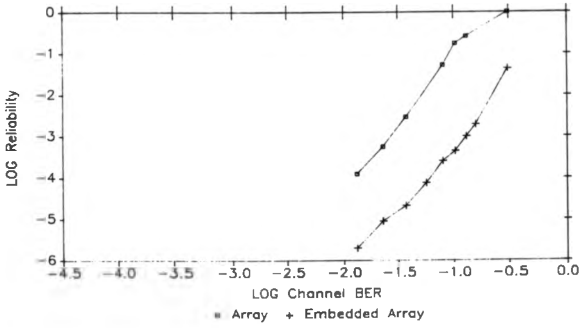


Fig.4.15 Simulated Reliability of Array and Embedded Array Codes under AWGN Channel Conditions



4.3 Modified Embedded Array Code

The embedded array code described in the previous section suffers from low throughput in the regions of low to moderate channel BER. This is the direct result of the additional redundancy of the repeated sub-blocks in the main body of the encoded block. Since in the regions of low channel BER the uncoded information packets are accepted with the greatest probability, under these conditions transmission of the encoded information packets are unnecessary. The modified version of the embedded code [Zolghadr 1987, Darnell Honary Zolghadr 1989] described here overcomes the problem by having new sub-blocks in each stream (see Fig.4.16), where each information packet consists of one information digit.

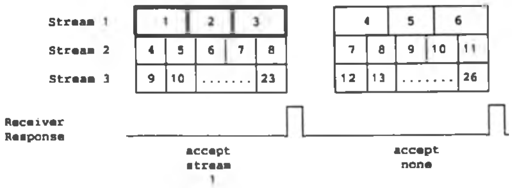


Fig.4.16 Modified Embedded Encoding Scheme

The encoding/decoding procedure and the protocols used for this code are the same as the embedded encoding method. As an example, the (64,23) modified embedded array code is shown in Fig.4.17, where the a_i are the information digits and P_j are the shortened array code's parity checks.

k_1	k_1	k_1	k_1	k_1	k_2	k_2	k_2	k_2	k_2	k_3	k_3	k_3	k_3	k_3	P_1
k_4	k_4	k_4	k_5	k_5	k_5	k_6	k_6	k_6	k_7	k_7	k_7	k_8	k_8	k_8	P_2
k_9	k_{10}	k_{11}	k_{12}	k_{13}	k_{14}	k_{15}	k_{16}	k_{17}	k_{18}	k_{19}	k_{20}	k_{21}	k_{22}	k_{23}	P_3
P_{19}	P_{18}	P_{17}	P_{16}	P_{15}	P_{14}	P_{13}	P_{12}	P_{11}	P_{10}	P_9	P_8	P_7	P_6	P_5	P_4

Fig.4.17 (64,23) Modified Embedded Array Code Block Structure

4.3.1 Performance Analysis

The performance of the original (52,12) embedded code, modified embedded code and also a (52,36) array code have been investigated under simulated AWGN channel condition. The throughput and reliability curves are shown in Figs.4.18 and 4.19. Clearly, for low channel BERs the modified embedded array code has a better throughput characteristics than the original embedded code; however, the throughput in the regions of high channel BER is lower than that of the embedded code. Comparing the throughput curves of these two codes, it is apparent that the desired stable throughput characteristics of the embedded code are preserved in the modified version.

Fig.4.18 Simulated Throughput under AWGN Channel for (52,12) Embedded code and (64,23) Modified Embedded Array Code

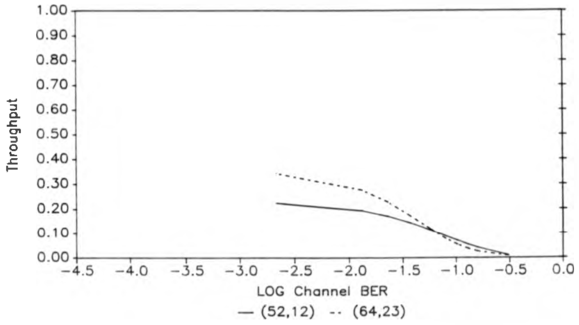
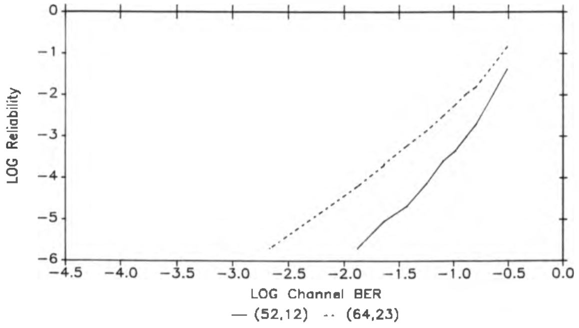


Fig.4.19 Simulated Reliability under AWGN Channel for (52,12) Embedded Code and (64,23) Modified Embedded Array Code



4.4 Discussions

Embedded encoding represents a technique of encoding multiple information packets in the same block in an ARQ system. The information packets are encoded independently, thus allowing a varying number of information packets to be accepted at each block transmission. This arrangement enables the system to operate at very high BER at which the conventional ARQ systems would cease to operate. Furthermore, the embedded code block structure allows the coding scheme to be adapted to the prevailing channel conditions present at the time of the transmission of the block. This is in contrast to the conventional adaptive ARQ systems which adapt the coding scheme in accordance with the channel information gained during the preceding block transmission.

A system of this type has many practical applications in situations where the transmission capacity of communication channel varies with time, e.g. on meteor-burst or HF (2-30MHz) radio links. Apart from its use in very badly disturbed channels, embedded encoding has many other practical applications relating to fixed capacity channels. For example, there are certain situations, e.g. associated with transmissions of vocoded digitised speech, in which the significance of the digits comprising the data stream to be transmitted varies; in the case of vocoder, the significance of, say, the digits describing pitch may be considerably greater than those specifying other parameters of the speech encoding model [Jewett Cole 1978, Goodman Sunberg 1984]. In these circumstances, it is possible to encode the more significant digits of the data stream with the inner code C1, thus minimising the overall distortion. Under these circumstances the embedded code is used in order to match the coding to the signal.

This page has intentionally been left bank.

CHAPTER 5

EMBEDDED CONVOLUTIONAL ENCODING

In this chapter the principles of the embedded convolutional codes [Zolghadr Honary Darnell 1988], which are based on the original embedded encoding format [Darnell Honary Zolghadr 1988], are described. In this implementation of the embedded codes, punctured convolutional codes are used for the purpose of error correction and rate variation. Error detection is performed via a new real time channel evaluation technique which is a by-product of the soft decision, maximum likelihood decoding of the convolutional codes. The performance of both the embedded convolutional encoding and the real time channel evaluation systems have been analysed using simulation techniques.

5.1 Introduction

The embedded convolutional code (EC) [Zolghadr Honary Darnell 1988] described here is based on the embedded block encoding procedure [Darnell Honary Zolghadr 1988] as shown in Fig.5.1. The EC code employs high rate punctured convolutional codes [Cain Clark Geist 1979, Yasuda Kashiki Hirata 1984], whilst maintaining the desirable reliability performance of the embedded codes.

The main block of the code is shown in Fig.5.2: the code comprises three streams, each encoded independently using convolutional codes of rates 1/2, 2/3 and 3/4. Each encoded stream consists of 60 bits, thus the number of information bits in each stream is 30, 40 and 45 respectively. In order to encode the data, three 1/2 rate convolutional encoders are used, the outputs of which are partitioned into blocks of length 60, 80 and 90 bits. Depending on the desired rate of a given stream, a number of bits are deleted from each of the blocks in accordance with a deleting map. The resulting sub-blocks are then transmitted over the channel. This map is also known at the receiver, which is kept in bit synchronisation with the transmitter, thus allowing the receiver to insert erasures for the deleted bits.

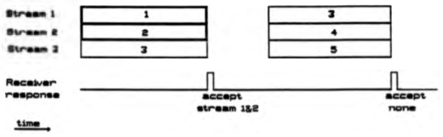


Fig.5.1 ARQ Protocol of the Embedded Convolutional Code

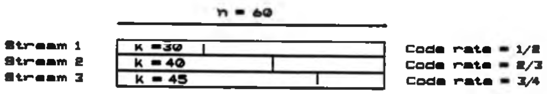


Fig.5.2 Embedded Convolutional Encoded Block Structure

After insertion of the erasure digits, the decoding commences. The decoder comprises two main elements, a soft decision Viterbi decoder and an error limiter. The Viterbi decoder [Viterbi 1971, Forney 1973], also referred to as a maximum likelihood decoder (MLD), is used specifically for error correction. The error limiter, however, limits the output probability of error of the of the overall system. This is achieved by considering the confidences of the path chosen by the Viterbi decoder (correct path) and the most likely error path. The differences between these paths are called the "error detection metrics". The error limiter is then used to initiate ARQ signals for the corrected streams whose averaged error detection metrics fall below a given threshold.

Due to the nature of the decoding procedure of the convolutional codes, in an ARQ situation the decoder buffer must be "flushed". In order to accommodate this requirement, in a repeat request condition, the transmitter re-transmits the rejected streams; this overwrites the old contents of the receiver buffer. The decoding then commences as soon as the decoder buffer is full.

The above error control scheme is another form of adaptive hybrid ARQ, in which the information is encoded in the same block using codes of decreasing error correcting capability. Thus, at relatively low channel SNR, only the stream encoded with the 1/2 rate convolutional code may be accepted, at the expense of low overall code rate. However, when the channel SNR is comparatively high, the streams encoded using the less powerful code are also accepted, resulting in an increase in the system throughput efficiency.

In Section 5.2, an overview of the embedded convolutional system is given. The main system elements are described in more detail in Section 5.3. The simulation results for the EC coding and the RTCE system under additive white Gaussian Noise (AWGN) channel conditions are given in Section 5.4 and are discussed in Section 5.5.

5.2 Embedded Convolutional Encoding

The embedded convolutional encoding is primarily based on the embedded encoding technique which was described in Chapter 4. The code was originally designed to operate at the high error rates encountered on HF communication channel (typically 0.01 to 0.1). Although an improvement in error rate of between two and three orders of magnitude is obtained, the code suffers from low throughput efficiency. This loss in throughput is the direct result of the use of repetition codes which, in practice, are highly sub-optimum. In convolutional embedded encoding, convolutional codes replace the original repetition codes in order to increase the throughput of the embedded codes.

The use of convolutional codes causes a variety of problems. The main difficulty encountered is associated with the decoder, which due to the nature of the convolutional codes has a buffer which is used to decode the incoming data. This implies that the incoming digit can only be decoded if a record of the most recent (say L) received symbols has been kept in the decoder's buffer; therefore given a repeat request condition the decoder's buffer must be flushed, since in an ARQ situation the rejected streams are not retransmitted at the same rate (i.e. in the same position within the encoded block).

This has two implications, firstly, since at any given time the decoder is processing two encoded blocks (i.e. one in the buffer and another used for decoding), if an ARQ situation arises the two blocks must be retransmitted. This results in a reduction of the overall rate of the system. Secondly, the decoding delays involved are higher than those which would have resulted from the use of block code, however, this is a minor disadvantage.

5.2.1 The Overall System

The overall system block diagram is shown in Fig.5.3: the system comprises the following elements:

- data source, generating binary data;
- 1/2 rate convolutional encoder;
- three bit deleting sections;
- transmitter, receiver and the channel;
- 17 level quantizer;
- three erasure inserting sections;
- metrics evaluator;
- 1/2 rate Viterbi decoder;
- error limiting system;

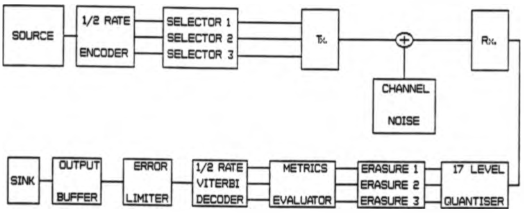


Fig.5.3 System Block Diagram

The convolutional encoder consists of a v-bit shift register, two sets of taps and a set of modulo-2 adders. The tapping points are defined by the generator polynomials of the code, G1 and G2. In the simulation program, the encoder is very general and any choice of G1 and G2 can be selected. For this study, the following pair of polynomials, from a set of near optimum generator polynomials, given in [Larsen 1973] is used:

$$G1 = X^6 + X^5 + X^3 + X^2 + 1 \tag{5.1}$$

$$G2 = X^6 + X^3 + X^2 + X + 1 \tag{5.2}$$

Fig.5.4 shows the resulting convolutional encoder.

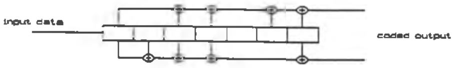


Fig.5.4 Convolutional Encoder for $133_{(8)}$, $171_{(8)}$ Code

The bit deleting sections (the selectors) are used in order to achieve code rate variation, which is required by the embedded code. The selectors operate on the basis of deleting maps [Yasuda, et al. 1983]. In the case of the convolutional embedded code, three such maps are used to delete a selection of encoded bits from each stream, (see Table 5.1) thus giving code rates of 1/2, 2/3 and 3/4.

The transmitter and the receiver operate at baseband using bipolar binary signalling. For the sake of simplicity, the actual transmitted levels are set at ±1 volt, corresponding to the binary digits 1 and 0 (i.e. average transmitter power of 1 Watt in a 1Ω load). The channel used in the simulation studies is an AWGN channel of zero mean and standard deviation of σ.

At the receiver, the output of the demodulator is sampled at the transmitted bit rate. The modulator output is a Gaussian random variable with conditional probability density function of:

$$p(y_i|x_i) = \frac{\exp[-(y_i - x_i)^2/2\sigma^2]}{\sqrt{(2\sigma^2\pi)}} \qquad (5.2a)$$

where y_i and x_i are the received and transmitted signal levels respectively.
Clearly, due to practical limitations, the quantised samples can only take one of finite possible values separated by δx. Consequently, the conditional probability of receiving y_i (in quantised form), $P(y_i|x_i)$ is given by:

$$P(y_i|x_i) = \delta x . p(y_i|x_i) \qquad (5.2b)$$

Let the sequence of m received symbols, $y^{(m)}$, be given by:

$$y^{(m)} = (y_1, y_2, ... y_m) \qquad (5.2c)$$

and one particular output candidate, $x^{(m)}$, be given by:

$$x^{(m)} = (x_1, x_2, ... x_m) \qquad (5.2d)$$

Since each symbol in $y^{(m)}$ is affected independently by noise, a likelihood function can be formed, given by:

$$Pi(y^{(m)}|x^{(m)}) = \prod_{i=1}^{m} P(y_i|x_i) \qquad (5.2e)$$

For optimum detection (or decoding) of $y^{(m)}$, the above likelihood function must be maximised [Viterbi 1971]. In practice however, due to practical limitations (e.g. complications of multiplication, etc.) a more appropriate likelihood function, the log-likelihood function, is used. The log-likelihood function overcomes these problems and since log is a monotonic function of its argument, optimum detection also involves maximising the log-likelihood function. The log-likelihood function can be derived by taking natural logarithms of equation (5.2e), with appropriate substitution from equations (5.2a) and (5.2b) which yields:

$$\log(Pi(y^{(m)}|x^{(m)})) = A \,.\, \sum_{i=1}^{m} y_i.x_i - B \qquad (5.2f)$$

where A and B are constants depending on the state of the channel, the base of the logarithm used and the quantisation step.

Clearly, equation (5.2f) describes the function of a maximum likelihood soft decision detector (or decoder). Thus an optimum detector forms the inner product between the samples of the received sequence (sampled at the symbol synchronisation point) and the channel symbol (in this case the transmitted bipolar signals (± 1).

In the communication system described here the sampler operates in the following fashion: the input analogue voltage is first limited to ± 1 volt and then sampled using the offset binary format. The resulting samples then vary between 0 and 16, corresponding to a high confidence '0' and '1' respectively. On this scale an erasure is represented by the confidence level 8. It is worth noting that equation (5.2f) assumes an infinite sampler dynamic range. In practice however, the limited dynamic range of the sampler causes signal clipping. For a more exhaustive treatment of this subject see [Honary Ali Darnell 1989]

The erasure inserter sections are used to restore the incoming streams to their original length, thus allowing the 1/2 rate decoder to be used. Since the nature of the deleted digits is not known at the receiver, the receiver inserts erasures (don't know digits) in place of the deleted bits, using the same deleting maps used at the transmitter. It is clear that bit and frame synchronisation must be established before reliable decoding can commence.

The metrics evaluator generates one complete trellis of metrics for every pair of soft decision data. The metrics are given by

$$M0(n) = Mc - C(n) \tag{5.3}$$

$$M1(n) = C(n) \tag{5.4}$$

where M0(n) and M1(n) are the metrics associated with '0' and '1' for symbol n, respectively, Mc is the maximum confidence (i.e. 16) and C(n) is the quantised level of the incoming digit in time slot n. Fig.5.5 shows the metric assignment for the soft decision data. Here a high confidence bit is allocated the metric 16, a low confidence bit the metric 0 and an erasure is represented by the metric 8. To each branch of the trellis, a branch metric is given which is the sum of the metrics of the pair of digits on that branch.

Clearly, the above metric description is identical to the one given by equation (5.2f). Previously, the sign of the sampled demodulated output is varied in accordance with the branch symbol, whereas here the sign of the demodulator output C(n) is stripped and the confidences of both zero and one are determined according to equations (5.4) and (5.5).

The Viterbi decoder, is then applied to perform error correction on a finite sample of the past trellises. The output of the decoder is the decoded data and its associated confidence metric (error detection metric).

The error limiter, limits the output probability of error of the system, through the use of the error detection metrics. These metrics are used to determine the highest rate code which can be accepted,

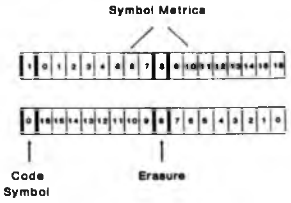


Fig.5.5 Metric Assignment for Soft Decision Data

---

without exceeding a particular output probability of error. A re-transmission of the unacceptable sub-blocks is initiated if none of the available rates can be accepted. The format of the retransmitted sub-blocks is identical to the original embedded encoding format described in [Darnell, Honary and Zolghadr 1988].

It is clear that, due to the varying encoding rate of the streams and the nature of the embedded encoding, a data sink buffer is required to realign the decoded sub-blocks.

5.2.2 ARQ Format

Deployment of convolutional codes in ARQ systems cause major protocol difficulties. As mentioned previously, the MLD, by its very nature, requires the storage of a large number of the past trellises. This does not result in any difficulties in FEC systems, since the presence of the buffer only leads to a fixed decoding delay.

In an ARQ system, however, the decoder buffer dictates a need for data buffering and more complex protocol. This problem is somewhat amplified, when the embedded encoding format is employed in the ARQ system.

As described in Section 5.1, in embedded convolutional encoding, the output of the encoder is composed of three different rate convolutional codes. At the decoder the MLD uses three buffers to store the most recent L_i, trellises (where, i is the stream number and L is the decoder search length). The search lengths, L_i, are arranged to be equal to the block size of the encoded streams plus one (i.e. 31, 41, 46). Therefore, decoding can only commence when one complete block has been stored in the MLD buffers. This arrangement simplifies the ARQ protocol, since at the end of decoding any given block, the most recently transmitted block will be in the buffer; at this point the error limiter can decide whether the decoded stream is acceptable. The error limiter can arrive at four possible decisions, these are:

- Accept the whole block;
- accept stream 1;
- accept stream 2;
- accept none of the streams.

If all the streams are accepted, the decoding continues as normal and the transmitter sends the next encoded block in the queue. In the case when only a segment of the block is accepted, the decoder prepares for its buffers to be flushed. This involves resetting all the decoder variables and pointers. In this situation the first received block is used to overwrite the contents of the decoder buffers and the decoding commences as soon as the second block is received.

In an ARQ situation the encoder behaves somewhat differently. If the transmitted block is accepted, the normal mode of operation is continued. The repeat request signal from the transmitter, causes the rejected streams to be encoded with a more powerful code. Depending on which of the streams has been accepted, the information digits in the rejected streams are collated with the addition of some new data (if required), to form a new data block of 115 digits (i.e. 30, 40 and 45 for streams 1, 2 and 3 respectively). In addition to this, the last 115 data digits immediately preceding the new data block are also collated. The two data blocks are then segmented and subsequently encoded, using the format described in Section 5.1. At the receiver, the first data block is used to fill the decoder buffers, prior to transmission of the new data block.

Although the above protocol is not optimum, no other feasible method of retransmission procedure has been developed, which would allow the system to operate without a staggering amount of complexity.

5.3 Main System Elements

In this Section a more detailed description of the convolutional encoding system is given.

5.3.1 Encoder

A binary 1/2 rate convolutional encoder is a linear finite state sequential circuit with elements 0 and 1. The encoder can be thought of having v+1 shift registers (where v is the constraint length of the code) with various stages. The outputs from each stage of the shift register are connected in a certain pattern to v two input modulo-2 adders.

The block diagram of the convolutional encoder used for this study of the embedded convolutional code is shown in Fig.5.4. The encoding procedure is as follows:

- the contents of every stage of the shift register is set to zero, prior to the start of encoding;
- the data bits are shifted into the shift register one bit at a time, at every time slot;
- at every time slot, the oldest bit in the shift register (corresponding to the input data, v time slots ago) is lost;
- the two outputs of the encoder are then taken to be the encoded bits (symbols) for that time slot.

