A UNIFIED APPROACH TO THE ANALYSIS AND DESIGN OF DIGITAL LINE CODES

by

Géorgy Penchev Petkov

A Thesis Submitted for the Degree of
Doctor of Philosophy

THE CITY UNIVERSITY
Information Engineering Centre
July, 1992

02027 3131

Table of Contents

List of Tables

List of Figures

List of Abbreviations

Time-Division Multiplexing	(TDM)
Finite State Sequential Machine	(FSSM)
Pulse-Amplitude Modulation	(PAM)
Signal-to-Noise Ratio	(SNR)
Coded Mark Inversion	(CMI)
Alternating Mark Inversion	(AMI)
High Density Bipolar	(HDBn)
Pulse Amplitude Modulation	(PAM)
Stationary Memoryless Source	(SMS)
Transition Probability Matrix	(TPM)
Power Spectral Density	(PSD)
Relative Bipulse Signalling	(RBS)
Arbitrary Return-to-Zero	(ARZ)
Modified Return-to-Zero	(MRZ)
Power Spectral Density	(PSD)
Information Capacity	(IC)
Operational Space	(OS)
Pseudo-Random Binary Sequences	(PRBS)
Spread Spectrum	(SS)
Direct Sequence	(DS)
Shift-Register Generators	(SRG-s)
Code-Division Multiple Access	(CDMA)
Binary-Multiplexed Code	(BMC)

Mathematical Notation

$R_a(\tau)$	the autocorrelation function of $a(t)$
$E\langle * \rangle$	expectation density function
$S_a(f)$	the spectral density function of $a(t)$
$\delta(*)$	delta function ($\neq 0$ at a particular instant of time and $= 0$ elsewhere)
$A[*,*]$	coder state function
$B[*,*]$	coder output function
$p\{*\}$	probability function
$p\{*\|*\}$	conditional probability function
$R_a(\rho T)$	discrete-time correlation function
$W_a(f)$	spectral density matrix function
$J_a(\rho T)$	discrete-time covariance
$g(*)$	characterising (transfer) function of a system
$G(f)$	frequency response of $g(*)$
$* \rightarrow *$	left side transforms into (becomes) right side
$* \Rightarrow *$	left side implies right side
$* \equiv *$	left side identical to (coincides with) right side
$*'$	transponse of $*$
$*^{\mathrm{T}}$	conjugate transpose of $*$
$* * \ldots *$	a group of variables in a partiular order, interpreted as one entity
$* \times *$	left side multiplied by right side

ACKNOWLEDGEMENTS

I would like to thank all academic, technical and secretarial staff in the Information Engineering Centre at the City University who have assisted me in various ways during my period as a research student. I would like to express my sincere gratitude to my fellow research students for their useful discussions and particularly my supervisor Dr. R. Comley for helping me out in difficult moments and for his encouragement throughout the past few years.

I would like to thank The Telecommunications Research Institute, Sofia, and the Science and Engineering Research Council (ORS Award) for their generous financial support. Also special thanks to Dr. A. Augusty who helped me to obtain a grant from The Benevolent Fund Committee of the Institute of Physics at a critical time. I would like to thank my present employer, Nene College, for the consideration they have shown during the writing of this thesis.

Finally, and most important of all, very special thanks go to my wife, Savellya and my daughter, Martina for all their patient understanding and unstinting support, without which none of the following would have been possible.

DECLARATION

ABSTRACT

In most areas of research the variety of possible approaches to analysis and design problems is very large. This is particularly true in the case of digital signal transmission where various conflicting requirements exist (e.g. minimum bandwidth for maximum information capacity and reliability). The lack of universally adopted analysis and evaluation methods is not due to any uncertainties or deficiencies in theoretical fundamentals, rather it is a problem of diversity of criteria and therefore modes of specification that apply.

The work presented in the thesis is concerned with the creation and evaluation of a universal algorithm suitable for the assessment of digital codes together with a systematic approach to the comparative evaluation of essential structural and spectral features of coding schemes.

The thesis begins with an overview of the basic theoretical principles of line coding as an essential part of the process of channel coding for reliable and efficient digital signal transmission. A general spectral analysis procedure is derived from the finite-state sequential machine model of fixed-length block coders, and is implemented in the form of a computer program. A technique for the conversion of coder rules, given in descriptive form into table and matrix form, suitable for the universal specification format used in the general spectral analysis procedure, is developed.

A new method of general classification of codes into categories, according to their complexity levels, is proposed. A modification of the spectral analysis routine into a universal block-code generating scheme is then introduced. The virtually unlimited capabilities for the design and analysis of new code structures is demonstrated. Following from this, a new method for evaluation of the performance of block codes is suggested. It is based on the introduction of an integral parameter, the Information Capacity, which determines the degree of possible spectrum modification for a particular coder specification. Using this method, it is demonstrated how an optimal combination of a code structure, spectral features and information capacity can be achieved.

The thesis concludes with a practical example of the application of the generalised analysis procedure, demonstrating the possibility to combine code multiplexing with modification of the spectrum of the line signal. A novel technique, based on the principles of spread spectrum for multichannel transmission, is proposed. It involves a Binary-Multiplexed Coding (BMC) scheme which is implemented in a generalised circuit, the performance of which is investigated and evaluated.

INTRODUCTION

In most areas of research the variety of possible approaches to analysis and design problems is very large. The choice of a suitable method for assessment of technical solutions is often hampered by the lack of a generally accepted basis and conceptual coherence. The conclusions from the development and the implementation of new ideas are sometimes very difficult to evaluate due to incompatible specifications and unnecessary diversity of the presentation of otherwise very similar results.

The problems outlined above are very common in the area of digital signal transmission. Consequences, such as inaccuracy and reinvention, are quite likely, especially in the design and analysis of codes for digital communications. Presentations related to various issues of line coding, for example, often resort to primitive descriptions of pulse waveforms to specify the code rules, while the frequency characteristics of coded signals are approximated and incomplete.

The lack of universally adopted analysis and evaluation methods is not due to uncertainties or deficiencies in the respective fundamental theoretical developments. Ever since C. Shannon published his remarkable "The Mathematical Theory of Communications", in 1949, followed by such profound works as those of E. R. Berlekamp, 1968, R. W. Hamming, 1980 and, indeed, many others [7, 10, 18], there has always been a sufficiently sound basis for uniform and exhaustive assessment of most results in the field of information transmission and coding. One of the main reasons why certain theoretical achievements are still not in general use, is the relatively small number of simple and efficient techniques for their practical implementation.

The purpose of the research work presented in the thesis has been the creation of a universal algorithm for assessment of digital codes together with a systematic approach in the comparative evaluation of the essential structural and spectral features of different coding schemes, which would allow the superficial diversity and inconsistency in their specification to be overcome. The main results in the accomplishment of those purposes are given in chapters 3, 4 and 5 of the thesis. Chapter 3 describes the implementation of a simple and powerful computational procedure, whose accuracy and precision are defined by an elaborate theoretical model of the digital coder [18]. The universal applicability and potential of the spectral analysis routine as a useful research tool is demonstrated in Chapter 4 through a detailed assessment of many existing and proposed new coding schemes.

The specialised software algorithm has been further extended into a flexible design and simulation program with the capability to construct a theoretically unlimited number of fixed-length block codes. A fundamental result from the uniform analysis and design approach is the method for general classification of digital coding techniques, according to suitably defined complexity levels, which is developed in Chapter 5. By detailed investigation of several categories, representing different complexity levels, it is shown that all existing codes, which have been analysed, appear exactly in the expected groups and classes. The suggested classification structure, combined with the computational flexibility of the analysis algorithm, has created additional possibilities for comparative assessment of the overall characteristics of different coding schemes. The introduction of a parameter providing an integral evaluation of the structural and spectral features of codes, is described in Chapter 6. The possibility to specify a measure of the overall characteristics derives from the unique code definitions produced through the uniform analysis and design approach. It is based on graphical interpretation of the relations between the sets of symbols and coder states used to determine the finite-state sequential machine model of a coder.

An additional application of the spectral analysis routine is presented in Chapter 7. The digital transmission technique, described in this chapter, illustrates the possibility to use pseudo-random sequences for spectrum spreading in multichannel binary transmission. The results in Chapter 7 are mainly suggestive, indicating alternative sources of very efficient block codes, suitable for high speed applications, which eliminate the need for look-up tables or other memory requirements. Apart from the possibility to combine the processes of channel multiplexing and coding for spectral shaping, the results show the potential of the frequency analysis procedure, whose performance is not restricted[1] by the length of the code blocks or the complexity of the coder.

Finally, it should be noted that the presentation of the research results intends to provide a sufficient background material, included in the introductory chapters, to make the thesis largely self contained. Chapter 1 is an overview and a summary of the basic principles of coding. It has been considered essential to identify the role and the significance of line coding as an integral part of the process of improving the efficiency and the reliability of the communication channel. The author's personal views on the importance of the different types of coding are expressed through an interpretation of the common theoretical basis and the interrelations between source coding, error-control coding and line coding.