Three such encoders are used in the embedded convolutional encoder, one for each of the streams. The choice of the generator polynomial is dependent on the decoder complexity and the channel conditions; for this reason in the simulation program any valid choice of the generator polynomial is acceptable (maximum shift register length of 30 bits).

In order to form high rate convolutional codes from the original half rate code, the output of the encoders are passed to three selector modules. The selectors, periodically delete a number of the encoded bits, depending on a set of deleting maps.

The selectors operate on the basis of deleting maps. Q blocks (2Q bits) of the original encoded data are periodically chosen, from which m bits are deleted according to the deleting maps; the remaining (2Q-m) bits are transmitted. The code rate of the resulting punctured code is then given by

$$R=Q/(2Q-m) \qquad (5.5)$$

In the case of the convolutional embedded code, three such maps are used to delete a selection of encoded bits from each stream, thus giving code rates of 1/2, 2/3 and 3/4. Obviously for the 1/2 rate code no bits are deleted; however, a separate selector module is provided for this stream for completeness. For a given required code rate there are two deleting maps. These operate on the outputs of the two generator polynomials independently. The deleting maps used in the simulation studies are given in Table 5.1, where a '0' corresponds to a deletion.

	DELETING MAP FOR			
CODE RATE	G1	G2	N	m
1/2	1	1	1	0
2/3	11	10	2	1
3/4	110	101	3	2

Table 5.1 The Deleting Maps for the Embedded Convolutional Code

5.3.2 The Channel

For the purpose of this simulation study the modulator and the channel both operate at baseband. The communication system described here employs a zero mean AWGN channel. At every bit time, depending on its binary input, the modulator outputs +1 or -1 volts corresponding to binary digit 1 or 0 respectively. The output of the modulator is then fed to the channel simulator.

The AWGN simulator generates a sample of AWGN every bit time in synchronous with the incoming bits. The generated noise samples are then added algebraically to the channel input voltages. The resulting composite signal (i.e. the received voltage) is then output. The signal to noise ratio of the system is set at this point in the system by selecting a value for the standard deviation of the noise, σ. It is clear that the rms noise voltage is given by σ, assuming a 1Ω load the noise power is then given by σ^2. Therefore the signal to noise ratio of the system is given by:

$$\text{SNR (dB)} = -20 \log(\sigma) \qquad (5.6)$$

5.3.3 Metrics Evaluator

The metrics evaluator is used in conjunction with the soft decision decoder to perform error correction. The input to this unit is provided by the analogue to digital converter (ADC). At the receiver, at each bit time, the output of the channel simulator is sampled and quantised into 17 levels. These values are sent to the metrics evaluator in order to obtain the corresponding metric values for a particular receiver symbol.

Since the basis of the system is a half rate convolutional code, for every input data bit at the transmitter, one channel symbol consisting of two bits is generated. The metrics evaluator therefore, generates the appropriate metrics once two consecutive bits have been received.

The metrics evaluator utilises two look-up tables. The first is used to decide which branches of the trellis are the legal transitions. The second is used to inform the unit of the actual encoder output corresponding to any given branch. This information is used to allocate metrics to the branches of the trellis.

Once the quantised values corresponding to one channel symbol have been entered into the metrics evaluator, the following steps are performed:

- decide on the possible transitions from the initial node;
- allocate metrics for all possible branches leading off that node;

- repeat the above steps for all possible nodes in the trellis;
- once completed, output the trellis to the decoder.

The relationship between the input quantised values and the bit metrics are given by equations 5.3 and 5.4. The branch metrics are then simply the result of the addition of the metrics of the two digits on any branch.

5.3.4 Soft Decision Decoder

The decoder performs two main functions, these are:

- Error correction within each sub-block,
- Evaluation of confidence metric for the output decoded data.

The first function is realised through the use of the Viterbi algorithm (VA) operating on a time division multiplexed basis among the three streams. In order to describe the operation of VA, a more detailed description of the encoder is required.

It was assumed that a convolutional code of rate 1/2 is used with two generator polynomials G1 and G2. Let the input to the encoder at time interval i be a_i then the output of the encoder, B_i, is pair of bits b_{i1}, b_{i2}. It is clear that B_i is not only dependent upon a_i but also on the previous v bits (i.e. $a_{i-v} \ldots a_{i-1}$) where

$$b_{i1} = X1(a_{i-v}, a_{i-v-1}, \ldots . a_i) \qquad (5.7)$$

$$b_{i2} = X2(a_{i-v}, a_{i-v-1}, \ldots . a_i) \qquad (5.8)$$

It is convenient to visualise the encoder as a finite state machine with 2^v states; where the state of the process is purely determined by the v past inputs (i.e. memory elements) and the transition between states is conditioned on the present input a_i. Fig.5.6 shows a compact representation of this process, in the form of a trellis diagram for a $7_{(8)}$, $5_{(8)}$ convolutional code. A sequence of binary input data is then encoded as a path through the trellis.

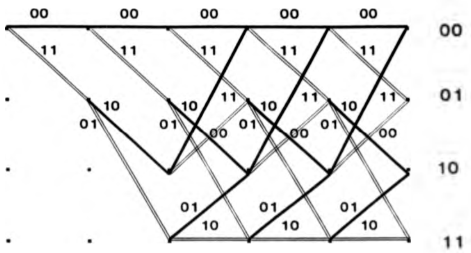


Fig.5.6 Trellis Diagram for $7_{(8)}$, $5_{(8)}$ convolutional code.

Noise in the channel corrupts the sequence of the transmitted channel symbols T_i = { t_{i1}, t_{i2} }. It is assumed that the sequence of the received symbols R_i = { r_{i1}, r_{i2} } is given by

$$R_i = T_i + Z_i \qquad (5.9)$$

where Z_i = { z_{i1}, z_{i2} } are independent zero mean Gaussian random variables of variance σ^2. The VA then attempts to find the greatest likelihood path through the L_t consecutive trellises given the received sequence R_i.

The Viterbi algorithm attempts to maximise the sum of M(i,j) over the search length, L. This is achieved by considering all the possible paths through the trellis. For each path the branch metrics are summed and the path with the highest metric is chosen as the greatest likelihood path through the trellis. Once this path is found, the decoder outputs the data bit which when encoded, would have lead to the first branch of the path. The decoding window, which is of length L, is then advanced by one trellis and the above operation is repeated. Ideally the search length of the decoder, L, should tend to infinity, however this would lead to unacceptable decoding delays. For this reason, L, is usually limited to about five times the constraint length of the code [Yasuda, et al. 1983] (i.e. $L \geq 5v$).

The structure of the embedded convolutional code results in the need for three Viterbi decoders. In the embedded convolutional encoding system however, only a single Viterbi decoder is used. For each stream of the code a separate decoder buffer is allocated; these buffers are switched in turn as the main decoder buffer. The lengths of the buffers are set to be equal to the stream lengths after the erasure inserter section, plus one (i.e. 31, 41, 46 for streams 1, 2 and 3 respectively). The buffers are implemented as three dimensional arrays of dimensions (Y, B, L_i), where, Y is the number of states (i.e. 64), B is the number of branches leading to a state (i.e. 2) and L_i is the decoder search length for stream i.

The second function of the decoder is to assign error detection metrics to the output bits (decoded). These metrics are derived form the original decoding processes by subtracting the confidence of the greatest likelihood path form that of the greatest likelihood error path.

5.3.5 Real Time Channel Evaluation System (Error Limiter)

The function of the error limiter is to initiate ARQ signals in order to limit the output probability of error of the system. This is achieved by utilising the Viterbi decoder even further by making use of the soft decision information within the decoder, which would have otherwise been lost.

As a by-product of the Viterbi algorithm, the accumulative confidence metric of the greatest likelihood path (correct path) is readily available. The RTCE system retrieves this information and causes the decoder to make an additional pass through the trellis in search of the greatest likelihood error path (second path).

In order to find the second path, the same Viterbi decoder is used, however, the decoder is not allowed to search all the possible branches. Clearly, for the second path to be the nearest error path to the correct path, it must originate from the same state as the correct path; furthermore, the second path should at least differ from the correct path by one branch. This branch must include the branch corresponding to the transition from the initial state to the next state, since it is only discrepancies in this portion of the path that leads to an erroneous digit at the output of the decoder. The decoder therefore is not allowed to pursue paths emerging from the first branch of the correct path.

It is evident that when the channel conditions cause an incorrect path to be chosen, the error detection metrics are reduced in magnitude. In the worst case the value of these metrics is reduced to zero indicating a total lack of confidence in the decoded bit.

Figs.5.7, 5.8 and 5.9 show the probability density functions (pdf) of the error detection metrics at the output of the Viterbi decoder, for the 1/2 rate code (channel SNR of 2dB, 3dB and 4dB sample size of $10^{[illegible]}$ input bits) with the decoder search length, L, set to 31. Evidently, the mean of the pdfs are distinct. Fig.5.10 shows the variation of the mean value of the error detection metrics and their standard deviations, with respect to the channel SNR. From Fig.5.10 it is clear that the error detection metrics are a good indication of the level of noise in the channel.

The probability of error of the Viterbi decoder is dependent on the channel signal to noise ratio; therefore, once the channel SNR ratio is determined, an estimate of the output probability of error from the decoder for each of the three codes can be obtained.

Fig.5.7 pdf of the Error Detection Metrics for Channel SNR=2dB, Rate 1/2.

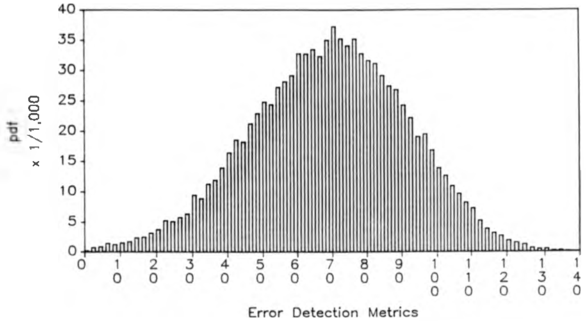


Fig.5.8 pdf of the Error Detection Metrics for Channel SNR=3dB, Rate=1/2.

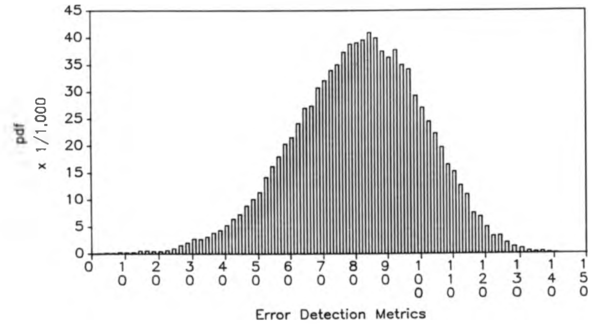


Fig.5.9 pdf of the Error Detection Metrics for Channel SNR=4dB, Rate=1/2.

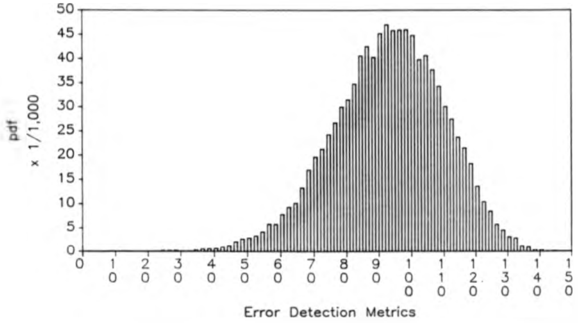


Fig.5.10 Mean and the Standard Deviation of the Error Detection Metrics vs Channel SNR (dB).

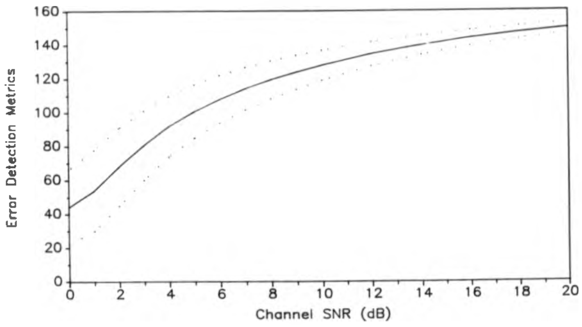
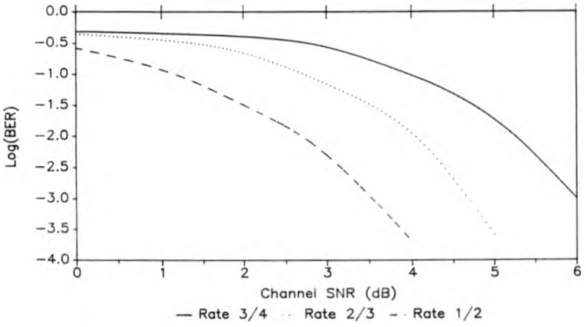


Fig.5.11 Output BER vs Channel SNR (dB) for Code Rates 1/2, 2/3 and 3/4 Convolutional Code.



The error limiter therefore averages the error detection metrics, say over m bits, where m would be chosen in accordance with the standard deviation of the error detection metrics. The average is then compared with a set of previously obtained thresholds. The thresholds are arranged such that, the overall output probability of error of the system is kept below a specified maximum value.

Fig.5.11 shows the effect of the channel SNR on the probability of error (BER) of the three convolutional codes. As an example, if the required output BER was required to be below 10^{-3} the following boundaries would be used: when the channel SNR is less than, say 3.5

dB, none of the streams are accepted, thus improving the reliability of the system. At higher channel SNR (3.5dB < SNR <4.5dB) stream 1 is accepted. It is clear that at this range of SNR, the overall probability of error of the system is bounded by the performance of the 1/2 rate code. Both streams 1&2 are accepted at higher values of SNR (i.e. 4.5dB < SNR < 6dB). Stream 3 however is accepted at SNR > 6dB, where the rate of the overall system is 0.64.

5.4 Performance Study

The results are in the form of throughput and reliability of the system in the presence of AWGN. Fig.5.12 shows the variations of the system throughput with channel SNR for the following system parameters:

- decoder search lengths, L, of 31, 41, 46 for code rates 1/2, 2/3 and 3/4 respectively;
- error limiter averaging length, m, set at 30.

Fig.5.13 depicts the overall reliability of the system versus channel SNR. As can be seen by comparison with Fig.5.11, the reliability of the embedded convolutional code is better than that of the 2/3 and 3/4 rate convolutional codes. The throughput of the system is greater than that of the 1/2 rate convolutional code for SNR > 7dB.

In order to investigate the effectiveness of the embedded convolutional code, an equivalent system employing the same code structure, primary encoder, ARQ protocol and decoder has been simulated, the main difference being that all the three streams have code rate of 1/2. The results of this study are also shown in Figs.5.12 and 5.13 which depict the comparison of the throughputs and reliabilities of the these two codes, respectively.

Fig.5.12 Throughput of the Embedded Convolutional and the Comparison Codes.

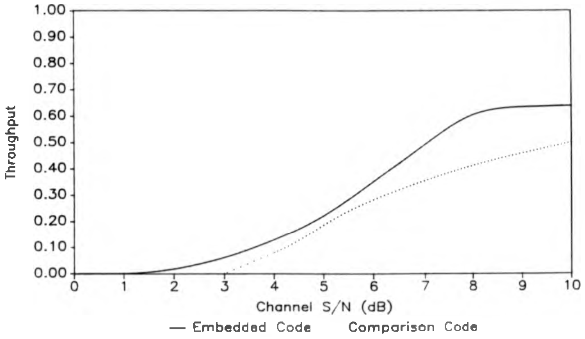
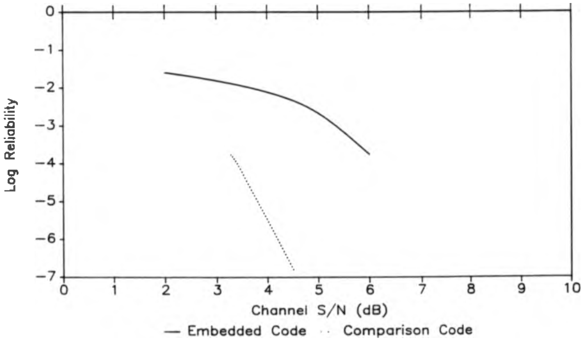


Fig.5.13 Reliability of the Embedded Convolutional and the Comparison Codes.



5.5 Discussions

Embedded convolutional codes are a combination of forward error correction (FEC) and modified ARQ systems. The ARQ is achieved through further analysis of the decoding trellis, thus extracting additional information from the Viterbi decoding procedure.

These codes perform well in environments where the channel SNR varies continuously. The results show that the overall throughput of the system is improved for channel SNR > 7dB, compared with the fixed rate FEC convolutional code (e.g. rate 1/2). Although, for channel SNR < 7dB the EC coding proves less efficient, this must be viewed in the context of a time variable channel where the standard deviation of the channel SNR is quite high. In this context, the EC coding would prove much more superior compared with conventional FEC systems. This in fact would be the direct result of the error limiting algorithm, used in conjunction with an ARQ architecture.

The method of channel evaluation described here has been shown to improve the reliability of conventional FEC convolutional codes, when used in and ARQ system. Moreover, this method can be extended for use as the control mechanism for simple "code-swapping" systems employing convolutional codes.

Even though the work described in this chapter was carried out independently of other researchers, recent literature search has shown that some attention has been given to this topic by other authors [Yamamoto Itoh 1980]. Although the system described in [Yamamoto Itoh 1980] attempts to detect uncorrectable errors, in an ARQ format, it differs from the work outlined here in the following points:

a) The growth of path metric differences on a running basis are considered. The difference is taken at each node between the two most confident paths, rather than the difference between the chosen path and the nearest error path, at the end of the decoding pass.

b) The ARQ protocol also differs from the one employed here. In [Yamamoto Itoh 1980], the repeat request in initiated at irregular intervals. Thus, the number of retransmitted digits is variable. Whereas here, the transmitted data stream is segmented into blocks, the advantage being the reduction in system complexity (both at the encoder and decoder) and reduced susceptibility to feedback errors.

This page has been intentionally left blank.

CHAPTER 6

MULTIPLE FREQUENCY SHIFT KEYING MODEM

Until recently, digital modems have been implemented as a direct translation of analogue design techniques to digital methods [e.g. Ralphs 1977]. The main draw back of this approach is the loss in performance caused by neglecting the powerful purely digital techniques.

As a by-product of using a completely digital modem implementation approach, numerous improvements can be achieved. In line with this philosophy and in the context of multi-functional coding [Darnell Honary 1986], a new digital MFSK modem has been developed. The modem employs a novel digital processing procedure termed code-assisted bit synchronisation (CABS) [Zolghadr 1988, Zolghadr 1989a, Honary Zolghadr Darnell 1989, Honary Zolghadr Darnell Maundrell 1989]. Signal detection is achieved via a set of non-coherent correlators, while symbol synchronisation and error correction are performed by a soft-decision Viterbi decoder [Viterbi 1971]. Practical tests of the modem have shown that it operates effectively under both additive white Gaussian noise and real HF channel conditions.

6.1. Introduction

Recent advances in digital electronics have led to the development of cheap and powerful digital signal processing (DSP) equipment (e.g. the TMS320c25 DSP chip); consequently an all digital modem implementation has become a feasible proposition.

In conventional communication systems various signal processing subsystems are employed [e.g. Stremler 1982], each performing a particular task, e.g. carrier recovery, demodulation, symbol timing recovery, error control, etc. Channel noise (man-made or natural) causes a variety of restrictions on the operation of these subsystems. These restrictions normally manifest themselves in the form of channel errors which, in essence, are the result of an amalgamation of many contributory effects, including:

a) noise;

b) variations in wanted signal level due to fading;

c) intersymbol interference;

d) carrier frequency and phase mismatch between the transmitter and the receiver;

e) frequency and phase ambiguities between the transmitter and the receiver data clock circuitries.

Conventionally, the channel errors are removed by appropriate error control coding. Although this approach is effective when powerful and complex error control coding is used, it normally takes little account of the underlying error generating mechanisms affecting the communication system.

In an all-digital modem, the operation of the individual processing sub-blocks have the potential to be combined, thus providing a more efficient and effective means of combating these errors. The modem described here is one such implementation, in which the functions of the demodulator, decoder and symbol timing recovery elements have been integrated. The complete modem is implemented on a single DSP board, operating in real-time at a rate of 250 baud. It is accepted that this involves a symbol length (4 ms) which is less than optimum in terms of inherent performance against severe multipath distortion [Ralphs 1977, Stein 1987]; however this modem may be considered to be a system demonstrator.

The principles, implementation and performance of this modem based on a 4-tone multiple frequency shift keying (MFSK) signalling format [Ralphs 1977, Ralphs 1985] are described in the following sections. The chapter is organised in the following manner: firstly, in Section 6.2, an overview of some common symbol synchronisation systems is given. In Section 6.3, the principles and the implementation of the modem are described in detail. Section 6.4 describes modem verification tests and the results of the simulation test and practical trials of the modem under both additive white Gaussian noise (AWGN) and real HF channel conditions.

6.2. Overview of Some Common Synchronisation Methods

Symbol timing recovery is one of the most critical functions in a synchronous communication system. Its purpose is to indicate the optimum sampling instant for data state decision within the received symbol interval [Mueller Müller 1976, Franks 1980]. It is therefore essential that the receiver and the transmitter clocks are kept in good phase and frequency alignment if erroneous decisions are to be avoided.