[1] Within the limitations of the computing environment.

The purpose of the extended introduction of the fundamentals of coding for digital communications is to outline the essential concepts in the use of digital structures for information transmission. The brief presentation of the underlying theory for the three types of coding aims to suggest the grounds for a unified assessment, analysis and design of codes in general. These ideas are further developed in Chapter 2 by the introduction of the main criteria for uniform assessment of coding structures and the possibility for their systematic classification.

1. DIGITAL SYSTEMS FOR INFORMATION TRANSMISSION

Transmission of signals in digital form is expected to dominate in more than 90 percent of the communication systems by the end of this century [Miller, 1988]. The rapid expansion of the variety and the volume of services provided via telecommunication networks demands continuous improvement of speed and efficiency of information processing. Satellite and fibre-optic communications are only two of many remarkable achievements in this direction.

1.1 Digital Communications, Main Objectives and Requirements

The principles of digital transmission had been known for many years before they were seriously considered for wide-spread and efficient practical implementation. Most of the factors which have contributed significantly towards the remarkable progress in the research and development of digital communication systems can be summarised in two groups[1].

A) The increased demand for information services with greater capacity and reliability:

- telephone networks have become an essential part of social life, while the integration of video and data is about to expand the information systems from public and business facilities to the premises of individual users;
- accumulation of data (scientific, business, etc.) and greater interaction in almost any area of human activity have made the distribution and exchange of information an important condition for success;
- communicating electronic systems for automated and distance operation/control have become intrinsic to many technical constructions.

B) The possibilities provided by many new technologies:

- the achievements in microelectronics and the development of components with amazing scales of integration have made higher levels of complexity accessible and manageable;
- optoelectronics and the expansion of research into open space have allowed the rapid growth of fibre-optic an satellite communication networks;
- the 'intelligent chip' and the computers provided for satisfactory levels of control and reliability of the increasingly complex structures of communication systems.

[1] A more elaborate list of reasons for introducing digital communications can be found in [Marshal, 1980].

How successfully and efficiently the digitally encoded information is transmitted over the lines of a communication network is the problem addressed in this presentation. Although the evolution period of digital systems has not been very long, the amount of research in different types of signal processing has been so substantial that separate areas have formed. Each one of them deals with problems so specific that results in the different areas have in common little more than the underlying theory. Most of these areas can be identified with respect to the functional diagram of a general communication system. Such a system is confined within the boundaries set by the concept of information transmission illustrated in Fig. 1.1.

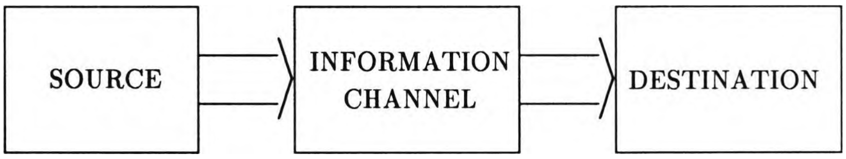


Fig. 1.1 The general communication channel

The diagram may not be very informative, but it suggests the important idea that not only the physical transmission lines or free space, but also any structure of systems and the respective signals connecting the source with the destination, may comprise the Information Channel. A more detailed representation of a digital communication system is given in the next section in order to determine the place and the role of line coding with respect to other areas of digital signal processing and transmission.

1.2 Functional Description of a Digital System for Information Transmission

In general the communication process can be described as follows. A message is 'originated' by the source, it 'travels' over the communication channel and 'arrives' at the destination. The word 'travel' has been used deliberately to indicate some important semantic notions about the terms used in describing the basic concepts of information transmission. The fact that a message 'travels' conveys one of the basic limitations in the process of communication – time. It can be expressed in the following two statements:

- The analysis of information transmission refers to finite time intervals greater than zero.
- The amount of information communicated per unit time is finite.

The limiting factor of time has its most significant implication in determining the information capacity of signals and systems based on the fundamental principles of the Information Theory, which are discussed later. The second unusual word used in the above description is 'arrives'. It indicates that the analysis of the communication process does not depend on any effects caused by the message at the destination.

At this stage it is essential to note that the engineering problems in communication theory refer to the representation of messages by sets of symbols, the statistical properties of these symbols, and the physical signals used for purposes of processing and transmission of the symbols. It is impossible to describe the whole variety of sources and symbol sets. Most are generically analogue which means that the message is embedded in the continuous-time variation of some feature of a natural phenomenon, like sound level, light intensity, etc. Another category of sources are those whose sets of symbols are discrete[2], i.e. the message is embedded in symbols which occur at discrete instants of time. These types of sources are usually artificial, like an alphabet or sampled evaluation of a continuous time process. In general the analogue information sources are studied through adequate models (e.g. a continuous-time function) of the essential characteristics of the phenomena corresponding to the possible messages. Discrete sources, on the other hand, are best described through evaluation of the statistics of the respective symbol sets which some times may require very involved application of the Probability Theory.

The models used in the theory of communication systems are based on the fact that messages are transformed into variations of some parameters of electrical signals. This is why the research and design in the area of information transmission from an engineering point of view is predominantly concerned with the development of electronic systems and the studying and implementation of electrical signals for communication purposes. The general functional diagram of a digital communication system is shown in Fig.1.2. It is assumed that the source is represented by the output of a device which generates an electrical signal. Analogously the destination is considered to be the output of the system where an electrical signal is produced so that it can be directly transformed into some recognisable effect, intended by the source. The rest of the diagram indicates most of the common processes between the source and the destination.

[2] It should be noted that so far symbols have been referred to as something intrinsic to the entity which generates the message. Further in this presentation the same term is redefined according to the theoretical model of a different part of the communication system.

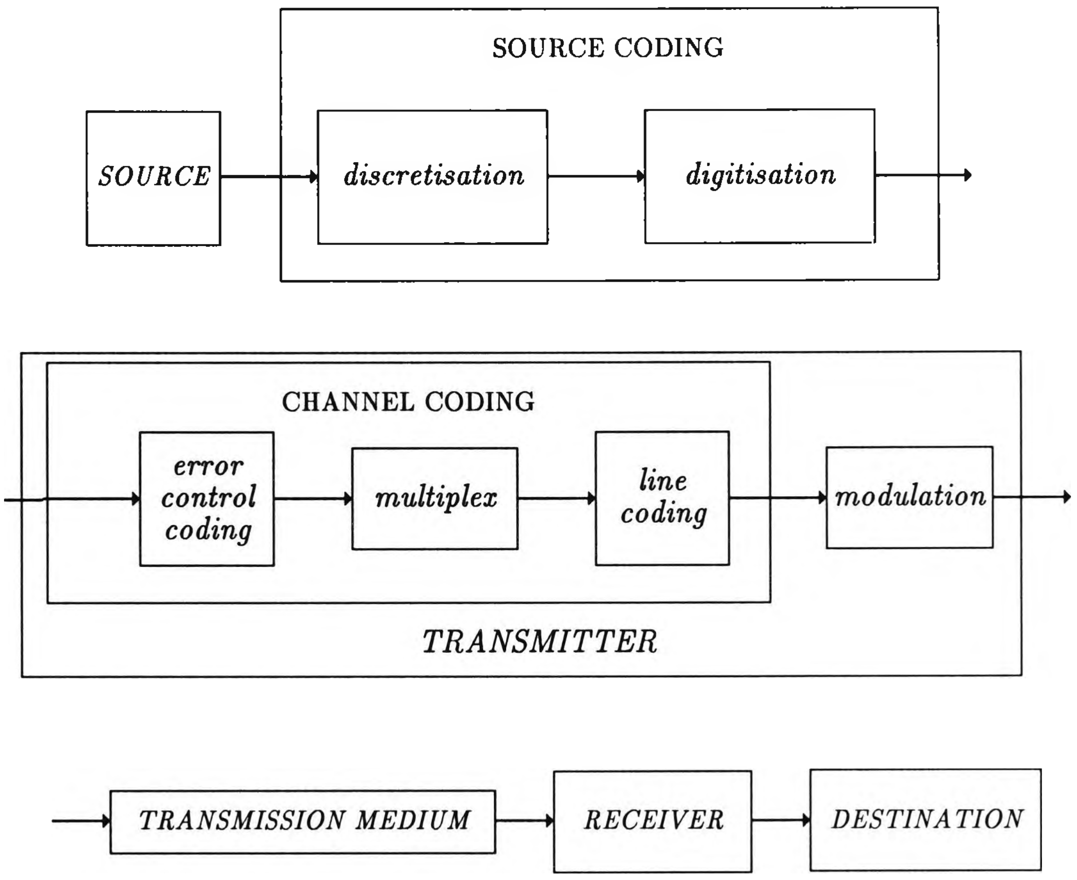


Fig. 1.2 Components of the general communication system

Fig.1.2 illustrates a conventional decomposition of the information channel into three functionally independent parts – transmitter, propagation medium and receiver. The amount of theoretical investigation and the practical results concerned with the specific processes in each of the three parts have been so extensive that separate areas of research and development have formed. The effects of various transmission media on the signals carrying information have been thoroughly studied and the diversity of the problems involved is suggested by mentioning only the main types of signal propagation environment: metal cables, free space, optical fibres. The signal waveforms propagating through a particular physical environment should have parameters allowing for minimum loss and efficient transmission of information. Topics which involve the analysis of signals suitable for various propagation media, like optical or electromagnetic waveforms are not discussed in the thesis. The research results concern specific problems of the adaptation of digital signals to the spectral characteristics of the transmission environment.