There are several methods of achieving symbol synchronisation for MFSK communication systems operating over the HF channel. These can be classified into four groups [Ralphs 1985 pp120-131]:

a) continuous synchronisation;

b) periodic synchronisation;

c) one-shot synchronisation;

d) intrinsic synchronisation.

In many practical communication systems the transmitted data is conveyed by modulating a constant amplitude carrier tone, e.g. via FSK and PSK. When the transmitted modulated waveform is nominally of constant amplitude, the synchronisation signal can be transmitted by an additional superimposed low-level amplitude modulation. The symbol synchronisation signal is therefore conveyed simultaneously with the transmitted data.

An example of continuous synchronisation was the PICCOLO system [Ralphs 1977, Ralphs 1985] in which the MFSK modulated carrier signal is amplitude modulated (10%) with a square wave signal at the symbol rate. At the receiver the synchronisation information is extracted using a peak detector. This technique suffers from a number of disadvantages, including:

i) degradation of the matched filter response, due to the introduction of correlation between the transmitted tones (giving an approximate overall performance loss of 1dB);

ii) long initial synchronisation response time (approximately 15-20s), due to the limited bandwidth of the timing control loop;

iii) other complications related to constraints on the actual receiver parameters (e.g. AGC time constant etc.);

iv) susceptibility of the system to synchronisation errors due to both selective fading and flat fading conditions.

In periodic synchronisation [e.g. Hague Jowett Darnell 1988], the receiver clock is synchronised to that of the transmitter at regular time intervals. This is achieved by allocating a single tone for the purpose of synchronisation. In this arrangement a synchronisation signal is transmitted on this tone at regular intervals, with the time intervals between corrections depending on the stability of the transmitter and the receiver clocks and the propagation path characteristics.

There are two main disadvantages in such a system: firstly, the initial synchronisation time is long due to the time response of the detection arrangement; secondly, the added redundancy of the extra tone degrades the overall performance of the system in terms of occupied bandwidth and transmitter power utilisation efficiency.

In certain types of channels, in which a single and very stable mode of propagation dominates (i.e. stable single mode channels), it is possible to dispense with the need for continuous synchronisation. Symbol synchronisation over such channels can be achieved by "one-shot" techniques; here, complete initial synchronisation is established prior to data transfer and the system then relies on the stability of the system clocks to ensure symbol timing integrity over the transmission interval.

Although this scheme has good bandwidth and power utilisation efficiency, it suffers from two major disadvantages, namely:
i) any variations of the absolute propagation time of the transmitted signal must be small, although this is not a problem at HF if relatively long symbol lengths are used [Ralphs 1985];
ii) high clock stability is required both at the transmitter and the receiver, in order to ensure sync. during long data transfers.

Intrinsic synchronisation is concerned with deriving the symbol timing signal directly from the received data. In the system of interest (i.e. MFSK) the frequency shifts caused by the transmitted data are used to derive the synchronisation information. This scheme is of great interest, since it overcomes the problems of the above systems. Intrinsic synchronisation is highly suited to communication paths in which many modes of propagation exist (e.g. the HF channel). In these channels the absolute signal propagation time is constantly varying due to the changes in the dominant propagation mode of the received signal. These changes can be effectively tracked by intrinsic synchronisation methods.

To date the main draw back of such systems has been the limitation of the analogue implementation methods employed. Consideration of the practical parameters and limitations of analogue circuitry has shown that a relatively poor approximation to the synchronisation waveform can be obtained using such methods. Therefore, until recently, this method of symbol timing recovery has been impractical; with the advent of fast and powerful DSP chips however it is now possible to obtain reliable symbol synchronisation via intrinsic synchronisation. An example of this type of procedure is CABS [Zolghadr 1989b, Honary Zolghadr Darnell 1989], which is described here.

6.3. Principles of the CABS System

In this section the concept of code assisted bit synchronisation (CABS) is described. CABS is a form of intrinsic synchronisation in which the frequency transitions due to the modulating data are used to derive symbol timing information. In CABS, the processes of demodulation and decoding are combined; hence, the demodulator in practice also performs the task of error control. In combining the processes of demodulation and decoding, a half-rate binary convolutional code is used in conjunction with a soft-decision Viterbi decoder [Viterbi 1971]. Symbol timing information is then derived from the Viterbi decoder which also performs error correction on the received data. The block diagram of the modulator is shown in Fig.6.1; it comprises:

a) 1/2 rate convolutional encoder;

b) 4-tone MFSK modulator.

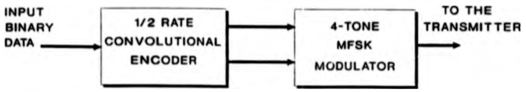
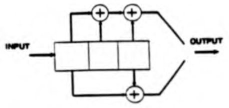


Fig.6.1 CABS Modulator Block Diagram

The 1/2 rate convolutional encoder used is shown in Fig.6.2. The encoder has a constraint length of 3 and consists of a 3-bit shift register, two sets of taps and a set of modulo-2 adders. Although the free distance of this code is 5 [Sklar 1988 pp349], its short constraint length allows a simple decoder structure.



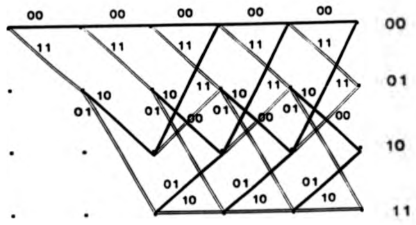


Fig.6.2 CABS System Convolutional Encoder and

Trellis Diagram

The input binary data to be transmitted is fed to the convolutional encoder, the output of which is a pair of binary bits, (i.e. the codeword C, where C ε (00, 01, 10, 11}). These codewords are used to select one of four possible baseband tones, the selected tone being transmitted for precisely one symbol interval T_s, with the following codeword to channel symbol assignment:

codeword	tone
00	f_1
01	f_2
10	f_3
11	f_4

A block diagram of the CABS demodulator comprises two main elements:

a) four noncoherent correlators;

b) soft-decision Viterbi decoder.

The block diagram of the noncoherent correlator [e.g. Whalen 1971 pp200-201] is shown in Fig.6.3. These correlators are each matched uniquely to one of the four assigned tone frequencies, with a correlation window equal to the transmitted symbol interval, T_s.

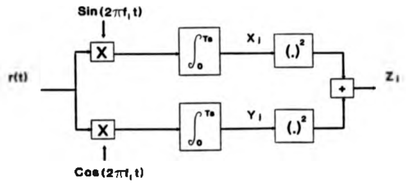


Fig.6.3 Block Diagram of the Noncoherent Correlator

The correlators operate by sampling the received signal at a frequency of f_s in order to evaluate the inphase (I_{ij}) and the quadrature (Q_{ij}) components of the received signal, where i is the tone number $(1 \le i \le 4)$ and j is the sample number within an arbitrary symbol $(1 \le j \le N)$ and where

$$N = T_s . f_s \qquad (6.1)$$

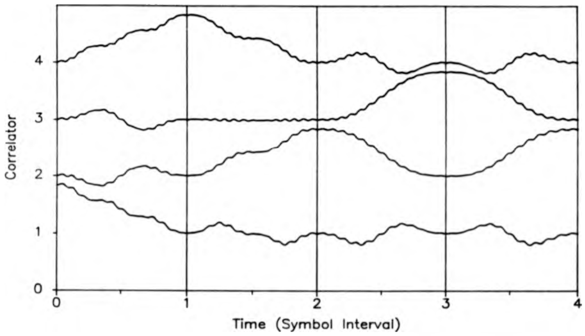
This is achieved by multiplying the incoming signal by sine and cosine functions generated locally at the four possible tone frequencies. The outputs of the multipliers are then summed over the preceding interval corresponding to the symbol duration. The inphase and quadrature components obtained in this manner are squared, summed and finally square-rooted to produce the desired output of the correlator. This process is carried out at every sampling point; thus the summed output of the correlators at each sampling instant is given by:

$$M_{ik} = \sum_{j=k-N+1}^{k} (I_{ij}^2 + Q_{ij}^2) \qquad (6.2)$$

where M_{ik} is the magnitude of the output of correlator i at sample reference number k $(1 \le k \le N)$, within an arbitrary symbol interval. Fig.6.4 shows the outputs of the individual correlators for 5 successive received symbols, where the received symbols are tones 1, 4, 2, 3 and 2 corresponding to the source data of 0, 1, 1, 1 and 0 respectively. It is clear from Fig.6.4 that the output of a given correlator rises to a relatively high level when the correlator is matched to the received symbol frequency.

The main problem with any digital demodulation process is to derive the optimum time instant for sampling the correlator outputs. From Fig.6.4 it is evident that, the sampling instant corresponds to the end of a given symbol interval; however, in the presence of noise this position becomes less obvious.

Fig.6.4 Noncoherent Correlator Outputs
For Input Tone Sequence 1, 4, 2, 3 and 2



In CABS the optimum decision instant is provided by a Viterbi decoder. A Viterbi decoder, by its very nature, is a maximum likelihood device, which bases its decisions on a number of received symbols. In this manner, the effects of noise are reduced and corruption of received symbols does not necessarily cause loss of synchronisation or data errors. The function of the Viterbi decoder in this context is therefore to provide both symbol timing information and error control.

In an idealised communication system, where the symbol synchronisation problem does not exist, the demodulator would provide the

Viterbi decoder with the output of the four correlators, once every symbol interval. Clearly, it must be established whether the sampled value of the correlator output at the synchronisation point is a measure of the likelihood of that symbol having been transmitted. In order to evaluate the pdf of the correlator output, two possible hypotheses must be defined. The first, Hyp0, corresponds to the condition when the frequency of the received signal is equal to the correlator reference signal. The second, Hyp1, on the other hand corresponds to the condition of unequal received signal and reference signal frequencies. For the sake of simplicity the pdf of the output of only one correlator is derived, since the result also holds for the remaining correlators, since the transmitted tones are all orthogonal.

The input to the correlators over one symbol interval can be represented as:

$$r(t) = \sqrt{(2E/T_s)}\sin(2\pi f_i t+\phi) + n(t) \qquad (1 \le i \le 4) \qquad (6.3)$$

where n(t) is the AWGN component with zero mean and standard deviation of σ_n, f_i is the transmitted frequency, ϕ is an arbitrary constant phase error (although in practice ϕ is random) and E is the transmitted signal energy and T_s is the symbol interval.
The inphase and the quadrature components of r(t), X_j and Y_j, are obtained by the following relationships:

$$X_j = \int_0^{T_s} [\sqrt{(2E/T_s)}\sin(2\pi f_i t+\phi) + n(t)].(\sqrt{(2/T_s)}\sin(2\pi f_j t)\, dt \qquad (6.4)$$

where $(1 \le j \le 4)$

$$X_j = \begin{cases} \sqrt{E}.\cos\phi + n_I & \text{for } i = j \quad \text{(i.e. Hyp0 true)} \\ n_I & \text{for } i \neq j \quad \text{(i.e. Hyp1 true)} \end{cases} \qquad (6.5)$$

$$Y_j = \int_0^{T_s} [\sqrt{(2E/T_s)}\sin(2\pi f_i t+\phi) + n(t)].(\sqrt{(2/T_s)}\cos(2\pi f_j t)\, dt \qquad (6.6)$$

where $(1 \le j \le 4)$

$$Y_j = \begin{cases} \sqrt{E}.\sin\phi + n_Q & \text{for } i = j \quad \text{(i.e. Hyp0 true)} \\ n_Q & \text{for } i \neq j \quad \text{(i.e. Hyp1 true)} \end{cases} \qquad (6.7)$$

where n_I and n_Q are independent Gaussian noise components corresponding to the inphase and the quadrature, respectively, having zero mean and identical variances, σ^2, given by $N_0/2$ [Arthurs 1962]. N_0 is the one sided power spectral density of n(t) given by:

$$N_0 = \sigma_n^2/B \qquad (6.8)$$

where B is the system bandwidth (in Hz) and σ_n is the standard deviation of n(t).
Clearly, X_j and Y_j are independent random variables with the probability density functions (pdf) $F_{Xj}(x)$ and $F_{Yj}(y)$ respectively [Papoulis 1965 pp127]:

$$F_{Xj}(x) = \frac{1}{\sqrt{(2\pi)}\sigma}\, e^{\frac{-(x-\sqrt{E}\cos\phi)^2}{2\sigma^2}} \quad \text{(i.e. Hyp0 true)} \qquad (6.9)$$

$$F_{Yj}(y) = \frac{1}{\sqrt{(2\pi)}\sigma}\, e^{\frac{-(y-\sqrt{E}\sin\phi)^2}{2\sigma^2}} \quad \text{(i.e. Hyp0 true)} \qquad (6.10)$$

where for $i \neq j$ (ie Hyp1 true), the pdf of X_j and Y_j are obtained by setting the signal energy, E, to zero.

The output of the correlator, Z_j (where, $Z_j = X_j^2 + Y_j^2$), is again a random variable with pdf $F_{z_j}(z)$. $F_{z_j}(z)$ can be determined by considering the joint pdf of, X_j and Y_j, $F_{x,y}(x,y)$. Due to the independency of X_j and Y_j, $F_{x,y}(x,y)$ is the product of F_{x_j} and F_{y_j}, thus:

$$F_{x,y}(x,y) = \frac{1}{2\pi\sigma^2} e^{\frac{-(x-\sqrt{E}\cos\phi)^2-(y-\sqrt{E}\sin\phi)^2}{2\sigma^2}} \qquad (6.11)$$

Now, since Z_j is obtained by the transformation $Z_j = X_j^2 + Y_j^2$, $F_{z_j}(z)$ can be obtained as follows [Papoulis 1965 pp188]:

$$F_{z_j}(z).dz = \iint_{D_z} F_{x,y}(x,y)\ dx.dy \qquad (6.12)$$

for $z \geq 0$ and equals to zero otherwise; where the region D_z describes a ring in the x-y plane.

Let $x=\sqrt{z}.\cos\theta$ and $y=\sqrt{z}.\sin\theta$, thus, $dx.dy = 0.5d\theta.dz$ and

$$F_{z_j}(z).dz = \frac{1}{4\pi\sigma^2}\int_0^{2\pi} e^{\frac{-(\sqrt{z}\cos\theta-\sqrt{E}\cos\phi)^2-(\sqrt{z}\sin\theta-\sqrt{E}\sin\phi)^2}{2\sigma^2}} d\theta.dz \quad z\geq0 \qquad (6.13)$$

and

$$F_{z_j}(z).dz = 0 \qquad \text{for} \qquad z < 0 \qquad (6.14)$$

which yields:

$$F_{z_j}(z) = \frac{1}{2\sigma^2} e^{\frac{-(z+E)}{2\sigma^2}} . \frac{1}{2\pi}\int_0^{2\pi} e^{\frac{\sqrt{(zE)}.\cos(\theta-\phi)}{\sigma^2}} d\theta \quad \text{for } z\geq0 \qquad (6.15)$$

and

$$F_{z_j}(z) = 0 \qquad \text{for} \qquad z < 0 \qquad (6.16)$$

using the fact that the modified Bessel function of order zero,

$I_o(x)$, is given by [Whalen 1971 pp104]:

$$I_o(x) = \frac{1}{2\pi} \int_o^{2\pi} e^{x.\cos(\theta-\phi)} \, d\theta \tag{6.17}$$

yields the pdf of the output of the correlator when signal is present (ie Hyp0 true), as:

$$F_{z_j}(z) = \begin{cases} \frac{1}{2\sigma^2} e^{\frac{-(z+E)}{2\sigma^2}} . I_o\left[\frac{\sqrt{(zE)}}{\sigma^2}\right] & z \geq 0 \\ 0 & \text{otherwise} \end{cases} \tag{6.18}$$

for the case when the received signal frequency is not equal to the frequency of the correlating signal (ie $i \neq j$), the pdf of the correlator output can be obtained by setting the received signal energy, E, to zero. Fig.6.5a and 6.5b depict the $F_{z_j}(z)$ for the hypotheses, Hyp0 and Hyp1 respectively. Clearly, the means of the two pdfs are quite distinct (for derivation of these see [Zolghadr 1989c]), thus the actual value of the j^{th} correlator output is a good measure of likelihood of the j^{th} symbol having been transmitted.

Fig.6.5a pdf of the Correlator Output Based on Hypothesis Hyp0, Ts=1 E=1,σ=0.1 with Mean of Z = 1.02

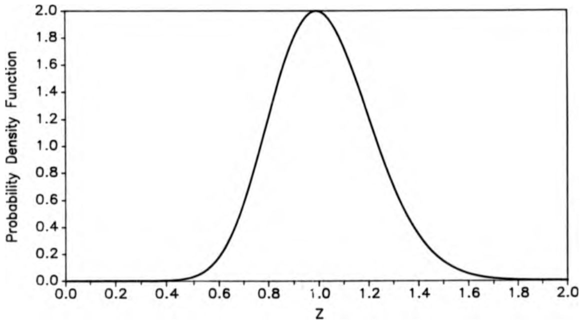
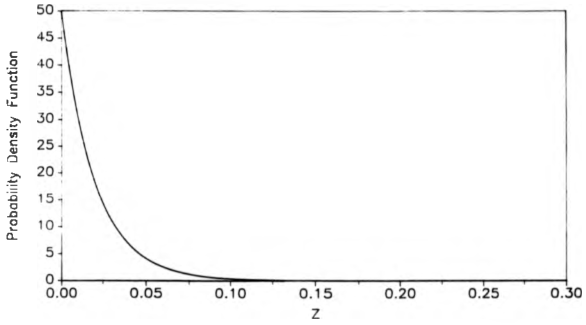


Fig.6.5b pdf of the Correlator
Output Based on Hypothesis Hyp1, Ts=1
E=1,σ=0.1 with Mean of Z = 0.02



The decoder would then assign the correlator output values (see expression 6.2) to the branches of the trellis diagram of Fig.6.2 using the following mapping rule:

correlator output	trellis branch
M_{1k}	00
M_{2k}	01
M_{3k}	10
M_{4k}	11

At any instant in time, L successive trellises would be stored in the decoder buffer, L being the search length of the decoder. In this way the decoder buffer would contain L trellises, corresponding to L successive received symbols, which would be then updated once every symbol interval. The function of the decoder would be to find the most likely path through the trellis. Once this path is found, the decoder would make a decision about the root of the path (ie the oldest entry in the trellis structure); this process would then be repeated every symbol interval.

In a practical communication system suffering from the effects of multipath propagation and also receiver/transmitter clock instabilities, the demodulator can only provide an estimate for each symbol interval, to the decoder. Consequently, in CABS, the correlators outputs are input to the decoder for every sample of the incoming signal. The system initially marks an arbitrary sample in time as the start of a symbol interval; the correlator outputs, corresponding to one symbol interval are then passed to the decoder, which updates a buffer containing L.N samples, where N is given by expression (6.1). The decoder then operates on L successive trellises separated in time by exactly N samples (ie one symbol interval), in a manner similar to that described above, as shown in Fig.6.6.

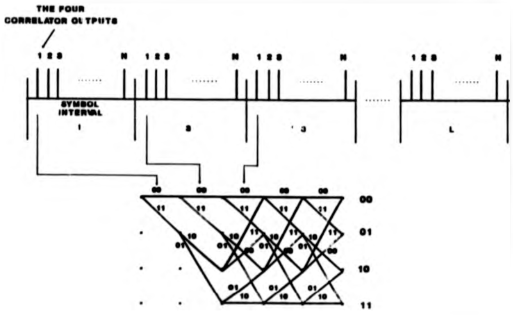


Fig.6.6 CABS System Decoding Procedure

Once the most likely path through the trellis is found, its value and the decoded symbol are stored for later comparison. In this method of decoding, the above procedure is repeated for every sample of the received signal (rather than for every symbol as in the case of idealised system). N such pairs of confidences and decoded symbols, corresponding to one complete symbol interval, are stored. The N stored pairs are then searched and the pair with the highest confidence is output as the demodulator/decoder decision for that particular symbol interval.

The time slot associated with the most likely path, amongst the N possible stored confidences, provides the demodulator with the desired symbol synchronisation information. The demodulator uses this information to correct for any symbol timing misalignments; this is achieved by forcing the duration of the proceeding symbol to vary by one sampling interval. In this way, the correlation window of the four correlators can be advanced or retarded in time, forcing the synchronisation point to lie at the centre of a given symbol interval.

6.4. Implementation of the CABS Modem

The CABS modem has been developed in two separate sections- the modulator and the demodulator. Each section consists of a TMS320C25 assembly program and a management utility program. The utility program resides on a host computer, the main function of which is to transfer data and program to and from the TMS320C25 board. The design and implementation of these sections is described below.

6.4.1 Modulator Design Philosophy

The principle of the modulator was discussed in the previous section: here, its design and implementation is described. The modulator consists of three main elements (see Appendix E):

(i) a high level utility program;

(ii) a 1/2 rate convolutional encoder;

(iii) a 4-tone MFSK modulator.

The utility program acts as a system management facility. The main task of this program are to transfer program and data to the DSP board and to control the operations of the TMS320C25; the program also acts as a direct interface between the user and the DSP board. The utility program initially down-loads the modulator assembly program to the DSP board. The data to be transmitted is stored in a file on the host computer hard disk unit; it is accessed by the utility program and is down-loaded to the DSP board's data-memory (total

128kbytes). The TMS320C25 is then set to run on the user's request. In addition, the utility program provides the facility for the manual fine tuning of the remote receiver. This is achieved by transmitting the channel symbol with the lowest tone frequency for approximately three minutes.

The encoder is implemented by utilising the bit manipulation instructions available on the TMS320C25. The raw data which is stored in the data-memory of the DSP board as bit patterns in a 16 bit word is loaded into the accumulator, a word at a time. The accumulator is masked off according to the tap positions; two masks are used, each providing two sets of binary bits. The bits in each set are XOR'd individually to form the encoder output as a pair of binary bits, which are then used to select one of the four possible baseband transmitted tones. The selected tone is then transmitted for precisely one symbol interval.

The choice of the transmitted tones frequencies is critical in MFSK systems because of the following considerations:

a) orthogonality requirement;

b) system bandwidth;

c) bandwidth power containment;

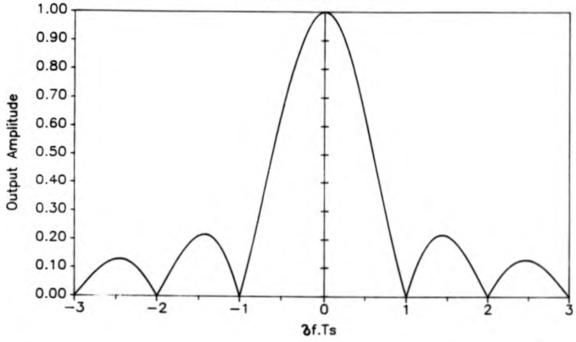
d) DSP limitations.

The orthogonality constraint can be best demonstrated by considering the frequency response M(f) of the noncoherent correlators output at the end of a symbol as illustrated in Fig.6.7:

$$M(f) = K.\mathrm{Sinc}(\pi.T_w.\delta f) \qquad (6.19)$$

where T_w is the correlation window duration (ie the symbol interval), δf is the difference between the correlator's input frequency and it's reference signal frequency (ie the matched frequency) and K is a constant.

Fig.6.7 Frequency Response of a Noncoherent Correlator



It is evident from Fig.6.7 that the maximum output is obtained when the frequency of the input signal to the correlator is equal to the frequency of the correlator reference signal. Furthermore, the output frequency response comprises nulls spaced at $1/T_s$ (Hz). It is evident from Fig.6.7 that the optimum frequency spacing for the four transmitted tones is $1/T_s$ (Hz). At this tone spacing, the output of the matched correlator would be at a maximum, whilst no residual effect would be observed in the non-matched correlators.

The frequency spacing of these tones is also dependent on the system bandwidth, which is primarily dictated by the bandwidth of the transmitter/receiver used. Fig.6.8 shows the frequency response of the combined transmitter/receiver (ICOM IC-735), obtained via a back-to-back test. Taking the 3dB points as the system bandwidth boundaries yields a lower and upper corner frequencies of 865 Hz and 2285 Hz; a total bandwidth of 1.42 kHz is therefore available which must be divided appropriately such that all the system requirements are satisfied.

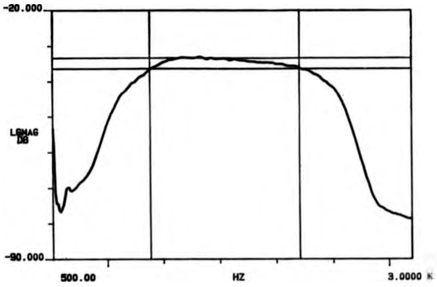


Fig.6.8 Transceiver Overall Frequency Response

The frequency spacing and, more importantly, the frequency guard spaces at the boundaries of the system bandwidth, dictate the power containment within that bandwidth. A figure of 99% is a typical value for this parameter. Previous work has shown that a frequency guard space of one tone spacing at each edge of the system passband is sufficient to meet this requirement [Shaw 1989 pp81-87]. Consequently, the total bandwidth of the system must be divided into five equal segments.

One further constraint controls the choice of the four frequencies: this is concerned with the limitations of the DSP board used. In order to obtain phase continuity in the transmitted signal, the periods of the generated tones must be integer multiples of the period of the DSP sampling clock; furthermore, a given symbol must contain integer number of whole cycles at the symbol tone frequency.

Taking the above constraints into consideration, the following tone frequencies were chosen:

Binary pair	selected tone frequency (Hz)
00	1250
01	1500
10	1750
11	2000

This choice of frequencies corresponds to:

a) orthogonal tones spaced 250 Hz apart (ie T_s = 4ms);

b) frequency guardspaces of 385 Hz and 285 Hz;

c) a DSP sampling clock rate of 6.25 kHz.

In this arrangement, the transmitted symbols contain 5, 6, 7 and 8 cycles for tones 1, 2, 3 and 4 respectively. Fig.6.9 shows the frequency spectrum of the modulator output with the four MFSK tones clearly visible. The bandwidth power containment requirement has been satisfied.

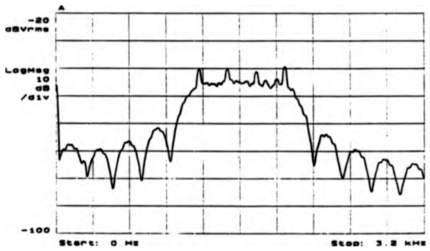


Fig.6.9 Modulator frequency Response

6.4.2 Demodulator Design and Implementation

The demodulator software (see Appendix F) comprises the following three main elements:

(i) a high level utility program;

(ii) four noncoherent detectors;

(iii) a 1/2 rate Viterbi decoder programmed in TMS320C25 assembly language.

The function of the utility program is very similar to that employed by the modulator; however, in this case, the decoded data is transferred from the DSP board to the Host computer. The block diagram of the elements (ii) and (iii) of above, is shown in Fig.6.10.

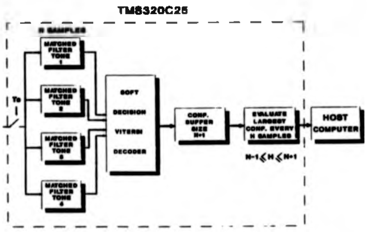


Fig.6.10 CABS Demodulator Block Diagram

The noncoherent detectors operate by evaluating the inphase and quadrature components of the incoming signal at every sampling instant, as shown in Fig.6.11. The eight products formed in this manner (four tones with two products per tone) are then saved in a "rotating buffer" for one symbol interval of 4 ms. In total, 200 values are stored in this buffer, ie 8N where N = 25.

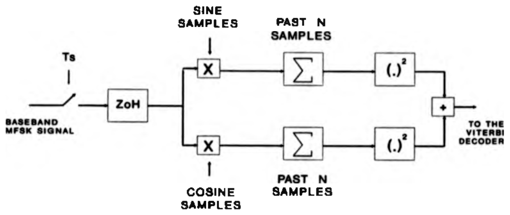


Fig.6.11 The Block Diagram of the Digital Noncoherent Correlator

For each correlator, the outputs of the multipliers are summed (integrated) over precisely the last 4ms, i.e. N=25 samples at 6.25 kHz. The summation process involves adding the new product to the integrator and subtracting the product evaluated exactly N samples ago. In this way, only four operations are required for each channel of the correlator, viz form a new product, add this to the sum, subtract the oldest product and save the new product in the buffer. The final output of the correlator is formed by summing the squares of the inphase and the quadrature components. Due to the complications involved in implementing a square-root function, the correlator outputs are actually the squared magnitude of the received vectors.

The outputs of the correlators for every sample are stored in a second rotating buffer, termed the "trellis buffer", in the trellis format of Fig.6.2 and in the manner described in Section 6.3; thus, for each symbol, N such trellises are stored. The trellises corresponding to L successive symbols, where L is the decoder search length = 16, are saved, ie 25.16.8 = 3200 entries. The two least significant bits of the 3200 trellis branch entries are set to zero; these two bits are later used by the Viterbi decoder.

The Viterbi algorithm is then applied to the stored trellises in the following manner (see also Fig.6.6):

1) Starting at time slot j, two 4-element arrays, VEC0 and VEC1 are initialised, VEC1 to all zeros and VEC0 to 0, 1, 2 and 3. Therefore, the two least significant bits (ie 00, 01, 10, 11) of each element in VEC0, point to one of the four possible states in the trellis diagram, as shown in Fig.6.12.

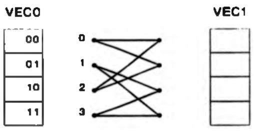


Fig.6.12 Diagrammatical Description of VEC0 and VEC1

VEC0 is used to store information relating to the last decoded trellis. Each element of VEC0, which consists of 16 bits, is split into two sets; the first set comprising the 14 most significant bits is used to store the accumulated metric of the surviving path; the second set, which contains the two least significant bits, is used to indicate the root state (starting point) of the surviving path. VEC1, on the other hand, is used to store the accumulated metrics of surviving paths in the trellis being decoded.

2) Starting at state i, the metrics of the two branches merging into this state are added to the contents of the appropriate elements of VEC0; this appropriate element is indicated by the origin states of

the two merging branches in the previous trellis. The two new accumulated metrics are compared and the biggest element is then stored in the ith element of VEC1 as the new accumulated metric. This process is repeated for the four states in the trellis, ie for i=0 --> 3.

3) The contents of VEC1 are then copied into VEC0 and VEC1 is initialised to zero.

4) The trellis buffer pointer, is advanced by N (ie 25) trellises so that the trellis corresponding to the same time slot in the next symbol is decoded next (see Fig.6.6).

5) Steps 2, 3 and 4 are repeated for the 16 trellises. The final four elements of VEC1 are then compared. The biggest element is stored in a third buffer as the decoded path information for time slot j, where the 14 most significant bits of this element indicate the decoded path confidence and the least significant two bits point to the starting state of the path.

6) Steps 1 to 5 are repeated for all j, $1 \leq j \leq H$, where H (initially equal to 25) is set by the synchronisation section of the demodulator. All the decoded path information for the H time slots, stored in the third buffer, is then compared and the biggest entry is chosen as the demodulator/decoder decision for that particular symbol.

7) Finally, the next symbol synchronisation time slot is adjusted. As indicated in Section 6.3, in CABS the symbol synchronisation point is set to fall in the middle of a given symbol. Consequently, the time slot associated with the chosen element is examined to ensure that it corresponds to the 13th time slot. Listed below are the possible test outcomes and the CABS responses:

	No. of samples in the next symbol (H)
sync. slot > 13	26
sync. slot < 13	24
sync. slot = 13	25

H is then adjusted according to the above rule. For continuous decoding, the above 7 steps are repeated.

A further point requiring careful consideration is the problem of intermediate result scaling (IRS). The TMS320C25 is an integer processor in which only provision for integer arithmetic operations is made. Although it is possible to perform floating point operations with the TMS320C25, the resulting degradation in speed is quite severe. When bounded by the limitations of the processor, IRS is essential since it affects both the dynamic range of the system and also the noise figure of the individual modules within the system.

The principle behind IRS is to operate with the largest possible operands, without allowing accumulator overflow. Scaling is then performed on the intermediate result such that the result of the next operation does not cause accumulator overflow (for a treatment of this topic see [Bogner 1975]. As a further safeguard against accumulator overflow errors, the TMS320C25 overflow mode is selected whereby, in the event of overflow, the accumulator is set to the largest possible positive or negative value.

6.5. Results of Simulation and On-Line Testing of the Modem

In this section the results of the simulation and practical tests of the modem are presented. The tests were basically grouped into four sections: firstly the performance of the CABS modem was investigated using simulation techniques under AWGN channel conditions; secondly, the modem was examined in order to verify its correct operation; thirdly, the performance of the modem over an AWGN channel was measured; finally, the modem was tested over a real HF channel.

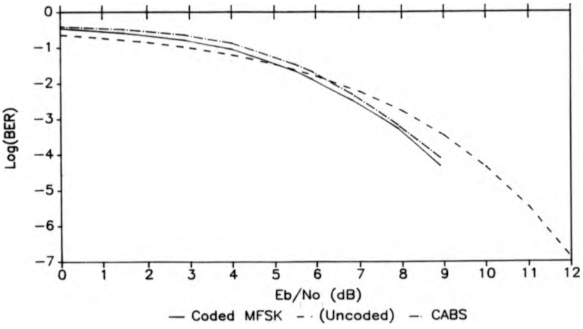
6.5.1 Simulation Results

The simulation tests were carried with the system parameters used in the practical system (see Section 6.4). Fig.6.13 shows the output bit error rate (BER) of the CABS system as a function of E_b/N_o, where E_b is the energy per information bit and N_o is the one-sided noise power spectral density. For the purpose of the simulation study E_b/N_o is given by:

$$E_b/N_o = 10\ \log_{10}(6.25/\sigma_n^2) \qquad (6.20)$$

where σ_n^2 is the standard deviation of the noise and the signal power is assumed to be 0.5W (ie unity amplitude sine wave). Also depicted in Fig.6.13 is the results of the simulation study of the trellis coded MFSK system (using code of Fig.6.2) and uncoded MFSK system. For these two results perfect symbol synchronisation is assumed.

Fig.6.13 Bit Error Rate of the CABS Modem, Noncoherent MFSK System (Uncoded) and Coded MFSK System vs Eb/No



6.5.2 Modem Operation Verification

Modem operation verification is an important stage in testing, under noise free conditions, the validity of the assembly programs. The operation of the modulator can be easily verified by examining its time and frequency domain outputs. Fig.6.9 shows the frequency spectrum of the modulator output; measurement of the frequencies of the individual tones gave the desired values.

The verification of the operation of the demodulator/decoder is more complex, since the correct demodulation/decoding process would not be sufficient to confirm its validity. For this reason, the verification process was split into two sections: firstly the operation of the correlators was examined; secondly, the functioning of the combined Viterbi decoder and the synchroniser was tested. Fig.6.14 shows the practical measurement of the correlator outputs for the repeated tone sequence 1, 4, 2, 3, 2 and 4. It is clear from the diagram that the four correlators are functioning correctly, with peaks occurring at regular intervals of 4ms, ie the symbol interval.

The function of the combined Viterbi decoder and the synchroniser was tested by storing the output of the decoder obtained at every sampling instant (see step 5 of Section 6.4.2) for consecutive symbols. The result is shown in Fig.6.15. It is clear that symbol synchronisation is obtained within 3-4 symbols; furthermore, the steady state condition is reached after only 64 ms, or 16 symbols.

Fig.6.14 Practical Measurement of the Four Noncoherent Correlator Outputs for the Tone Sequence of 1, 4, 2, 3, 2 and 4

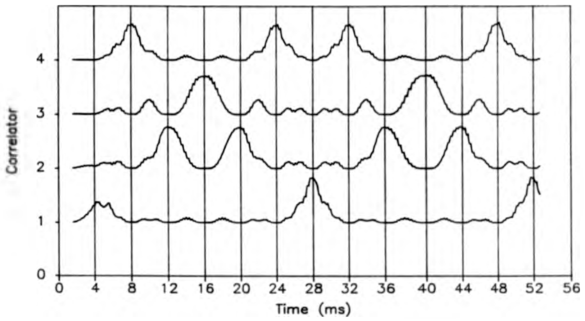
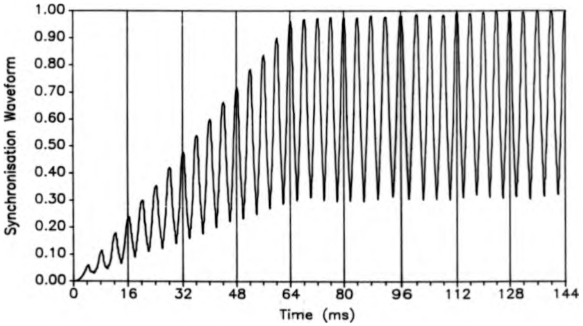


Fig.6.15 Practical Synchronisation Waveform of the CABS Demodulator/Decoder Observed in Real-Time



6.5.3 AWGN Channel Test Results

In order to evaluate the performance of the modem under AWGN channel conditions, the configuration shown in Fig.6.16 was used. A prerecorded, noise free, sample of the modulator output was added to the output of a Gaussian noise generator. The composite signal was then filtered using a low-pass filter with the frequency response of Fig.6.17. The corner frequency of the low-pass filter was set to 3kHz. The output of the filter was then connected to the demodulator.

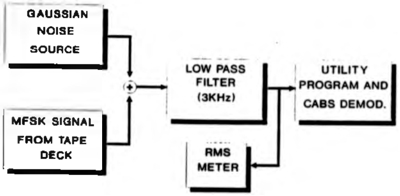


Fig.6.16 Block Diagram of the AWGN Test Configuration

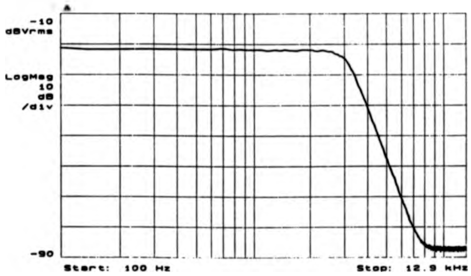


Fig.6.17 Low-Pass Filter Frequency Response

For a given value of signal-to-noise ratio (SNR), the demodulated data was processed in order to obtain the overall bit error rate (BER) of the modem. The test procedure listed below was followed:

1) the MFSK signal was disconnected from the summing junction;

2) with the noise signal connected, the output of the filter was measured on a true rms meter and the level of the noise adjusted to the required value;

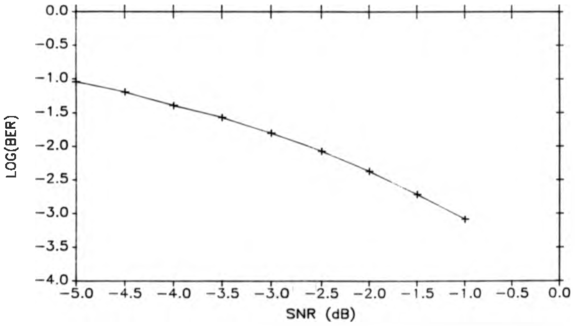
3) the noise signal was then disconnected from the summing junction;

4) with the MFSK signal connected, the output of the filter was measured on the rms meter and its level adjusted to the required value;

5) the noise and the MFSK signals were then re-connected to the summing junction and the demodulation process carried out.

The result of the above test is shown in Fig.6.18. No measurements were taken at high SNR values since for an accurate measure of the system BER, very long tests would have been required.

Fig.6.18 Practical Measurement of the CABS Modem BER for an AWGN Channel Condition



6.5.4 HF Trials Test Results

The modem has also been extensively tested over real HF path established between the Universities of Hull and Warwick. As an example the results of two such tests are outlined. The test parameters were as follows:

Test 1:

date	:	13/2/89
time	:	1400-
RF freq.	:	5.75 MHz USB
Tx power	:	100 W

Test 2:

date	:	22/3/89
time	:	1500-
RF freq.	:	5.75 MHz LSB
Tx power	:	100 W

The results of the two tests are presented in the form of consecutive errors and cumulative error-free run distributions, as shown in Figs. 6.19-6.22.

Test 1 corresponds to a "good" channel condition with a low overall BER of 1.1×10^{-3}. From Fig.6.19 it is evident that error-free runs of up to 12500 bits occurred; this represented almost one minute of error-free communication.

Test 2, on the other hand, could be classed as a "bad" channel with an overall BER of 1.07×10^{-2}. This is reflected directly in the error-free run distribution shown in Fig.6.21. Here, the maximum error-free run observed was only 5500 bits (ie approx. 1/3 minute).

Fig.6.19 Cumulative Error–Free Run Distribution For Test1. Overall Output BER=0.0011, Sample Size=51000.

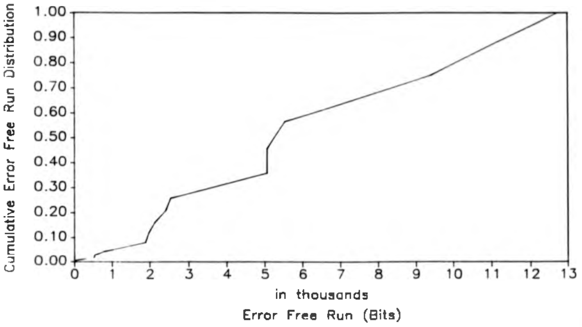


Fig.6.20 Consecutive Burst Distribution For Test1. Overall BER=0.0011, Sample Size=51000 bits

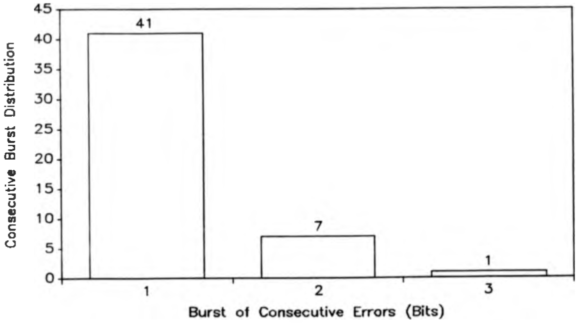


Fig.6.21 Cumulative Error–Free Run Distribution For Test2. Overall Output BER=0.0107, Sample Size=52000 Bits

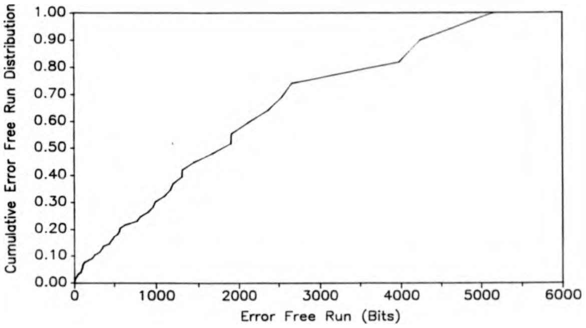
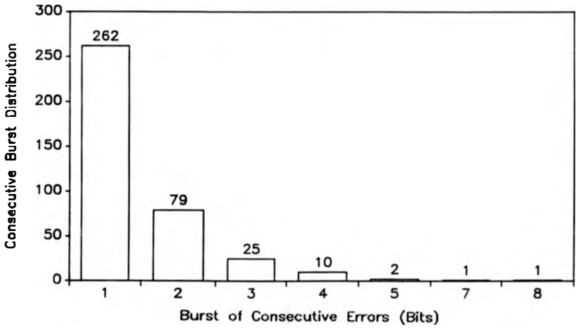


Fig.6.22 Consecutive Burst Distribution For Test2. Overall BER=0.0107, Sample Size=52000 Bits



Although the overall BER for these tests is relatively low, it is possible to apply further error control coding to remove the remaining errors. Figs.20 and 22 could be used to evaluate the power of the coding scheme required. It is clear that in these tests, consecutive errors not exceeding 8 bits were present.