Further considerations, narrowing down the subject of the presented work and revealing its place relative to the other parts of the communication system are given below. Some assumptions, which allow the presentation to concentrate on specific topics in the theoretical investigation of line coding techniques, are worth noting.

1) The problems to be analysed in the following chapters can be confined to the transmitting side without diminishing their general applicability. This assumption is based on the fact that most solutions developed for various functions of the transmitting side are suggestive with respect of the corresponding functions of the receiving side. Indeed the principal operation of the receiver is to reverse the adequately synchronised signal transformations back to the point where they can be directly interpreted as the intended message.

2) For most purposes in the design of the transmitting part it is sufficient to specify the characteristics of the signals at the transmitter output. Therefore the analysis of digital line codes and the suggested coding techniques are not confined to particular implementations.

It is a popular view that line coding theory lacks the intellectuality and the elegance of source and error control coding which require a very involved use of the Theory of Information and Probabilities. However, it is the author's opinion that most fundamental principles of both areas also apply to the design and analysis of line codes. In fact one of the purposes of this work is to suggest the possibility of developing a common approach to the problems of coding in general. As a first step in this direction a more detailed introduction to the different types of coding is given in the following subsections, in order to outline the differences and the similarities between the respective areas. The main objective of this extended overview is to determine the essentials of source and error control coding. At the same time the underlying concepts of coding for information transmission are revealed with the intention to suggest the basis for a unified representation, design and analysis which can be applied to most coding techniques.

1.2.1 Source Encoding

Some of the ideas describing the process of digitisation are briefly presented below mainly for the purpose of completeness. These ideas may not bear a direct relation to the problems of digital line coding but they constitute the theoretical basis for the most essential difference between analogue and digital signals – the possibility of time multiplexing. In other words, the sampling principles provide the conditions for recovery of a signal from samples of it taken at discrete instants of time. Whatever the subsequent presentation and processing of the samples the fundamental achievement is the possibility to group and interleave in time discrete values of signals from one or more sources. Finally, the problems of Time-Division Multiplexing (TDM) can be interrelated to line coding as will be shown in Chapter 7.

As mentioned above, any signal can be correctly represented by a time series of discrete samples of signal values. The major question is how to define the conditions which would allow for recovery of the original signal. The answer is given by a theorem which for the case of sampling a baseband[3] signal in the time domain can be stated as follows:

> If $a(t)$ is a signal with frequency components $f_a \leq |F_a|$, it can be uniquely determined by taking at least one sample of the signal every time interval $T_s \leq \frac{1}{2F_a}$.

The above is known as the sampling theorem and $F_s = 1/T_s$ is referred to as the sampling rate. In the case of bandpass signals whose frequency range is given by $F_b \leq f_a \leq F_b + F_a$ the conditions for determining F_s are more complex. No further detail is provided here regarding sampling theory as the problems are thoroughly discussed in the literature [Peebles, 1987] and they are not directly related to the subsequent presentation.

A direct implementation of sampling theory are communication systems which employ various pulse modulation formats. In these systems the amplitude, the width or the position of a pulse waveform take values corresponding to the samples of signal. The problems addressed in the analysis and the design in this type of signal transmission involve mainly the choice of suitable pulse shape, optimal ratio between the duration of the pulses and their repetition period and what part of of the sampled waveform is to be represented by each pulse. In these systems digitisation is not performed explicitly unless all possible values of the modulated pulse stream are considered symbols of some suitably defined number system.

Quantisation is the second stage of the source encoding function of the transmitter, where the samples of the original signal are substituted with quantities from a finite subset of the range of possible signal values. The main consequence of this process is that the original signal can no longer be exactly

[3] The frequency range of a baseband or lowpass signal contains the zero frequency.

recovered from its quantised version. The values of the latter are different from the initial samples and no practical method can be applied to keep a record of every individual error. This in fact is the main concern in designing methods for efficient quantisation with minimum overall error. Signal waveforms representing natural information sources usually exhibit very little uniformity. This may result in big variations of the quantisation error if the quantising values are not suitably chosen. To overcome these problems the statistical features of the signals have to be identified and the optimum pattern of error variations determined. One possible approach is to use nonuniform distribution of the quantising values which are spaced closer in the range of signal values with higher probabilities. An even better solution are the adaptive techniques where the range and the distribution of quantisation values may vary with the changes of the original signal.