6.6. Discussions

The principles, the practical implementation and the performance of the CABS modem, were described. It is evident from the AWGN channel performance curve that the modem is highly resilient to additive noise. The on-line HF channel tests also suggest that the modem also operates satisfactorily under non-Gaussian noise conditions.

The performance of the modem could be improved however by increasing its dynamic range and by using optimum parameters e.g.: longer symbol length. Currently, the modem exhibits a dynamic range of approximately 21dB, due to DSP limitations caused by its inability to perform floating point operations. Under HF channel conditions, with severe fading, this restricted dynamic range limits system performance. It would therefore be desirable to provide a further signal conditioning unit in the form of say a fast-acting AGC.

The demodulation/decoding procedure described above is highly flexible and offers a number of advantages over a conventional modem design approach. Of particular importance are the elimination of any initial synchronisation and the fact that variations in signal propagation time can easily be tracked. The work described represents a practical implementation of the multi-functional coding concept.

This page has been intentionally left blank.

CHAPTER 7

CONCLUSIONS & SUGGESTION FOR FURTHER WORK

7.1 Conclusions

The research reported in this thesis has been carried out in order to develop various techniques applicable to digital data transmission systems, operating over time variable channels. The techniques developed cover many aspects of a communication system, i.e. channel coding, modem design and RTCE. Throughout this work, emphasis has been placed on the amalgamation of the various sub-blocks of conventional communication systems, with the view to enhance the performance of the overall system. The conclusions we obtain from the study, including further research topics for the future, are classified in the following sections of this chapter.

7.1.1 Statistical RTCE

In Chapter 3, the principles of a statistical RTCE scheme, based on source coding techniques, were described. It was shown that, for poor channel conditions, reliable RTCE can be established without any additional overheads. The SRTCE algorithm has an advantage over conventional RTCE techniques in that it provides both channel error rate estimates and additional information about the statistical properties of the error generating mechanism. This information enables the on-line estimation of many useful channel characteristics e.g. error free run and consecutive error distributions, etc.

The theoretical analysis of the SRTCE algorithm has shown that accurate on-line estimates of the channel model can be obtained. Al-

Although the analysis in Section 3.2 only considers DMC and channel models with one bit dependency, the extension of this idea to more complicated channel models is possible.

7.1.2 Embedded Block Encoding Techniques

The concept of embedded block encoding was introduced and extensively investigated, in Chapter 4. The embedded encoding can be viewed as the amalgamation of the two basic error control strategies, i.e. error correction and error detection in conjunction with ARQ. This approach leads to the generalisation of conventional ARQ principles and enables the communication system to adapt to the prevailing channel conditions present at the time of the transmission of the block.

Both the theoretical and the simulation analysis of this scheme have been carried out. For the purpose of comparison the performances of an embedded array code and an equivalent array code have been investigated under AWGN channel conditions. The results indicate that the embedded encoding algorithm is highly suited to time variable channels. In fact at high channel BER, when the conventional ARQ systems cease to operate, the embedded encoding procedures continues to operate effectively.

7.1.3 Embedded Convolutional Encoding

The embedded block encoding concept was extended to convolutional codes in Chapter 5. The resulting code differs from embedded block encoding in that no outer code for error detection is utilised. Error detection is performed via an RTCE technique which is a by-product of the Viterbi algorithm.

An investigation of the behaviour of the above techniques under

AWGN conditions has been carried out using simulation techniques. The statistical analysis of the result obtained has shown that the developed RTCE technique yields a low rms error of estimation. The attractive feature of this technique is that it requires no additional overheads for the evaluation process. Moreover, its application is not limited to embedded encoding.

For the purpose of comparing the effectiveness of the embedded technique, additional results have been obtained for a convolutional coding scheme which utilises the same ARQ principles without the embedded structure. The comparison shows that the throughput is improved using the embedded algorithm. The technique described here paves the way for more widespread use of convolutional codes under an ARQ environment.

7.1.4 MFSK Modem Algorithms

Conventional communication systems design philosophy, views the design of the modem and the error control subsystems as two separate issues. This has lead to the division of these two fields. In the course of this research programme, it has consistently been shown that amalgamation of the various subsystems of a communication system can provide additional gains. This technique has been broadly referred to as multi-functional coding [Darnell Honary 1986].

The potential of multi-functional coding is further highlighted in Chapter 6; where the integration of the demodulation and the channel decoding processes has led to the development of a new reliable symbol synchronisation process. The modem developed, is termed the CABS modem.

The algorithm behind CABS can be viewed as a demonstration of the effectiveness of purely digital modem design techniques. In the context of CABS, the power of the channel coding procedure is used not only for error correction but also for symbol synchronisation. This results in a modem where, the symbol timing recovery is derived directly form the received signal, without any additional overheads.

The performance of the CABS algorithm has been extensively studied using simulation techniques, practical implementation and some theoretical procedures. The simulation results of the modem indicate that even in the absence of symbol synchronisation, the CABS modem performs only marginally (approximately 1dB) worse than the ideal receiver.

The practical implementation of this algorithm emphasises the attractive benefits gained by adopting a multi-functional coding approach. These benefits range from, a more efficient implementation to the low overall cost. Extensive testing of the modem over AWGN and the HF channels has shown that the modem is highly resilient towards both additive and the real channel noise effects.

7.2 Further Work

In this section some further research areas are outlined. The topics included are in some instances an extension of the work described in this thesis; whilst in some cases these stem from my own research interests.

7.2.1 Extensions of Multi-Functional Coding

The logical progression of the SRTCE technique described in Chapter 3 would seem to be its integration with channel coding schemes. An obvious choice of the channel coding scheme would be convolutional codes. These codes bear a good resemblance to some conventional source coding techniques [Honary Darnell Shaw 1989]. The combination of channel coding and source coding, under the constraints outlined in Chapter 3, would result in a more efficient system.

The benefits gained would be two fold. Firstly, the combination of source and channel decoders enables a complete soft-decision decoding to be performed. Secondly, the inclusion of the SRTCE principle allows on-line estimates of the channel status to be obtained.

7.2.2 Improvement of the Embedded Encoding

The merits of the embedded encoding procedure have been extensively outlined in this thesis. Clearly, the use of more powerful burst error correcting codes (including interleaving) is warranted. Here, it is proposed that Reed-Solomon (RS) codes [Reed Solomon 1960] should be employed. In this method, puncturing and shortening techniques of RS codes could be employed for the purpose of code rate variation. Error detection can be performed by utilising the extended decoding technique known as the Euclid's algorithm [Gallager 1968 pp216-217]. This algorithm enables the detection of uncorrectable errors, thus provides an ideal coding scheme for the embedded coding algorithm. Although the decoding procedure for these codes is more complex, their more powerful error protection capabilities would outweigh their complex decoder implementation.

7.2.3 Embedded Convolutional Codes Under CABS Environment

With the philosophy of multi-functional coding in mind, the integration of the embedded coding procedure and the demodulation process is a logical progression. This would in fact be performed under the framework of the CABS algorithm. The RTCE technique developed in Chapter 5 would then readily provide on-line estimates of the prevailing channel conditions. The result would be a totally adaptive modem in which the coding, channel evaluation and the symbol synchronisation are performed not only by one module but also in real time.

7.2.4 Improvements of the CABS Modem

Although the CABS modem is a highly robust modem, it suffers form two basic limitations. Firstly, the modem suffers form a low dynamic range which stems from the use of fixed point arithmetic in its practical implementation. This problem can be remedied by using an Automatic Gain Control (AGC) system. The function of this unit would be to maintain the signal level, at the input to the modem, at a constant predetermined value. Some work in this area has been carried out by the author [Zolghadr Honary 1989]. This has led to the development of the Matched-Correlator AGC (Mc-AGC) system. In Mc-AGC the gain control signal is derived directly from the noncoherent correlator outputs. In addition, the Mc-AGC provides reliable channel SNR estimates, by utilising the outputs of the non-matched correlators as noise estimates. The next stage of the work would be to combine the two schemes into one complete unit.

The second limitation of the current CABS modem is the short symbol period (i.e. 4 ms) employed. This choice was dictated by the practical limitations imposed on the original implementation of the algorithm. It is well established that long symbol duration, in the order of 40-90 ms, would provide much more improved immunity towards multipath propagation effects experienced on the HF channel.

Clearly, increasing the symbol duration would reduce the system baud rate, however, this would enable more compact tone spacing; thus more MFSK tones could be packed in the available bandwidth. Consequently, the number of information bits conveyed by each tone is increased. It is therefore necessary to investigate the effect of increasing the symbol duration on the performance of the CABS modem, with the view to maximise the information throughput of the system.

7.2.5 Extensions of the CABS Modem

At the heart of the CABS algorithm lies the fundamental concept of multi-functional coding. Although, in the implementation described a convolutional decoder (Viterbi algorithm) and a conventional non-coherent MFSK demodulator are used, this by no means limits the possibilities of using other coding or modulation formats.

Two main extensions of the current work can be identified. Firstly, the use of other modulation schemes (e.g. PSK, ASK etc.), must be extensively studied. The emphasis should be placed on the application of the CABS algorithm and the development of new digital signal processing techniques. In principle the operation of the CABS algorithm would be identical to that described in Chapter 6, however, the channel signal detection method would have to be modified.

Secondly, the use of more powerful burst error correcting codes, e.g. RS codes, could provide more immunity towards flat fading conditions experienced over the HF channel. The use of block codes however would not enable the synchronisation process to operate on a symbol by symbol basis. Here, the synchronisation would be established once for every block interval.

7.2.6 Semi-Orthogonal MFSK Modulation Format

The main draw back of conventional MFSK systems is the low data throughput rate (under limited transmission bandwidth conditions) resulting from the orthogonality constraint. Moreover, this low level of system throughput is constant regardless of the channel SNR.

Ideally, in an optimum system, the throughput of the system should increase with an increase in the channel SNR. This could in practice be achieved with either the removal of some of the channel coding redundancies or by increasing the actual transmission rate (i.e. reducing the symbol duration). The latter suggests that when the channel SNR is high the orthogonality constraint should be relaxed in order to achieve a higher transmission rate.

Relaxing the orthogonality constraint which is one of the most rigid limitations of the MFSK modulation techniques, would yield a Semi-Orthogonal MFSK modulation format. In this scheme, the system objective would be to maintain a low output BER rather than to minimise it. Clearly, this would involve the utilisation of a very accurate RTCE techniques. In fact the technique employed in the Mc-AGC algorithm would be an ideal candidate for this task. The system would operate by varying the symbol duration in response to the evaluated channel SNR. This results in the adaptation of the data transmission rate in response to the instantaneous channel capacity.

This page has been intentionally left blank.

REFERENCES

Abramson, N. , "Information Theory and Coding", McGraw-Hill, 1963.

Arthurs, E., Dym, H., "On the Optimum Detection of Digital Signals in the Presences of White Gaussian Noise- a Geometric Interpretation and a Study of Three Basic Data Transmission Systems", IRE Trans. on Com. Systems, Vol. CS-10, pp336-372, Dec. 1962.

Balser, M., Silverman, R. A., "Coding for Constant-Data-Rate Systems, Part I: A New Error Correction Code" Proc. IRE, Vol. 42, No. 9, pp1428, Sep. 1954.

Balser, M., Silverman, R. A., "Coding for Constant-Data-Rate Systems, Part II: Multiple-Error-Correction Code", Proc. IRE, Vol. 43, No. 6, pp728, Jun. 1955.

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

Betts, J. A., Ellington, R. P., Jones D. R. L., "C.W. Sounding and its Use for Control of H.F. (3-30MHz) Adaptive System for Data Transmission", Proc. IEEE, Vol. 117, No. 12, pp2209-2215, Dec. 1970.

Blahut, R. E., "Principles and Practices of Information Theory", Addison Wesley Publishing Co., 1987.

Bloom, F. J., "Improvement of Binary Transmission by Null-Zone Reception", Proc. IRE, Vol. 45, pp963, 1957.

Bogner, R. E., "Introduction to Digital Filtering", Wiley-Interscience, 1975.

Cain, J. B., Clark, JR., G. C., Geist, J. M., "Punctured Convolutional Codes of Rate (n-1)/n and Simplified Maximum Likelihood Decoding", IEEE Trans. on Info. Theory, Vol. IT-25, No.1, pp97-100, Jan. 1979.

Cahn, C. R., "Binary Decoding of Non-Binary Demodulation of Phase Shift Keying", IEE Trans., Vol. COM-17, No. 5, Oct. 1969.

Chase, D., "A Class of Algorithms for Decoding Block Codes with Channel Measure Information", IEEE Trans. Info. Theory, Vol. IT-18, No. 1, pp170, Jan. 1972.

Clark, A. P., "Advanced Data-Transmission Systems", Pentech Press, London, 1977.

Clark, A. P., "Principles of Digital Data Transmission", 2nd Edition, Pentech Press, 1983.

Daniel, J. S., Farrell, P. G., "Burst Error Correcting Array Codes: Further Developments", Proc. of the 4th International Conf. on Digital Processing of Signals in Com., Loughborough, Uk, April 1985.

Darnell, M., "Channel Evaluation Techniques for Dispersive Communications Paths", in "Communications Systems and Random Process Theory", ed. J. K. Skwirzynski, Sijthoff, The Netherlands, pp425-460, 1978.

Darnell, M., "HF System Design Principles", AGARD lecture series No.127 on "Modern HF Communications", pp8, May/Jun. 1983a

Darnell, M., "Embedded Real-Time Channel Evaluation Techniques", AGARD lecture series No. 127, on Modern HF Communications, May-Jun. 1983b.

Darnell, M., Honary, B., "Multi-Functional Coding Schemes Applicable to Secure Communication", Proc. IEE Int. Conf. on 'Secure Communications', Conf. Proc., No.269, Oct. 1986.

Darnell, M., Honary, B., Zolghadr, F.,: "Embedded Coding Techniques: Principles and Theoretical Studies.", IEE Proc. -F, Com. Radar and Signal Processing, Feb. 1988

Darnell, M., Honary, B., Zolghadr, F., "Embedded Array Coding For HF Channels, Theoretical & Practical Studies", Proc. of IMA Conf. on "Cryptography and Coding", Ed. (Beker, H. J.), Oxford Univ. Press, pp135-152, 1989.

Edwards, J.R., " A Comparison of Modulation Schemes for Binary Data Transmission", The Radio and Electronic Engineer, 43, No. 10, pp562-568, Sept. 1973.

Elias, P., "Coding for Noisy Channels", in 1955 IRE Nat. Conv. Rec., Vol.3, pt. 4, pp37-46, 1955.

Ellington, R. P., "The Use of Sounding to Control Adaptive Systems for High Frequency Radio Data Transmission", University of Southampton Electronics Department, Publication No. 201, Jan. 1970.

Elliot, E. O., "Estimates of Error Rates for Codes on Burst-Noise Channels", Bell. Syst. Tech. J., vol. 42, pp1977-1997, Sept. 1963.

Farrell, P.G., Honary, B.K. and Bate, S.D.: "Adaptive Product Codes with Soft/Hard Decision Decoding," Proc. IMA conference on cryptography and coding, Cirencester, Dec. 1988.

Ferguson T. J. and Rabinowitz J. H., "Self-Synchronising Huffman codes", IEEE Trans. Info. Theory, Vol. IT-30, No.4, pp687-693, Jul. 84.

Forney, G. D., "Concatenated Codes", MIT Press, 1966

Forney, G. D., "The Viterbi Algorithm", Proc. IEEE, Vol. 61, No.3, pp268-278, Mar. 1973.

Franks, L. E., "Carrier and Bit Synchronisation in Data Communication- A Tutorial Review", IEEE Trans. on Com., Vol. COM28, No.8, Aug. 1980.

Gallager, R. G., "Information Theory and Reliable Communication", John Wiley and Sons, 1968.

Gilbert, E. N., "Capacity of a Burst-Noise Channel", Bell Syst. Tech. J., Vol. 39, pp1253-1266, Sept. 1960.

Goodman, D.J. and Sundburg, C.W.: "The Effect of Channel Coding on the Efficiency of Cellular Mobile Radio Systems", Proc of Seminar on "Digital Communications", Zurich, pp85-91, Mar. 1984.

Hague, J., Jowett, A.P., Darnell, M., "Adaptive Control and Channel Encoding in an Automatic HF Communication System", Fourth Inter. Conf. on HF Radio Systems and Techniques, pp17-23, Apr. 1988.

Held, G., "Data Compression - Techniques and Applications, Hardware and Software Considerations" , Second Edition, John Wiley and Sons, Chichester, 1987.

Honary, B. K., "Error Correction Techniques for Bursty Channels", Ph.D. Thesis, Department of Electronics, University of Kent at Canterbury, Nov. 1981.

Honary, B., Ali, F., Darnell, M., "Information Capacity of Additive White Gaussian Channel with Practical Constraints", IEE Proc. Part I, Submitted for Publication Apr. 1989.

Honary, B., Darnell, M., Shaw, M., "Combined Huffman & Convolutional Codes: Concatenated form Without Feedback", Submitted to Electronics Letter for Publication, Nov. 1989.

Honary, B. Zolghadr, F., Darnell, M., "A Code-Assisted Bit Synchronisation (CABS) Scheme", Electronics Letters, Vol. 25, No. 23, 19th Jan. 1989.

Honary, B., Zolghadr, F., Darnell, M., Maundrell, M., "Multi-Functional MFSK Coding over Poor Narrow-Band HF Channels", IEE Proc. Part I, Submitted for publication Jun. 1989.

Huffman, D. A., "A Method for Construction of Minimum-Redundancy Codes", Proc. IRE, Vol. 40, pp1098-1101, Sept. 1952.

Jewett, W.M. and Cole, R.: "Modulation & Coding Study for the Advanced Narrowband Digital Voice Terminal", Naval Research Lab. (Washington DC) Report 3811, 1978.

Kanal, L.N. and Sastry, A.R.K., "Models for Channels with Memory and Their Application to Error Control", Proc. IEEE, Vol.66, No.7, pp724-744, Jul. 1978.

Kemeny, J.G. and Snell, J.L., "Finite Markov chains", reprint of the 1960 ed., Published by Van Nostrand, Princeton, N.J., 1976.

Larsen, K. J.: "Short Convolutional Codes with Maximal Free Distance for Rate 1/2, 1/3 and 1/4", IEEE Trans. Inform. Theory, Vol. IT-19, pp371-372, May 1973.

Lin, S. and Costello, D.J.: "Error Control Coding: Fundamentals and Applications", New Jersey: Prentice-Hall, pp458-497, 1983.

Lin, S., Costello, D.J., Jr and Miller, M.J.: "Automatic-Repeat Request Error-Control Schemes,"IEEE Communications Magazine, Vol. 22 No. 12, pp5-17, Dec. 1984.

Lindsey, W. C., Didday, R. L., "Subcarrier Tracking Methods and Communication System Design", IEEE Trans. Com. Tech., Vol. COM-16, pp541-550, 1968.

MacWilliams, F. J., Sloane, N. J. A., "The Theory of Error Correcting Codes", North Holland Publishing Company, 1977.

Massey, D. L., "Probability and Statistics", McGraw-Hill, 1971.

Massey, J. L., "Threshold Decoding", Cambridge, Mass., USA, 1963

Michelson, A M., Levesque, A. H., "Error-Control Techniques for Digital Communication", John Wiley & Sons, 1985.

Papoulis, A., "Probability, Random Variables, and Stochastic Processes", McGraw-Hill, 1965.

Psillides, C., Issaias, R., Farrell, P. G., Gott, G. F., "Soft Decision Decoding for the HF Channel", IEE Conf. Publ. 4245, (HF Com. Sys. and Techniques), pp171-175, Feb. 1985.

Mueller, K. H., Müller, M., "Timing Recovery in Digital Synchronous Data Receivers", IEEE Trans. on Com., Vol. COM-24, No. 5, May 1976.

Perl, J. M., Shpigel, A., Reichman, A., "Adaptive Receiver for Digital Communication Over HF Channels", IEEE Jour. on Selected Areas in Com., Vol. SAC-5, No. 2, pp304-308, Feb. 1987.

Peterson, W. W., Weldon, E. J., "Error-Correcting Codes", The MIT Press, Cambridge, Mass. 1972.

Proakis, J. G., "Digital Communications", International student edition, McGraw-Hill, 1983.

Ralphs, J. D., "The Application of MFSK Techniques to HF Telegraphy", Radio and Elect. Engineer, Vol. 47, No. 10, pp435-444, Oct. 1977.