There is a great number of factors which influence the process described above. The main problem, however, is to determine the optimum balance between precision and symbol transmission rate. The engineering area of research investigating this problem is commonly referred to as signal processing. It has been developed extensively in recent years and amazing results have been achieved by using powerful computational methods and fast computers. Most of the efforts are directed towards analysis of the information content of signals representing the 'natural' sources of information and the possibilities to reduce eventual redundancies. Another problem studied in this respect is how to redistribute the information of a source in order to use the capacity of the information channel more efficiently. This problem can be successfully dealt with by the use of source coding.

While the purpose of source coding is different from that of line coding, significant similarities can be found in the underlying principles of the two types of signal processing. Both are basically concerned with devising suitable transformations of sets of symbols into different set of symbols, in order to meet certain requirements of the communication system. Furthermore, by specification of most important problems in developing appropriate coding techniques, it can be shown that source coding and line coding have common objectives and requirements.

The most essential relations used in the description and the analysis of symbol sets are introduced below through a brief reference to the basics of number systems. These relations reveal some fundamental problems of coding in general and establish a common basis for the analysis and design of different codes.

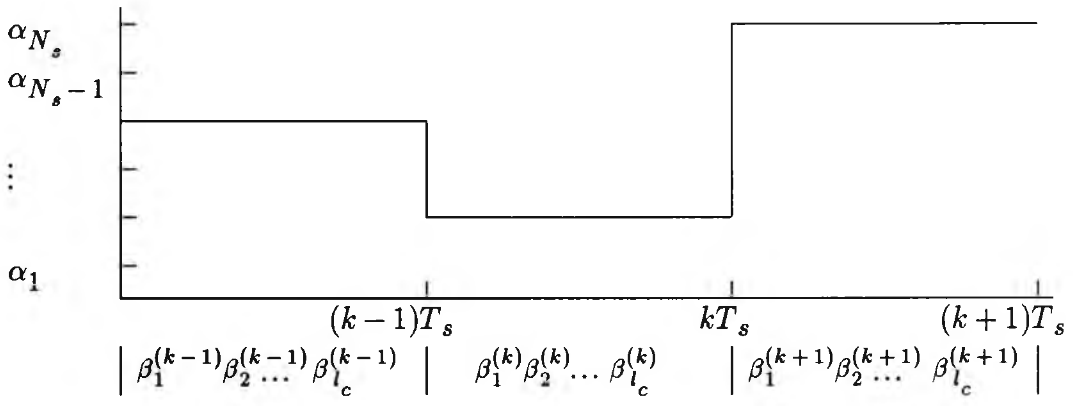
A precise definition of a number system is not considered essential for the purposes of this introduction as most of the ideas will be presented in a descriptive manner. A more rigorous mathematical approach to the problems of coding theory is given in [Lin, 1983; Berlekamp, 1968]. It is sufficient to say that quantities are represented by numbers and, in order to determine some relations between different quantities, a suitable set of operations over all possible numbers is defined. Numbers are specified through a finite set of digits (symbols) which define the base of a particular number system. Thus a useful presentation of the number N from a K-digit system with symbols $\alpha_1, \alpha_2, \ldots, \alpha_K$ is given by

$$N = \alpha_n \alpha_{n-1} \ldots \alpha_1 = \sum_{i=1}^{n} \alpha_i (K^{i-1})$$

where $\alpha_i \in \{\alpha_1, \alpha_2, \ldots, \alpha_K\}$ and n is the number of symbols used for the notation of N. In the process of source coding the initial digital signal is substituted with some other digital signal whose values are more suitable for the subsequent processing. If N_s and N_c are the numbers of all possible values of the original and the coded signals respectively, and K_c is the number of symbols (signal levels) used to represent the coded signal, then the following requirement has to be fulfilled

$$N_c = K_c^{l_c} \geq N_s = K_s^{l_s} \tag{1.1}$$

where l_c is the number of coded symbols replacing l_s source symbols. In most practical systems the number of source levels is $K_s = N_s$ and each level is viewed as a distinct symbol. Usually one source symbol ($l_s = 1$) is replaced with $l_c > 1$ code symbols. This process is illustrated in Fig. 1.3.