Ralphs J. D., "Principles and Practices of Multi-Frequency Telegraphy", IEE telecom. series: II, 1985.

Reed, I. S., Solomon, G., "Polynomial Codes Over Certain Finite Fields", SIAM-8, pp300-304, 1960.

Reiffen, B., "Sequential Decoding for Discrete Input Memoryless Channels", IRE Trans. Info. Theory, Vol. IT-8, No. 2, pp208-220, Apr. 1962.

Schwartz, M., "Information Transmission, Modulation and Noise", Third Edition, 1982.

Seymour, S., "Fading Channel Issues in System Engineering", IEEE Jour. on Selected Areas in Com., Vol. SAC-5, No. 2, pp68-89, Feb. 1987.

Shannon, C. E., "A Mathematical Theory of Communication" Bell Syst. Tech. J., Vol. 27, pp379-423, 1948.

Shaw, M., "Modem Design Employing Real-Time Channel Evaluation", University of Warwick Ph. D. Thesis, Jul. 1989.

Sklar, B., "Digital Communications Fundamentals and Applications", Prentice-Hall, 1988.

Stein, S., "Fading Channel Issue in System Engineering", IEEE Journal on Selected Areas in Com., Vol. SAC-5, No.2, Feb. 1987

Stremler, F. G., "Introduction to Communication Systems", Addison-Wesley Publishing Co., 1982.

Stroud, K. A., "Engineering Mathematics", The MacMillan Press, 2nd Ed., 1982.

Thiede, E. C., "Decision Hysteresis Reduces Digital Pe", IEEE Trans. Com., Vol. COM-20, No. 5, pp1038, Oct. 1972.

Titchener M. R., "Construction and Properties of the Augmented and Binary Depletion Codes." IEE Proc.-E, Vol. 132, pp163-169, May 1985.

Titchener M. R., "Synchronisation Process for the Variable-Length T-codes", IEE Proc.-E, Vol. 133, No.1, pp54-64, Jan. 1986.

Titchener M. R., "Synchronisation and Encryption Over Noisy Channels", IEE Colloquium on "LOW BIT-RATE SPEECH COMMUNICATION VIA RADIO" Savoy Place London, 29 Apr. 1987.

Usubuchi, T., Omachi, T. and Iinuma, K., "Adaptive Predictive Coding for Newspaper Facsimile", Proc. IEEE, Vol. 68, pp807-812, Jul. 1980.

Viterbi, A. J., "Error Bounds for Convolutional Codes and an Asymptotically Optimum Decoding Algorithm", IEEE Trans. If. Theory, Vol. IT13, pp260-269, Apr. 1967.

Viterbi, A. J., "Convolutional Codes and Their Performance in Communication Systems", IEEE Trans. Com. Vol. COM-19, pp751-772, Oct. 1971.

Whalen, A. D., "Detection of Signals in Noise", Academic Press, 1971.

Wozencraft, J. M., Jacobs, I. M., "Principles of Communication Engineering", Wiley & Sons Publishing Co., 1968.

Yamamoto, H., Itoh, K., "Viterbi Decoding Algorithm for Convolutional Codes with Repeat Request", IEEE Tran. on Info. Theory, Vol. IT-26, No. 5, pp540-547, Sept. 1980

Yasuda, Y., et al.,: "Development of Variable Rate Viterbi Decoder and its Performance Characteristics.", Sixth Int. Conf. on Satellite Com. Phoenix AZ., Sept. 1983.

Yasuda, Y., Kashiki, K., Hirata, Y., "High-Rate Punctured Convolutional Codes for Soft Decision Viterbi Decoding", IEEE Trans. on Com., Vol. COM-32, No.3, pp315-319, Mar. 1984.

Yu, P.S. and Lin, S.: "An efficient Selective-Repeat ARQ Scheme for Satellite Channels and its Throughput Analysis" IEEE Trans. Commun., COM-29, No.3., pp353-363, Mar. 1981.

Zolghadr, F., "Preliminary Investigation of Embedded Array Encoding", Internal Report, Coventry Polytechnic, Jan. 1987

Zolghadr, F., Honary, B., Darnell, M.,: "Embedded Convolutional Coding.", Proceedings of the IERE, Fifth Int. Conf. on Digital Processing of Signals in Com., Sept. 88.

Zolghadr, F., Honary, B., Darnell, M., "Statistical Real Time Channel Evaluation (SRTCE) Techniques Using Variable Length T-codes", IEE Proc., Com., Speech and Vision, Vol.136, Pt.I, No.4, pp259-266, Aug. 1989.

Zolghadr, F., Honary, B., "Digital Matched Correlator Automatic Gain Control (Mc-AGC) Scheme Applicable to MFSK Modems", Submitted to IEE Proc. part I, 1989.

Zolghadr, F., "Reliable, Interference Resistance, Data Transmission Techniques for Multi-User HF Communications", Progress Report No.2 to Royal Aerospace Establishment (Farnborough) Contract No. 2119/037, University of Warwick, Aug. 1988.

Zolghadr, F., "Reliable, Interference Resistance, Data Transmission Techniques for Multi-User HF Communications", Progress Report No.3 to Royal Aerospace Establishment (Farnborough) Contract No. 2119/037, University of Warwick, Jan. 1989a.

Zolghadr, F., "Reliable, Interference Resistance, Data Transmission Techniques for Multi-User HF Communications", Progress Report No.4 to Royal Aerospace Establishment (Farnborough) Contract No. 2119/037, University of Warwick, Apr. 1989b.

Zolghadr, F., "Reliable, Interference Resistance, Data Transmission Techniques for Multi-User HF Communications", Progress Report No.6 to Royal Aerospace Establishment (Farnborough) Contract No. 2119/037, University of Warwick, Sep. 1989c.

This page has been intentionally left blank.

Symbols and Abbreviations

ADC	Analogue to Digital Convertor
AGC	Automatic Gain Control
ARQ	Automatic Repeat Request
ASCII	American Standard Code for Information Interchange
ASK	Amplitude Shift Keying
AWGN	Additive White Gaussian Noise
B	Bandwidth
BER	Bit Error Rate
Bit	Binary Digit
BSC	Binary Symmetric Channel
C	Channel Capacity
CABS	Code Assisted Bit Synchronisation
d	Hamming Distance
dB	Decibel
DMC	Discrete Memoryless Channel
DSP	Digital Signal Processor
E	Received Signal Energy
E_b	Bit Energy
E_b/N_0	Bit Energy to Noise Energy Ratio
EC	Embedded Convolutional
Erfc	Complementary Error Function
E_s	Symbol Energy
e.g.	for Example
FEC	Forward Error Correction
F_i	Error Correctability of Code i
f_s	Sampling Frequency
FSK	Frequency Shift Keying

GF	Galois Field
H_{∞}	Average Conditional Entropy of a Source
HF	High Frequency (2-30MHz)
Hz	Hertz
H.D	Hard Decision
H(X)	Source Entropy
IRS	Intermediate Result Scaling
i.e.	That is
I(X)	Information Content of X
L	Viterbi Decoder Search Length
LSB	Least Significant Bit
M	Number of Tones in an MFSK System
Mc-AGC	Matched-Correlator Automatic Gain Control
MFSK	Multiple Frequency Shift Keying
MLD	Maximum Likelihood Decoder
ms	Millisecond
N	Noise Power
N_o	One Sided Noise Power Spectral Density
n_I	Inphase Component of the Noise Signal
n_Q	Quadrature Component of the Noise Signal
n(t)	Noise Signal
OOK	On-Off Keying
P_c	Probability of a Block Without any Errors
P_d	Probability of a Block with Detectable Errors
pdf	Probability Density Function
P_e	Probability of a Block with Undetectable Errors
PLL	Phase Locked Loop
PSK	Phase Shift Keying
P(e)	Probability of Error

p, q	State Transition Probabilities
Q	Size of the Alphabet Set
Q_n	Current output of a Finite State Machine
R	Code Rate
RF	Radio Frequency
rms	Root Mean Square
RS	Reed-Solomon
RTCE	Real Time Channel Evaluation
s	Second
S	Signal Power
SE	Standard Echelon
SNR	Signal To Noise Ratio
SRTCE	Statistical Real Time Channel Evaluation
S.D	Soft Decision
T	Sampling Clock Period
T_s	Symbol Duration
Tx	Transmitter
USB	Upper Side Band
v	Constraint Length of a Convolutional Code
VA	Viterbi Algorithm
W	Watts
Z_n	Current State of a Finite State Machine
ZoH	Zero Order Hold
θ	Arbitrary Phase Angle
σ	Standard Deviation
ϕ	Phase Angle
δf	Small Frequency Increment

This page has been intentionally left blank.

APPENDIX A

Error Probability of Partial Array Code.

Consider the case when the digits of the partial array code are distributed as shown in Fig.A.1: then the probability of $\leq$ D, or no errors, in this case is given by the probability of i or no errors in 'a' AND the probability of j or no errors in 'b', where $i+j=D$ for all possible values of i and j. However, if D is less than K or C (the number of digits in 'a' and 'b' respectively, where $K < C$), it is clear that only a maximum of D errors can occur in 'a' or 'b'; therefore, i and j, the possible number of errors in 'a' and 'b', must be limited to D. Clearly the probability of G or no digits in error with a BER of P is given by:

$$\sum_{i=0}^{G} \binom{m}{i} P^{i} (1-P)^{m-i} \qquad \text{(A.1)}$$

where G is the maximum number of errors.

Hence, the probability of D or fewer errors P(ER), occurring in 'a' or 'b' is given by:

$$P(ER)=\text{Probability of D or no errors - Probability of no errors} \qquad \text{(A.2)}$$

The probability of no errors is given by:

$$(1-P_1)^{K} (1-P_2)^{C} \qquad \text{(A.3)}$$

and therefore

$$P(ER) = \sum_{i=0}^{H} \binom{K}{i} P_1^{\,i} (1-P_1)^{K-i} \cdot \left[\sum_{j=0}^{D-i} \binom{C}{j} P_2^{\,j} (1-P_2)^{C-j} \right]$$

$$- (1-P_1)^{K} (1-P_2)^{C} \qquad \text{(A.4)}$$

where $H=K$ for $D \geq K$ (A.5)

and $H=D$ for $D < K$

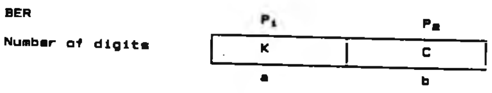


Fig.A.1 BER distribution of a (K+C,K) partial array code.

APPENDIX B

Error Probability of Embedded Array Code.

The probability tree of Fig.5.10 has four main features:

i) the probability tree comprises four levels, where sub-block ki is accepted at level i;

ii) a repeat request condition (ie branch leading off P_{ai} branch) extends to infinity;

iii) the P_{ci} branches of a given level converge to the same point on the next level;

iv) for the sake of clarity, the P_{ei} branches of any level are discontinued, since the error probability of these branches is P_{ei}.

It is apparent that the total probability error of the tree is given by

$$P(E)= P_{e1} + P_{a1}P_{e1} + P_{a1}^{2}P_{e1} + \ldots\ldots + P_{a1}^{n}P_{e1}$$
$$+ P_{c1}P(H) + P_{c1}P_{a1}P(H) + P_{c1}P_{a1}^{2}P(H) + \ldots\ldots + P_{c1}P_{a1}^{n}P(H)$$

As $n \dashrightarrow \infty$

$$P(E)= P_{e1}/(1-P_{a1}) + P_{c1}P(H)/(1-P_{a1}) \qquad \text{\_\_\_\_\_\_\_\_} \qquad (B.1)$$

where P(H) is the total probability of error of the remaining section of the tree, given by

$$P(H)= P_{e2} + P_{c2}P_{e1} + P_{c2}P_{c1}P_{e1} +.....+ P_{c2}P_{c1}^{n}P_{e1} + P_{e2}P(K)$$
$$+ P_{c2}P_{e1}P(J) + P_{c2}P_{e1}P_{c1}P(J) +.....+ P_{c2}P_{e1}P_{c1}^{n}P(J)$$

As n-->∞

$$P(H)= P_{e2} + P_{c2}P_{e1}/(1-P_{c1}) + P_{e2}P(K) + P_{e1}P_{c2}P(J)/(1-P_{c1}) \qquad (B.2)$$

where P(K) and P(J) are the total probabilities of error of the branches leading off P_{e2} and P_{e1} branches of level two respectively, given by

$$P(K)= P_{e2} + P_{c2}P_{e1} + P_{c2}P_{c1}P_{e1} +.....+ P_{c2}P_{c1}^{n}P_{e1}$$
$$+ P_{c2}P_{e1}P(L) + P_{c2}P_{e1}P_{c1}P(L) +.....+ P_{c2}P_{e1}P_{c1}^{n}$$

As n-->∞

$$P(K)= P_{e2} + P_{c2}P_{e1}/(1-P_{c1}) + P_{c2}P_{e1}P(L)/(1-P_{c1}) \qquad (B.3)$$

and

$$P(J)= P_{e2} + P_{c2}P_{e1} + P_{c2}P_{c1}P_{e1} +.....+ P_{c2}P_{c1}^{n}P_{e1}$$
$$+ P_{c2}P_{e2} + P_{c2}P_{c2}P_{e1} + P_{c2}P_{c2}P_{c1}P_{e1} +....+ P_{c2}P_{c2}P_{c1}^{n}P_{e1}$$
$$+ P_{c2}P_{e1}P(L) + P_{c2}P_{c1}P_{e1}P(L) +.....+ P_{c2}P_{c1}^{n}P_{e1}P(L)$$

As n-->∞

$$P(J)= P_{e2} + P_{c2}P_{e1}/(1-P_{c1}) + P_{c2}P_{e2} + P_{c2}P_{c2}P_{e1}/(1-P_{c1})$$
$$+ P_{c2}P_{e1}P(L)/(1-P_{c1}) \qquad (B.4)$$

where P(L) is the total probability of error of the branch leading off P_{a1} branch of level three, given by

$$P(L) = P_{a2} + P_{a2}P_{a1} + P_{a2}P_{a1}P_{a1} + \ldots\ldots + P_{a2}P_{a1}{}^{n}P_{a1}$$

As $n \rightarrow \infty$

$$P(L) = P_{a2} + P_{a2}P_{a1}/(1-P_{a1}) \qquad \text{(B.5)}$$

APPENDIX C

Performance Analysis of the Outer Decoder.

For the k1 sub-block, the following weight distribution can be obtained using parity checks P_1 and $P_{[illegible]}$ -$P_{[illegible]}$:

W(0)   =1

W(2+N3)   =3

W(1+3xN3)=2   (C.1)

W(1+4xN3)=1

W(3+3xN3)=1

where W(i) is the number of code vectors of weight i.

It is evident from the above weight distribution that the minimum Hamming distance of the partial array code of k1 is 2+N3 [MacWilliams and Sloane 1977]. It should be noted at this stage that the new bit transition probabilities only apply to the 3 information digits and not to the check digits of the outer code, since no correction has been performed on these digits; thus the bit error probability of the outer code parity digits still remains at P. Let $P_{[illegible]}$, $P_{[illegible]}$ and $P_{[illegible]}$ denote the probabilities of detectable errors, undetectable errors and error-free partial array code of k1 respectively; then

$$P_{[illegible]} + P_{[illegible]} + P_{[illegible]} = 1 \qquad (C.2)$$

It is now necessary to derive an expression for $P_{[illegible]}$. From

Fig.4.3 it can be seen that the number of parity checks of k1 is equal to (1+3xN3); it was also shown that the minimum Hamming distance of the partial array code is (2+N3). Therefore, (1+N3) error digits within the partial code can be detected; however, the bit transition probabilities, P and P' are distributed as shown in Fig.A.1.

It is shown in Appendix A that the probability of occurrence of (1+N3) or fewer errors in the above is given by:

$$P_{e1}=\sum_{i=0}^{H} \binom{3}{i} P'^{i} (1-P')^{3-i} \left[\sum_{j=0}^{N3+1-i} \binom{1+3xN3}{j} P^{j} (1-P)^{1+3xN3-j} \right]$$

$$- (1-P')^{3} (1-P)^{1+3xN3} \qquad (C.3)$$

where H=3 for (N3+1) ≥ 3

and H=(N3+1) for (N3+1) < 3 (C.4)

Similarly, the partial array code of k2 consists of outer code check digits P_{2} and $P_{[illegible]}$-$P_{[illegible]}$ with the following weight distribution

W(0) =1

W(2+N3) =3 (C.5)

W(2+2xN3)=3

W(4+3xN3)=1

Therefore, the minimum Hamming distance of the k2 partial array code is (2+N3) and from Fig.4.3, it can be seen that the number of parity checks of the partial code is (1+3xN3); again the partial code is split into two sections, as shown in Fig.A.1, with the exception that the transition probability of the information digits is now

P''. Using similar reasoning to that employed for the k1 sub-block, P_{e2} is given by:

$$P_{e2} = \sum_{i=0}^{H} \binom{3}{i} P''^{i} (1-P'')^{3-i} \left[\sum_{j=0}^{N3+1-i} \binom{1+3\times N3}{j} P^{j} (1-P)^{1+3\times N3-j} \right] - (1-P'')^{3} (1-P)^{1+3\times N3} \qquad (C.6)$$

where $H=3$ for $(N3+1) \geq 3$

and $H=(N3+1)$ for $(N3+1) < 3$ (C.7)

The k3 and k4 sub-blocks are, however, decoded simultaneously by the outer decoder. From Fig.4.3, it is evident that the number of parity checks of the partial array code is (6xN3+1) and the minimum distance of the partial array code [Daniel and Farrell 1985] is 4. The bit transition probabilities P''' and P are distributed as indicated in Fig.A.1. Therefore, P_{e34} is given by (see Appendix A):

$$P_{e34} = \sum_{i=0}^{3} \binom{6}{i} P'''^{i} (1-P''')^{6-i} \left[\sum_{j=0}^{3-i} \binom{1+6\times N3}{j} P^{j} (1-P)^{6\times N3+1-j} \right] - (1-P''')^{6} (1-P)^{6\times N3+1} \qquad (C.8)$$

Expressions for P_{ei} and P_{ci} will now be derived. The partial array code of ki is correct if all the digits in the partial code are correct, hence:

$$P_{c1} = (1-P')^{3} (1-P)^{1+3\times N3} \qquad (C.9)$$

$$P_{c2} = (1-P'')^{3} (1-P)^{1+3\times N3} \qquad (C.10)$$

$$P_{c34} = (1-P''')^{6} (1-P)^{6\times N3+1} \qquad (C.11)$$

and from equation (C.2)

$$P_{e1} = 1-P_{d1}-P_{c1} \tag{C.12}$$

$$P_{e2} = 1-P_{d2}-P_{c2} \tag{C.13}$$

$$P_{e3a4} = 1-P_{d3a4}-P_{c3a4} \tag{C.14}$$

APPENDIX D

Performance Analysis of (52,36) Array Code in an ARQ System.

Theoretical expressions for throughput efficiency and reliability of an array code of dimensions (n,k,4) are now derived. Consider the case when the array code is used in a selective repeat ARQ arrangement [Lin and Costello 1983, Lin and Costello 1984], in a Binary Symmetric Channel (BSC) with bit transition probability of P and a noiseless feedback channel: the following probabilities can be obtained:

(i) Pc= probability that a received block is error free;

(ii) Pe= probability that the received block contains undetectable errors;

(iii) Pd= probability that the received block contains detectable errors.

Where

$$Pc=(1-P)^{n} \tag{D.1}$$

$$Pd=\sum_{i=1}^{3}\binom{n}{i} P^{i}(1-P)^{n-i} \tag{D.2}$$

$$Pe=1-Pd-Pc \tag{D.3}$$

The lower bound on the throughput efficiency is defined as the ratio of the number of error-free blocks accepted by the receiver to the total number total number of blocks transmitted [Lin and Costello 1983]. For a block to be successfully accepted by the receiver, the average number of transmissions needed is [Lin and Costello 1984, Yu and Lin 1981]

$$M = 1Pc + 2(1-Pc)Pc + 3(1-Pc)^2\,Pc + \ldots + (m+1)(1-Pc)^m\,Pc + \ldots\ldots \qquad \text{(D.4)}$$

$$= 1/Pc \qquad \text{(D.5)}$$

$$\text{Efficiency}_a = R_a \times 1/M = R_a \times Pc \qquad \text{(D.6)}$$

where R_a is the code rate (k/n) and Efficiency_a is the overall throughput of system.

The reliability of an ARQ system [Yu and Lin 1981], P(e), is defined as the ratio of the number of digits in error at the output to the total number of digits at the output and is given by

$$P(e) = Pe/Pa \quad = 1 - Pc/Pa \qquad \text{(D.7)}$$

where Pa is the probability of acceptance given by

$$Pa = Pe + Pc \qquad \text{(D.8)}$$

Substituting from (D.1) and (D.8) in (D.7)

$$P(e) = 1-(1-P)^n/[1-\sum_{i=1}^{3} \binom{n}{i} P^i\,(1-P)^{n-i}] \qquad \text{(D.9)}$$

APPENDIX E

This appendix contains the modulators, complete TMS320C25 assembly program listings.

The TMS320C25 Modulator assembly program is as follows:

```
                    ************************************************
                    ***         CABS MODULATOR MK2.             ***
                    ***  THIS IS THE TMS320C25 PROGRAM FOR 4    ***
                    ***     TONE MFSK MODULATOR THE TONE        ***
                    ***  FREQUENCIES ARE AT 1250Hz, 1500Hz      ***
                    ***  1275Hz AND 2000Hz.                     ***
                    ***  THE SYMBOL DURATION IF 4 MILI-SEC, WITH ***
                    ***  BAUD RATE OF 250. CONVOLUTIONAL CODE RATE ***
                    ***  CODE USED IS OF RATE 1/2, THE INFORMATION ***
                    ***  BIT RATE IS THUS 250 BITS/SEC.         ***
                    ***                                         ***
                    ***  THIS PROGRAM MUST BE ASSEMBELED USING THE ***
                    ***  TEXAS INSTRUMENTS MACRO ASSEMBELER, XASM25, ***
                    ***          VER. PC 3.1                    ***
                    ***                                         ***
                    *** AUTHOR F. ZOLGHADR. 15/12/88            ***
                    ************************************************
*
*
*
**********************************************
*        START OF INITIALISATION SECTION    *
**********************************************
*
*
*
* ADDRESS CONSTANTS
*
*                                    PAGE 0 OF DTA MEM FOR MEM-REGS
IMR      EQU      >4                 ADDRESS OF INT MASK REG IN PAGE 0
TEMP     EQU      >63                TEMP IS A TEMP VARIABLE
SMBL     EQU      >64                SMBL HOLDS THE NEXT TRANSMITTED SYMBOL xx
SCNT     EQU      >65                SCNT IS THE SYMBOL COUNT FOR A WORD MAX=16
DATA     EQU      >66                DATA IS THE ADDRESS OF I/P DATA WORD IN
*                                    DATA MEMORY PAGE 1
STEP     EQU      >67                STEP IS THE OFFSET IN THE SIN TABLE
SIN      EQU      >68                SIN IS THE ACTUAL ADDRESS OF THE SIN TABLE
*                                    IN DATA PAGE 4
SINC     EQU      >69                SINC IS THE SIN SAMPLE COUNT FOR A SYMBOL
SR       EQU      >6A                SR HOLDS THE CONTENT OF LST1
TEMP1    EQU      >6B
TRANS    EQU      >6C                No. OF Tx. BYTES No.OF SYMBOLS = TRANS * 8
*
TIM      EQU      1                  PORT 1 IS THE TIMER ADDRESS
DAC      EQU      2                  PORT 2 ON WRITE IS DAC
*
* DATA CONSTANTS
*
```

```
TIMVAL    EQU      >FCE1         TIMER VALUE FOR 6.25KHz
IMASK     EQU      >FFC2         INT MASK TO ENABLE INT1 INTO AND RS
OFSET     EQU      5             OFSET IS THE MIN OFFSET IN SIN TABLE
PAGE0     EQU      0             PAGE 0 OF DTA MEM
SYMBOL    EQU      8             SYMBOL IS No. OF SYMBOLS IN A BYTE
SHIFT     EQU      2             SHIFT FOR SYMBOL = LOG2(TONES)
BASE      EQU      >C00          BASE FOR SIN TABLE PAGE 4
DBASE     EQU      >400          START OF DATA IN D_MEM
DEND      EQU      >FFFF         END OF         #
SAMPLE    EQU      25            SAMPL IS THE No. OF SAMPLES PER SYMBOLS
*
* SETTING RESET VECTOR
          AORG     0
          B        STRT
*
*
* SETTING THE ISR VECTOR
*
          AORG     >4            ADDRESS OF INT1 VECTOR
          B        ISR
*
*
*
=============================================
*         END OF INITIALISATION SECTION     *
=============================================
*
*
*
* START OF MAIN
*
STRT      AORG     >400          MAIN STARTS AT >400
          LDPK     PAGE0         PAGE 0 OF DATA MEM
          LRLK     AR1,IMASK
          SAR      AR1,IMR
          DINT
          LARP     AR0
*
*    INITIALISE THE VARIABLES
*
          LACK     SYMBOL        SETTING SCNT
          SACL     SCNT
          LALK     DBASE         BASE ADDRESS OF DATA TABLE
          SACL     DATA
          ZAC
          SACL     SIN           SETTING OFFSET IN SIN TABLE TO ZERO
          LACK     SAMPLE        SET No. OF SAMPLES
          SACL     SINC
          LALK     DEND          SETTING THE END OF DATA IN TRANS
          SACL     TRANS
*
*
* SET LST1 FOR RESET CONDITION
*
          LALK     >01F0
          SACL     SR
          LST1     SR            SET ST1 TO RESET CONDITION
          CNFD                   SO B0 BECOME DATA PAGE 4
```

```
* SET THE TIMER
*
        LALK    TIMVAL          TIMER VALUE IN ACC
        SACL    TEMP            PUT IN TEMP
        OUT     TEMP,TIM        SET TIMER TO GO
*
* GET DATA SYMBOL TO BE TRANSMITTED
*
        LARP    AR0
        LAR     AR0,DATA
        LAC     *,SHIFT         GET NEXT DATA IN THE ACC
        SACL    *               SAVE REMAINING DATA
        SACH    SMBL            SMBL HAS NEXT SYMBOL
*
*   DEC SYMBOL COUNT
*
        LAC     SCNT            ACC=SYMBOL COUNT FOR ONE WORD
        SUBK    1               DEC SYMBOL COUNT
        SACL    SCNT            SAVE SYMBOL COUNT
        LACK    OFSET           PUT MIN OFFSET IN ACC
        ADD     SMBL            ACC= TRUE OFFSET FOR CURRENT DATA SYMBOL
        SACL    STEP            SAVE STEP SIZE IN STEP
*
*   END OF MAIN INIT FROM HERE INT1 CAUSES THE CURRENT SIN VALUE
*   TO BE OUTPUT
*
*#################################################################
*
MAIN    LALK    BASE            BASE OF SIN IN P-MEM
        ADD     SIN             ACC=ACTUAL ADDRESS OF SIN
        TBLR    TEMP1
        EINT
        IDLE
        DINT
*
* SAMPLE O/P DEC SINC GET NEXT SIN AND CHECK IF FINISHED
* WITH CURRENT SYMBOL
*
        LAC     SINC            GET SAMPLE COUNT
        SUBK    1               DEC
        BZ      FINISH          IF ZERO END OF SYMBOL GET NEXT SYMBOL
        SACL    SINC            SAVE SAMPLE COUNT
*
        LAC     SIN             ACC=SIN TABLE POINTER
        ADD     STEP            ACC= NEXT SIN TABLE POINTER
*
*   TEST FOR WRAP
*
        SUBK    SAMPLE          No. OF SAMPLES IS 39 AFTER 39 WRAP
*                               IF < 0 WRAP NOT REACHED
        BLZ     NRAP            SIN IS VALID RESTORE
*
*   WRAP IS NEEDED THERFORE STORE CURRENT VALUE
*
        SACL    SIN
        B       MAIN            SIN NOW VALID GOTO MAIN FOR IDLE
*
*   WRAP NOT NEEDED RESTORE TO ORIGINAL VALUE AND SAVE
```

```
NRAP     ADDK      SAMPLE
         SACL      SIN
         B         MAIN            SIN NOW VALID GOTO MAIN FOR IDLE
*
* DONE WITH OLD SYMBOL GET NEW SYMBOL TO BE TRANSMITTED
*
*
*    DEC SYMBOL COUNT
*
FINISH   LAC       SCNT            ACC=SYMBOL COUNT FOR ONE WORD
         SUBK      1               DEC SYMBOL COUNT
         SACL      SCNT            SAVE SYMBOL COUNT
*                                  CHECK IF 8 SYMBOLS TRANSMITTED
         BGZ       NEXT            IF ACC > 0 MORE DATA LEFT IN WORD
*
*    8 SYMBOLS TRANSMITTED
*    INC AND CHECK DATA POINTER
*
         LACK      SYMBOL
         SACL      SCNT
*
         LAC       DATA            SET UP THE ADDRESS OF THE NEXT WORD TO Tx.
         ADDK      1
         SACL      DATA            SAVE DATA ADDRESSS
         SUB       TRANS           CHECK IF TRANS WORDS HAVE BEEN TRANSMITTED
         BLZ       NEXT            IF ACC < 0 TRANS NOT REACHED
         DINT                      DISABLE INT
LOOP     B         LOOP            GO INT IDLE LOOP
*
*    DATA ADDRESS OK
*
NEXT
         LARP      AR0
         LAR       AR0,DATA
         LAC       *,SHIFT
         SACL      *
         SACH      SMBL            SMBL HAS NEXT SYMBOL
*
         LACK      SAMPLE          ACC=No. OF SAMPLES
         SACL      SINC
*
*
*    DATA HAS BEEN SET UP IN SMBL AND ALL COUNTERS OK
*    GET APPROPRIATE POINTER TO THE SIN TABLE FOR THE SYMBOL
*
         LACK      OFSET           PUT MIN OFFSET IN ACC
         ADD       SMBL            ACC= TRUE OFFSET FOR CURRENT DATA SYMBOL
         SACL      STEP            SAVE STEP SIZE IN STEP
         ZAC
         SACL      SIN             SIN TABLE OFFSET =0 FOR NEXT SYMBOL
*
* NEW SYMBOL ATTRIB SET GOTO MAIN FOR IDLE LOOP
*
         B         MAIN
*
*
*
*######                  END OF MAIN PROGRAM          #######
```

```
*****************************************************************
*              ISR USED FOR OUTPUTING A SAMPLE OF CURRENT       *
*                              SYMBOL                           *
*****************************************************************
*
*
ISR      DINT
         SST       TEMP          STORE MAIN PROG STATUS IN TEMP
         OUT       TEMP1,DAC     O/P SIN VALUE TO DAC
         LST       TEMP          RESTORE OLD STATUS
         EINT
         RET
*
*#########################################################*
***********         END OF ISR DEFINITION       ************
*#########################################################*
*
*
*************************************************************
*       LOADER DIRECTIVE TO LOAD THE SIN TABLE INTO         *
*                     TMS PROGRAM MEMORY                    *
*************************************************************
*
*
*   INITIALISING THE SIN TABLE          *
*
*
*   SIN TABLE IN B0 OF P-MEM WHICH WILL CHANGE TO B0 OF
*   D-MEM PAGE 4, ON CNFD
*
         AORG      BASE          SIN STARTS AT 0 OF B0
         DATA      0
         DATA      8148
         DATA      15785
         DATA      22430
         DATA      27666
         DATA      31163
         DATA      32702
         DATA      32186
         DATA      29648
         DATA      25247
         DATA      19259
         DATA      12062
         DATA       4106
         DATA      -4106
         DATA     -12062
         DATA     -19259
         DATA     -25247
         DATA     -29648
         DATA     -32186
         DATA     -32702
         DATA     -31163
         DATA     -27666
         DATA     -22430
         DATA     -15785
         DATA      -8148
*
                   END
```

APPENDIX F

This appendix contains the CABS demodulators, complete TMS320C25 assembly program listings.