α_{N_s}
$\alpha_{N_s - 1}$
$\vdots$
α_1
$(k-1)T_s$
kT_s
$(k+1)T_s$
$\beta_1^{(k-1)}\beta_2^{(k-1)}\ldots\beta_{l_c}^{(k-1)}$
$\beta_1^{(k)}\beta_2^{(k)}\ldots\beta_{l_c}^{(k)}$
$\beta_1^{(k+1)}\beta_2^{(k+1)}\ldots\beta_{l_c}^{(k+1)}$

Fig. 1.3 Symbol transformation in source coding

Many real systems are based on binary coding ($K_c = 2$). In general, binary digital systems translate the source symbols into sequences of digits usually represented as 0 and 1 or -1 and $+1$. If the number of source symbols is $K_s > 2$ and every one of them ($l_s = 1; N_s = K_s$) is coded by a block of binary digits (a code word), then from (1.1) it follows that the minimum length of the binary code words is limited by $l_c \geq \log_2(N_s)$, (assuming that l_c is not variable).

At this stage the most crucial question arises: How to choose the code. The answer is given by the Information Theory. The fundamentals for precise quantitative evaluation of information capacity have been laid by C. Shannon in his mathematical theory of communication [Shannon, 1949]. The main aspects of this theory require more involved discussion and are briefly presented in a following section. Without going into detail, it suffices to mention that a great amount of scientific thought and engineering effort has been devoted to the analysis and development of many different coding formats.

The most simple example is natural binary coding, where the source symbols are represented as decimal numbers ($K_s = 10$) and each one is replaced with the respective binary representation of that number. Obviously if the number of source symbols is not an exact power of two, the possible binary words are more than the source symbols as required by (1.1). Therefore some codewords are not used, which means that the amount of information conveyed by the code is less than the amount which could be represented by the complete set of binary words. This illustrates the first problem to be addressed by coding theory, i. e.:

1) How to devise an efficient code (one whose redundancy is as small as possible)?

An example is Gray coding, which assigns binary words in such a way that for transitions of the source signal between adjacent levels, the corresponding binary blocks differ by one digit only. This requirement indicates another problem which is to be solved by the source code:

2) What is the best correspondence between the source symbols and the code words?

There are various types of Gray codes as well as many other codes which provide for particular patterns of assigning the binary blocks to the source levels so that some structure of the input signal is preserved or changed in a specific way. A general rule in answering question 2 cannot be offered easily because there could be as many different requirements as possible types of source signals. One simple example of such a generalisation is based on the assumption that it is essential to have as many transitions in the coded signal as possible. In this case the binary words with short strings of identical symbols substitute the source values which occur more frequently.

The above considerations indicate that the statistical characteristics of the source are of great significance in digital coding. This point becomes more apparent when noting that for most practical cases it is impossible to study all combinations of source symbols, therefore a powerful method of analysis is the evaluation of signal probabilities. The results achievable by this approach suggest the answer of the most important question of digital communication:

3) What is the optimum source coding?

The last problem is considered fundamental to efficient digital signalling because no subsequent processing can improve on this efficiency (in terms of information content). In fact the tasks of all following stages in the digital communication system can only be achieved by adding more information and thus decreasing the efficiency with respect to the source information.

Before proceeding with the overview of the next block in the general communication system from Fig. 1.2, a few more considerations are given with respect to source coding. Having introduced the notion of unequal-source symbol probabilities, a further step in improving the efficiency of coding is made by the use of variable-length codes. It is obvious that the smaller the blocks of code symbols for representing a given amount of information the more efficient that code is. With fixed-length codes the limitations in this respect have been indicated above. For codes with variable-length words the corresponding measure is the average number of symbols per code-block. The advantages of variable-length codes can be suggested by considering the possibility to use short words for the most likely source symbols and longer ones for the less probable symbols. The immediate and most significant implication is the problem of 'undoing' the coding. This is why the first steps towards investigating this type of code is to establish the conditions for recognising the code-words and correct recovery of the source signal.

These ideas open the vast area of the coding theory and it is beyond the scope of this work to give a detailed presentation of all aspects of coding for information transmission. There are many literature sources where the fundamentals of the Information Theory and Coding are extensively revealed, a classic example being [Hamming, 1986]. In order to complete the description of the source coding function it is sufficient to mention a few more essential points. The problems outlined above are approached through

a) establishing the quantitative limitations of the characteristics of the codes

b) devising the coding rules and techniques for their practical implementation.