The TMS320C25 Modulator assembly program is as follows:

```
                    =====================================================
                    ***          CABS DEMODULATOR MK2.                ***
                    ***    THIS IS THE TMS320C25 PROGRAM FOR 4        ***
                    ***    TONE MFSK DEMODULATOR/VITERBI DECODER USING ***
                    ***    THE CABS SYSTEM FOR SYMBOL SYNCHRONISATION. ***
                    ***    THE DEMODULATOR IS BASED ON 4 NONCOHERENT  ***
                    ***    CORRELATORS AT FREQUENCIES 1250Hz, 1500Hz  ***
                    ***    1275Hz AND 2000Hz.                         ***
                    ***    THE SYMBOL DURATION IF 4 MILI-SEC. WITH    ***
                    ***    BAUD RATE OF 250. CONVOLUTIONAL CODE RATE  ***
                    ***    CODE USED IS OF RATE 1/2, THE INFORMATION  ***
                    ***    BIT RATE IS THUS 250 BITS/SEC.             ***
                    ***                                               ***
                    ***    THIS PROGRAM MUST BE ASSEMBELED USING THE  ***
                    ***    TEXAS INSTRUMENTS MACRO ASSEMBELER, XASM25, ***
                    ***               VER. PC 3.1                     ***
                    ***                                               ***
                    ***    AUTHOR F. ZOLGHADR. 27/1/89                ***
                    =====================================================
*
*
*
*
*############################################################*
*    THIS MACRO PERFORMS DECODING ON ONE STATE               *
*############################################################*
*
*              IT = STATE NUMBER
*              V0 = VEC0 AUX. REG
*              V1 = VEC1 AUX. REG
*
STATE   $MACRO   IT,V0,V1
*
        LAC      *BR0+,0,AR1              ACC=VEC0 :V0.S:=NEXT VEC0
        ADD      *+,0,:V1.S:              ACC=NEW CONF AR1=NEXT BRANCH
        SACL     *,0,:V0.S:               VEC1=NEW CONF
        LAC      *BR0+,0,AR1              ACC=VEC0 :V0.S:=NEXT VEC0
        ADD      *+,0,:V1.S:              ACC=NEW CONF AR1=NEXT BRANCH
        SUB      *                        NEW CONF-OLD CONF
        BLEZ     $+4
        ADD      *                        ACC = NEW
        SACL     *                        VEC1=NEW
        $IF      IT.V=4                   TIME TO SWAP VECTORS
        MAR      *BR0+,AR1                AR1 ACTIVE TO CHECK FOR WRAP AROUND
        $ELSE
        MAR      *BR0+,:V0.S:             :V0.S:=NEW VEC0
        $ENDIF
        $END
```

```
*##############################################################*
*    THIS MACRO INITIALISES VECO POINTED TO BY CURRECT        *
*                  ACTIVE AUX. REG.                            *
*##############################################################*
*
VECO       $MACRO
*                                          INITIALISING VECO BITS 0&1
           LACK      0                     THESE HOLD ROOT STATE INFO
           SACL      *+
           ADDK      1
           SACL      *+
           ADDK      1
           SACL      *+
           ADDK      1
           SACL      *BRO+                 BACK TO INITIAL ENTERY VALUE OF AR
           $END
*
*
*
*##############################################################*
*    THIS MACRO FINDS THE BIGEST ELEMENT OF A RECORD          *
*         POINTED TO BY THE CURRENT AUX. REG.                  *
*##############################################################*
*
BIG        $MACRO    POS,X
*
*    INPUTS ARE      POS = NAME OF THE RESULT VARIABLE
                     X   = FLAG FOR THE LAST COMPARISON
*
*
           LAC       *+                    READ VEC1
           SUB       *-                    SUB NEXT ELEMENT
           BLEZ      $+3
           DMOV      *                     NEW<OLD SWAP
           MAR       *+
           $IF       X.V=3
           LAC       *BRO+                 LAC WITH BIG & RESET VECO POINTER
           SACL      :POS.S:
           $ENDIF
           $END
*
*
*
*######################################################################*
*    THIS MACRO FINDS THE BIGGEST PATH CONFIDENCE FROM THE RECORD     *
*       KEPT AT >1080, POINTED TO BY AR6. AR6 MUST BE ACTIVE           *
*######################################################################*
*
BIGY       $MACRO
*
           LAC       *+                    ACC= CONFIDENCE VALUE
           SUB       CONF
           BLEZ      $+5
           ADD       CONF
           SACL      CONF
           DMOV      REGIST                REGIST OVERWRITES SLOT
           LAC       REGIST
           SUBK      1
```

```
        SACL     REGIST
        LAR      AR0,CONRAP                   AR0=RAP ADDRESS FOR AR6
        CMPR     2                            TEST FOR AR6 > AR0
        BBZ      $+3
        SBRK     25                           AR6 IS NOW CORRECTED
        $END
*
*
*
**********************************************************
*    THIS MACRO PERFORMS DECODING ON ONE TRELLIS    *
**********************************************************
*
*       INPUTS ARE:          TC=TRELLIS COUNT INDICATOR
*                            VE0=NAME OF THE AUX. REG. POINTING TO VEC0
*                            VE1=NAME OF THE AUX. REG. POINTING TO VEC1
*
TRELIS  $MACRO   TC,VE0,VE1
*
        $IF      TC.V=1                        FIRST TRELLIS STORE VALUE OF AUX. REG
        SAR      AR1,POINT
        $ENDIF
        STATE    1,:VE0.S:,:VE1.S:
        STATE    2,:VE0.S:,:VE1.S:
        STATE    3,:VE0.S:,:VE1.S:
        STATE    4,:VE0.S:,:VE1.S:

*       INC AR1 TO POINT TO THE NEXT TRELLIS AR1=AR1+312
        LAR      AR0,JUMP
        MAR      *0+
        LAR      AR0,WRAP
        CMPR     2                             TEST FOR AR1 > END OF BUFFER
        BBZ      $+4                           CHECK TC BIT
        LAR      AR0,OFFSET                    PASSED END OF BUFFER SUBTRACT OFFSET
        MAR      *0-,:VE1.S:                   WRAP AROUND AND SET AR TO POINT TO THE NEXT VEC0
        LARP     :VE1.S:                       SET ARP TO POINT TO NEXT VEC0
        LARK     AR0,2                         SET AR0 FOR NEXT TRELLIS DECODING

*       CHECK FOR ONE COMPLETE DECODING
*       IF COMPLETE FIND NEW STATE AND CONF.
*
        $IF      TC.V=16
*
*       CALL MACRO TO FIND BIGGEST ELEMENT      *
*
        BIG      NEW,1
        BIG      NEW,2
        BIG      NEW,3
        LAR      AR1,POINT                     RESET AR1 FOR NEXT SAMPLE AR1 NOW
*                                              POINTS TO THE OLDEST TRELLIS TO BE REPLACED
        LAC      NEW
        LARP     AR6
        SACL     *+                            SAVE SURVIVING PATH CONF. IN BUFFER
        LDPK     5                             PAGE 5 HOLDS THE CONSTANTS FOR AR6 CORRECTION
        LAR      AR0,CONRAP                    AR0=RAP ADDRESS FOR AR6
        CMPR     2                             TEST FOR AR6 > AR0
        BBZ      $+3
        SBRK     25                            AR6 IS NOW CORRECTED
        LDPK    0
        LAC     COUNTR
        SUBK    1
        SACL    COUNTR
        BGZ     JUMPY
        LDPK    5                       PAGE FIVE FOR ALL CONSTANTS
        LACK    25                      DEMODULATE ONE SYMBOL
        SACL    REGIST                  SETUP SYMBOL COUNTER
        SACL    SLOT                    SETUP TIME SLOT LOCATION
        ZAC
        SACL    CONF                    SET MAX CONF=0 READY FOR SEARCH
*   AT THIS POINT AR6 IS POINTING TO THE OLDEST ELEMENT IN CONF. BUFFER
        BIGY
        BIGY
        BIGY
        BIGY
        BIGY
        BIGY
        BIGY
        BIGY
        BIGY
        BIGY
        BIGY
        BIGY
        BIGY
        BIGY
        BIGY
        BIGY
        BIGY
        BIGY
        BIGY
        BIGY
        BIGY
        BIGY
        BIGY
        BIGY
        BIGY
*
*   AT THIS POINT AR6 IS POINTING TO THE OLDEST ELEMENT IN CONF. BUFFER
*   WE ARE ON PAGE 5
*
        LAC     CONF                    ACC=BIGGEST PATH METRIC
        LARP    AR7
        SACL    *+
        LRLK    AR0,OEND                CHECKING FOR ODD BUFFER END REACHED
        CMPR    0
        BBZ     OENDNO                  GOTO OENDNO IF NOT REACHED
        B       SIGNAL                  GOTO SIGNAL SINCE ODD BUFFER FULL
OENDNO  LRLK    AR0,EEND                CHECKING FOR EVEN BUFFER END REACHED
        CMPR    0
        BBZ     EENDNO                  GOTO EENDNO IF NOT REACHED
        LRLK    AR7,RESULT              RESET AR7 TO POINT TO START OF ODD BUFFER
SIGNAL  OUT     CONF,0
EENDNO  LARP    AR6
        LAC     SLOT
        SUB     OSLOT                   IF ACC>0 READ ONE LESS SAMPLE FOR NEXT SYMBOL
*                                       IF ACC<0 READ ONE MORE SAMPLE FOR NEXT SYMBOL
*                                       IF ACC=0 READ THE SAME NUMBER OF SAMPLES
```

```
         LDPK     0
         BZ       SAME
         BLZ      MORE
*                                      ACC < 0 DEC COUNTER
LESS     LACK     24
         SACL     COUNTR
         B        JUMPY
*                                      ACC = 0 LEAVE
SAME     LACK     25
         SACL     COUNTR
         B        JUMPY
*                                      ACC > 0 INC COUNTER
MORE     LACK     26
         SACL     COUNTR
*
*   INITIALISE VECO FOR NEXT DECODING PASS        *
*
JUMPY    LARP     AR2
         LARK     AR0,2                SET UP AR0 FOR VECO MACRO
         VECO
         $ENDIF
         $END
*
*
*##################################################################*
*        THIS MACRO STORES THE INTEGERTAED VALUES OF I & Q          *
*     COMPONENTS INTO THE TRELLISES. ON ENTERY AR1 IS ACTIVE        *
*    AND CONTAINS BASE ADDRESS OF THE TRELLIS. AR0 IS SET TO 3      *
*            ON EXIT AR2 IS ACTIVE AR0=OFFSET                       *
*##################################################################*
*
*
STORE    $MACRO
         LAC      ITONE0               ACC=SUM FOR FILTER 0
         SACL     *,2                  STORE IN TRELLIS BUFFER
         ADRK     5
         SACL     *+,2
         LAC      ITONE1               ACC=SUM FOR FILTER 1
         SACL     *0-,2                STORE IN TRELLIS BUFFER
         SACL     *+,2
*
         LAC      ITONE3               ACC=SUM FOR FILTER 3
         SACL     *0-,2                STORE IN TRELLIS BUFFER
         SACL     *+,2
*
         LAC      ITONE2               ACC=SUM FOR FILTER 2
         SACL     *,2
         ADRK     5
         SACL     *+,2                 AR1=ADDRESS OF NEXT TRELLIS (OLDEST ONE)
*
*        CHECK FOR WRAP AROUND THE BUFFER          *
         LAR      AR0,WRAP             AR0=WRAP ADDRESS
         CMPR     2                    TEST FOR AR1 > END OF BUFFER
         BBZ      $+4                  CHECK TC BIT
         LAR      AR0,OFFSET           PASSED END OF BUFFER SUBTRACT OFFSET
         MAR      *0-,AR2              WRAP AROUND
         LARP     AR2                  SET ARP TO POINT TO NEXT VECO
         $END
```

```
*#################################################################*
*     THIS MACRO TRANSFERS DATA FROM THE EXTERNAL P_MEM TO        *
*                REMAPED BLOCK 0 IN D_MEM                         *
*#################################################################*
*
SWAP     $MACRO
         MAR       *                        NOP
         MAR       *                        NOP
         MAR       *,AR4                    NOP
         LRLK      AR4,SIN                  DESTINATION B0 IN D_MEM
         RPTK      SAMPLE                   SAMPLE=SAMPLE-1
         BLKP      SBASE,*+                 TRANSFER SINE TABLE
         $END
*
*
*############################################################*
*                  END OF MACRO DEFINITIONS                  *
*############################################################*
*
*
*
*
*
*############################################################*
***********                MAIN PROGRAM           ***********
*############################################################*
*
*
*******************************************
*        START OF INITIALISATION SECTION    *
*******************************************
*
*
*
*ADDRESS CONSTANTS ARE
*
IMR      EQU      >4                        ADDRESS OF MASK REG. IN PAGE 0
ONE      EQU      >68                       CONSTANT ONE
NEW      EQU      >69                       NEW CONF+STATE INFO.
ITONE0   EQU      >6A                       SUM FOR FILTER 0
ITONE1   EQU      >6B                       SUM FOR FILTER 1
ITONE2   EQU      >6C                       SUM FOR FILTER 2
ITONE3   EQU      >6D                       SUM FOR FILTER 3
POINT    EQU      >6E                       STORAGE FOR OLDEST TRELLIS ADDRESS
WRAP     EQU      >6F                       STORAGE FOR WRAP ADDRESS
JUMP     EQU      >70                       INC FOR THE TRELLIS BUFFER
OFFSET   EQU      >71                       OFFSET FOR WRAPPING
SAMPLN   EQU      >72                       CURREST SAMPLED VALUE FROM ADC
ISIN0    EQU      >73                       SUM OF INPHASE FOR TONE 0
ICOS0    EQU      >74                       SUM OF QUADRATURE FOR TONE 0
ISIN1    EQU      >75                       SUM OF INPHASE FOR TONE 1
ICOS1    EQU      >76                       SUM OF QUADRATURE FOR TONE 1
ISIN2    EQU      >77                       SUM OF INPHASE FOR TONE 2
ICOS2    EQU      >78                       SUM OF QUADRATURE FOR TONE 2
ISIN3    EQU      >79                       SUM OF INPHASE FOR TONE 3
ICOS3    EQU      >7A                       SUM OF QUADRATURE FOR TONE 3
SBACK    EQU      >7B                       OFFSET FOR WRAPPING IN SIN TABLE
SWRAP    EQU      >7C                       ADDRESS FOR END OF SIN TABLE
```

```
PWRAP   EQU     >7D                     ADDRESS FOR THE END OF PRODUCT BUFFER
COUNTR  EQU     >7E                     SAMPLE COUNTER WITHIN A SYMBOL
TEMP    EQU     >7F                     TEMPORARY VARIABLE


*       CONSTANTS ON PAGE 5      *
*
CONRAP  EQU     >70                     WRAP ADDRESS FOR CONFIDENCE BUFFER
CONOFF  EQU     >71                     OFFSET FOR THE CONFIDENCE BUFFER
CONF    EQU     >72                     HOLDS MAX CONFIDENCE METRIC
REGIST  EQU     >73                     REGIST IS COUNTER FOR TIME SLOTS
SLOT    EQU     >74                     HOLDS THE DEMODULATED TIME SLOT NUMBER
ONE5    EQU     >75                     HOLD CONSTANT 1
OSLOT   EQU     >76                     HOLDS PREVIOUS VALUE OF TIME SLOT
*
*
*DATA CONSTANTS ARE
*
SBASE   EQU     >EF00                   ADDRESS OF SINE TABLE IN E_P_MEM
SAMPLE  EQU     >C8                     NUMBER OF ENTERIES IN SIN TABLE=25*8-1
SIN     EQU     >200                    ADDRESS OF SIN TABLE IN I_D_MEM B0
PRODU   EQU     >300                    ADDRESS OF THE PRODUCT BUFFER
TIM     EQU     >1                      TIM IS THE TIMER PORT
ADC     EQU     >2                      ADC IS PORT 2 ON READ
TIMVAL  EQU     >FCE1                   6.25KHZ SAMPLING RATE
IMASK   EQU     >FFC2                   INTRRUPT MASK TO ENABLE INT1 & RESET
OUTPUT  EQU     >2C8                    START ADDRESS OF THE OUTPUT VECTORS IN B0
RESULT  EQU     >1080                   START ADDRESS OF THE RESULT
OEND    EQU     >1580                   END OF ODD BUFFER ADDRESS
EEND    EQU     >1A80                   END OF EVEN BUFFER ADDRESS
*
*
*
*       SETING UP JUMP VECTORS IN PAGE 0
        AORG    >00
        B       START                   JUMP TO START OF PROG IN RESET
*
        AORG    >4                      ADDRESS OF INT1 VECTOR
        B       ISR
*
START   AORG    >400
*
**********************************************************
*       ZERO DATA MEMORY B2,B0,B1 AND E_D_MEM          *
**********************************************************
*
        DINT
        ZAC
        LARK    AR1,>60
        LARP    AR1
        RPTK    >1F                     ZERO BLOCK B2
        SACL    *+
        LRLK    AR0,>FFFE
        LRLK    AR1,>200                ZERO BLOCK B0 --> FFFF
LOOP    SACL    *+
        CMPR    2                       TEST FOR AR1 > FFFF
        BBZ     LOOP
*
```

```
        LDPK    0                       PAGE 0 ACTIVE FOR INDIRECT ADDRESSING
        SOVM                            SET OVERFOLW MODE
        SSXM                            SET SIGN EXTENSION MODE
        CNFD
        SWAP                            MACRO CALL TO TRANSFER SIN TABLE TO B0
*
        LRLK    AR0,IMASK               SETTING UP INTERRUPT MASK
        SAR     AR0,IMR
        DINT
        LRLK    AR1,>400                START OF TRELLIS BUFFER IN EXTERNAL D-MEM
        LARK    AR2,>60                 START OF VEC0 IN B2
        LARK    AR3,>64                 START OF VEC1 IN B2
        LRLK    AR4,PRODU               START OF PRODUCT TABLE IN AR4
        LRLK    AR5,SIN                 START OF SIN TABLE IN AR5
        LRLK    AR6,OUTPUT              START OF OUTPUT VECTORS IN E_D_MEM
        LRLK    AR7,RESULT              START OF RESULT IN D-MEM. PATH CONF, BIT, SLOT
        LACK    1
        SACL    ONE
        LALK    >107F                   WRAP ADDRESS OF TRELLIS BUFFERS=8*16*25
        SACL    WRAP
        LACK    >C0                     ACC=INC FOR THE TRELLIS BUFFER=25*7 TO START
        SACL    JUMP                    OF NEXT TRELLIS OF THE NEXT SYMBOL
        LALK    >C80                    ACC=OFFSET FOR WRAPPING
        SACL    OFFSET
        LALK    >C8                     ACC=OFFSET FOR WRAPPING FOR SIN TABLE
        SACL    SBACK
        LALK    >2C7                    ACC=ADDRESS FOR SIN TABLE WRAP ADDRESS
        SACL    SWRAP
        LALK    >3C7                    ACC=ADDRESS FOR PRODUCT BUFFER WRAP
        SACL    PWRAP
        LDPK    5                       PAGE FIVE IS END OF B0
        LALK    >2E0                    ACC=ADDRESS FOR CONFIDENCE BUFFER WRAP
        SACL    CONRAP
        LACK    25                      ACC=OFFSET FOR WRAP FOR CONFIDENCE BUFFER
        SACL    CONOFF
        LACK    1
        SACL    ONE5
        LACK    12                      ACC=MIDLE OF SYMBOL
        SACL    OSLOT                   OLD SYNC. SLOT=MIDLE OF SYMBOL TO START
        LDPK    0
        LACK    25                      ACC=NUMBER OF SAMPLES WITHIN A SYMBOL
        SACL    COUNTR
*
        LARP    AR2
        LARK    AR0,2                   2 BIT REVERSED ADDRESS MODE
*                                       INITIALISING VEC0 BITS 0&1
        VEC0                            THESE HOLD ROOT STATE INFO
*
*       SETTING THE TIMER       *
*
        LALK    TIMVAL                  TIMER VALUE IN ACC = FFFF-(5*PERIOD IN MICRO SEC)+2
        SACL    TEMP
        OUT     TEMP,TIM                SET UP TIMER
*
*
--------------------------------------------
*       END OF INITIALISATION SECTION      *
--------------------------------------------
```

```
        LARP    AR5
MAIN    EINT                            INITIALISATION FINISHED ENABLE INTERRUPT
        IDLE                            WAIT FOR AN INTERRUPT
*
        LARK    AR0,3                   REQUIREMENT OF STORE MACRO
        LARP    AR1
        STORE                           MACRO CALL TO INITIALISE TRELLIS
*
        LARK    AR0,2                   2 BIT REVERSED ADDRESS MODE
*
*       START WITH AR2 ACTIVE       *
*
        TRELIS  1,AR2,AR3
*
        TRELIS  2,AR3,AR2
*
        TRELIS  3,AR2,AR3
*
        TRELIS  4,AR3,AR2
*
        TRELIS  5,AR2,AR3
*
        TRELIS  6,AR3,AR2
*
        TRELIS  7,AR2,AR3
*
        TRELIS  8,AR3,AR2
*
        TRELIS  9,AR2,AR3
*
        TRELIS  10,AR3,AR2
*
        TRELIS  11,AR2,AR3
*
        TRELIS  12,AR3,AR2
*
        TRELIS  13,AR2,AR3
*
        TRELIS  14,AR3,AR2
*
        TRELIS  15,AR2,AR3
*
        TRELIS  16,AR3,AR2
*
        B       MAIN,*,AR5
*
*
*
*#############################################################*
************        END OF MAIN PROGRAM        ************
*#############################################################*
*
```

```
*#############################################################*
***********          ISR DEFINITIONS              ***********
*#############################################################*
*
*
*=====================================================================
*            ISR USED FOR SAMPLNG AND PERFORMING MATCH                *
*                        FILTER OPERTION                              *
*=====================================================================
*
*
*
*==========================================
*            MATCH FILTERS SECTION        *
*==========================================
ISR      DINT
         IN        SAMPLN,ADC              SAMPLE ADC PORT
         LAC       SAMPLN,15               SCALING SO THAT SAMPLE FITS 15 BITS
         SACH      SAMPLN                  NEW SAMPLN= +-2^14
         LT        SAMPLN                  T=LATEST SAMPLE
*
*
*============================  FILTER 0  ===========================================
*
         MPY       *+,AR4                  P=SAMPLE*SIN0, ACTIVATE PRODUCT POINTER
         ZALH      ISIN0                   ACCH=ISIN0
         SUBH      *                       ACCH=ISIN0-OLDEST0
         APAC                              ACCH=ISIN0-OLDEST0+NEWEST0
         SACH      ISIN0                   STORE NEW SIN COMPONENT OF FILTER 0 O/P
         SPH       *+,AR5                 (AR4)=NEWEST0/2^16
*
         MPY       *+,AR4                  P=SAMPLE*COS0, ACTIVATE PRODUCT POINTER
         ZALH      ICOS0                   ACCH=ICOS0
         SUBH      *                       ACCH=ICOS0-OLDEST0
         APAC                              ACCH=ICOS0-OLDEST0+NEWEST0
         SACH      ICOS0                   STORE NEW COS COMPONENT OF FILTER 0 O/P
         SPH       *+,AR5                  (AR4)=NEWEST0/2^16
*
*
*============================  FILTER 1  ===========================================
*
         MPY       *+,AR4                  P=SAMPLE*SIN1, ACTIVATE PRODUCT POINTER
         ZALH      ISIN1                   ACCH=ISIN1
         SUBH      *                       ACCH=ISIN1-OLDEST1
         APAC                              ACCH=ISIN1-OLDEST1+NEWEST1
         SACH      ISIN1                   STORE NEW SIN COMPONENT OF FILTER 1 O/P
         SPH       *+,AR5                  (AR4)=NEWEST1/2^16
*
         MPY       *+,AR4                  P=SAMPLE*COS1, ACTIVATE PRODUCT POINTER
         ZALH      ICOS1                   ACCH=ICOS1
         SUBH      *                       ACCH=ICOS1-OLDEST1
         APAC                              ACCH=ICOS1-OLDEST1+NEWEST1
         SACH      ICOS1                   STORE NEW COS COMPONENT OF FILTER 1 O/P
         SPH       *+,AR5                  (AR4)=NEWEST1/2^16
*
*
*
*
```

```
*******************************   FILTER 2   ******************************************
*
        MPY     *+,AR4              P=SAMPLE*SIN2, ACTIVATE PRODUCT POINTER
        ZALH    ISIN2               ACCH=ISIN2
        SUBH    *                   ACCH=ISIN2-OLDEST2
        APAC                        ACCH=ISIN2-OLDEST2+NEWEST2
        SACH    ISIN2               STORE NEW SIN COMPONENT OF FILTER 2 O/P
        SPH     *+,AR5              (AR4)=NEWEST2/2^16
*
        MPY     *+,AR4              P=SAMPLE*COS2, ACTIVATE PRODUCT POINTER
        ZALH    ICOS2               ACCH=ICOS2
        SUBH    *                   ACCH=ICOS2-OLDEST2
        APAC                        ACCH=ICOS2-OLDEST2+NEWEST2
        SACH    ICOS2               STORE NEW COS COMPONENT OF FILTER 2 O/P
        SPH     *+,AR5              (AR4)=NEWEST2/2^16
*
*
*******************************   FILTER 3   ******************************************
*
        MPY     *+,AR4              P=SAMPLE*SIN3, ACTIVATE PRODUCT POINTER
        ZALH    ISIN3               ACCH=ISIN3
        SUBH    *                   ACCH=ISIN3-OLDEST3
        APAC                        ACCH=ISIN3-OLDEST3+NEWEST3
        SACH    ISIN3               STORE NEW SIN COMPONENT OF FILTER 3 O/P
        SPH     *+,AR5              (AR4)=NEWEST3/2^16
*
        MPY     *+,AR4              P=SAMPLE*COS3, ACTIVATE PRODUCT POINTER
        ZALH    ICOS3               ACCH=ICOS3
        SUBH    *                   ACCH=ICOS3-OLDEST3
        APAC                        ACCH=ICOS3-OLDEST3+NEWEST3
        SACH    ICOS3               STORE NEW COS COMPONENT OF FILTER 3 O/P
        SPH     *+,AR5              (AR4)=NEWEST3/2^16
*
*
*       FINDING THE MAGNITUDE SQUARED VALUES OF THE FILTER O/PS         *
*
*******************************   FILTER 0   ******************************************
*
        SQRA    ISIN0               PREG=ISIN0*ISIN0
        ZAC
        SQRA    ICOS0               ACC=ISIN0*ISIN0
*                                   PREG=ICOS0*ICOS0
        APAC                        ACC=ISIN0*ISIN0+ICOS0*ICOS0
        SACH    ITONE0              STORE MAG. SQUARED IN I_D_MEM
        LAC     ITONE0,11           ACCH=ITONE0/2^5
        SACH    ITONE0
*
*******************************   FILTER 1   ******************************************
*
        SQRA    ISIN1               PREG=ISIN1*ISIN1
        ZAC
        SQRA    ICOS1               ACC=ISIN1*ISIN1
*                                   PREG=ICOS1*ICOS1
        APAC                        ACC=ISIN1*ISIN1+ICOS1*ICOS1
        SACH    ITONE1              STORE MAG. SQUARED IN I_D_MEM
        LAC     ITONE1,11           ACCH=ITONE1/2^5
        SACH    ITONE1
*
```

```
******************************   FILTER 2   ******************************************
*
         SQRA      ISIN2                 PREG=ISIN2*ISIN2
         ZAC
         SQRA      ICOS2                 ACC=ISIN2*ISIN2
*                                        PREG=ICOS2*ICOS2
         APAC                            ACC=ISIN2*ISIN2+ICOS2*ICOS2
         SACH      ITONE2                STORE MAG. SQUARED IN I_O_MEM
         LAC       ITONE2,11             ACCH=ITONE2/2^5
         SACH      ITONE2
*
*
******************************   FILTER 3   ******************************************
*
         SQRA      ISIN3                 PREG=ISIN3*ISIN3
         ZAC
         SQRA      ICOS3                 ACC=ISIN3*ISIN3
*                                        PREG=ICOS3*ICOS3
         APAC                            ACC=ISIN3*ISIN3+ICOS3*ICOS3
         SACH      ITONE3                STORE MAG. SQUARED IN I_O_MEM
         LAC       ITONE3,11             ACCH=ITONE3/2^5
         SACH      ITONE3
*
*
*
*
*        CHECK FOR WRAP FOR SIN TABLE AND PRODUCT BUFFER       *
*
*                  TEST THE SIN TABLE          *
*                  ******************          *
*
         LAR       AR0,SWRAP             AR0=WRAP ADDRESS FOR SIN TABLE
         CMPR      2                     TEST FOR AR5 > END OF SIN TABLE
         BBZ       $+4                   CHECK TC BIT
         LAR       AR0,SBACK             PASSED END OF BUFFER SUBTRACT SBACK
         MAR       *0-                   WRAP AROUND
         LARP      AR4
*
*                  TEST THE PRODUCT BUFFER          *
*                  ***********************          *
*
         LAR       AR0,PWRAP             AR0=WRAP ADDRESS FOR PRODUCT BUFFER
         CMPR      2                     TEST FOR AR4 > END OF PRODUCT TABLE
         BBZ       $+4                   CHECK TC BIT
         LAR       AR0,SBACK             PASSED END OF BUFFER SUBTRACT SBACK
         MAR       *0-                   PASSED END OF BUFFER SUBTRACT SBACK
*
         RET
*
************************************************************************************
************************************************************************************
```

```
**************************************************************
*       LOADER DIRECTIVE TO LOAD THE SIN AND COS TABLE INTO  *
*                    TMS PROGRAM MEMORY                      *
**************************************************************
*
*
*    INITIALISING THE SIN TABLE AND COS TABLE         *
*
*    THE FORMAT IS AS FOLLOWS                         *
*       SIN0   COS0   SIN1   COS1   SIN2   COS2   SIN3   COS3
*
          AORG      SBASE
          DATA      0,10485,0,10485,0,10485,0,10485
          DATA      9971,3240,10464,658,10299,-1964,9487,-4464
          DATA      6162,-8482,1314,-10402,-3859,-9748,-8078,-6683
          DATA      -6162,-8482,-10299,-1964,-8852,5618,-2607,10155
          DATA      -9971,3240,-2607,10155,7177,7643,10299,-1964
          DATA      0,10485,9971,3240,6162,-8482,-6162,-8482
          DATA      9971,3240,3859,-9748,-9487,-4464,-5051,9188
          DATA      6162,-8482,-9487,-4464,-2607,10155,10464,658
          DATA      -6162,-8482,-5051,9188,10464,658,-3859,-9748
          DATA      -9971,3240,8852,5618,-1314,-10402,-7177,7643
          DATA      0,10485,6162,-8482,-9971,3240,9971,3240
          DATA      9971,3240,-8078,-6683,5051,9188,-1314,-10402
          DATA      6162,-8482,-7177,7643,8078,-6683,-8852,5618
          DATA      -6162,-8482,7177,7643,-8078,-6683,8852,5618
          DATA      -9971,3240,8078,-6683,-5051,9188,1314,-10402
          DATA      0,10485,-6162,-8482,9971,3240,-9971,3240
          DATA      9971,3240,-8852,5618,1314,-10402,7177,7643
          DATA      6162,-8482,5051,9188,-10464,658,3859,-9748
          DATA      -6162,-8482,9487,-4464,2607,10155,-10464,658
          DATA      -9971,3240,-3859,-9748,9487,-4464,5051,9188
          DATA      0,10485,-9971,3240,-6162,-8482,6162,-8482
          DATA      9971,3240,2607,10155,-7177,7643,-10299,-1964
          DATA      6162,-8482,10299,-1964,8852,5618,2607,10155
          DATA      -6162,-8482,-1314,-10402,3859,-9748,8078,-6683
          DATA      -9971,3240,-10464,658,-10299,-1964,-9487,-4464
          END
```