A simple example of a) but a very essential result is the Kraft inequality which concerns the existence of instantaneous codes. Two preliminary definitions have to be made before this result can be given:

D1. A code is uniquely decodable when any sequence of code blocks represents only one sequence of source symbols.

D2. A code is instantaneous if every code word is recognisable as soon as its last digit appears.

An immediate consequence of D2 is that no part of a code word, which starts from its beginning, is identical to another code word. It is clear that if a set of source symbols is coded with a variable-length code the main problem of decoding the message is resolved if the code words are instantaneously identifiable[4]. A binary code example $(\beta_1 = 0, \beta_2 = 1)$ for six source symbols $\alpha_1, \alpha_2, \ldots, \alpha_6$ is shown below

α_1	=	0	α_4	=	1 0 1 0
α_2	=	1 1	α_5	=	1 0 1 1 0
α_3	=	1 0 0	α_6	=	1 0 1 1 1

In general there is more than one set of code words with different lengths which are uniquely decodable for a given number of source symbols. The Kraft inequality gives a condition whether an instantaneous code for N_s symbols can be found. Such a code exists if the lengths of the code blocks $l_1 \leq l_2 \leq \ldots \leq l_{N_s}$, satisfy the following

$$\sum_{i=1}^{N_s} \frac{1}{K_c^{l_i}} \leq 1, \qquad \text{where } K_c \text{ is the number of code symbols.}$$

The important issue stated as b) above is remarkably illustrated by another classic example - the Huffman codes. The Kraft inequality can tell if an instantaneous code exists for a particular combination of code-word lengths, but it

[4] In principle uniquely decodable codes, which are not instantaneous can be used for variable-length coding but many problems arise when they are implemented (delays, memory requirements, etc). In addition it can be proven that for any set of source symbols an instantaneous code with the highest possible efficiency exists. Therefore nothing is gained (in terms of efficiency) by using codes other than instantaneous.

does not help in finding the actual code words. Such a code can be constructed by the use of Huffman coding. The rules are simple and the result is an efficient code.

There are many other types of source coding and a great variety of codes have been devised. Although in most practical systems multi-level signals are coded in binary form, it is also possible to use codes with more than two symbols. In general in the process of source coding blocks of source symbols are converted into blocks of code symbols. The coding considerations presented so far refer to source signals with independent symbols. More complex structures emerge if the source symbols are correlated. Codes which account for such cases are based on the theory of Markov processes. This type of source coding is referenced in the next section, where greater attention is given to the fundamentals of information theory.

1.2.2 Channel Coding

The second main function of the communication system from Fig. 1.2 is to condition the digital signal for transmission over the information channel. The complexity of the theory underlying channel coding compares with that of source coding and the remarkable achievements in recent years have produced large subareas of research. The need for conditioning implies that one way of sending a signal is better than another. Obviously the best way of transmitting is to secure that what is received at the destination is identical to what has been sent. Unfortunately this is impossible for real information channels due to the presence of noise. The effect of noise in digital transmission is errors[5]. Thereof the most important objective of channel coding is to counter the effects of noise by keeping error-probability within controllable limits. The only way of doing this is to add information which allows detection and possibly correction of the errors.

The general problem is again how to devise an efficient code. As the goal is to detect and eventually correct errors at the expense of adding more information, it is obvious that efficient code is one which for a given number of detected and corrected errors requires minimum increase in information. The solution of the problem is approached again through the following two stages:

A) establishing the limitations in what could possibly be achieved through coding and

B) devising methods for the construction of codes.

[5] The word suggests the discrete nature of the phenomenon which is generally referred to as distortion and describes the same effect for continuous-time processes.

A fascinating result in the direction of A) are the mathematical fundamentals of communication theory first published by C.E. Shannon in 1948. The concepts of this theory have been further developed and implemented to produce a number of application areas so large that even a general overview will take a considerable effort and space. This is why the principles of error detecting and control codes are only briefly outlined here and a few examples illustrating these principles are presented. An indepth treatment of the subject is made in many books a good example of exhaustive work being [Lyn and Costello, 1983].

The general theory of codes for error detecting and control has been developed for multi-level as well as binary coding, however only the latter are considered in this presentation. There are two main groups of codes in use at present: block codes and convolutional codes. Common for both types is that every l symbols of the coded output signal represent n symbols of the input sequence. For binary coding $l > n$ and the code efficiency is defined as $\frac{n}{l}$, while $1-(\frac{n}{l})$ is the code redundancy. The main difference between conventional block codes and convolutional codes is that the latter require memory. In other words, when no memory is used, every coded block depends on the input block of symbols only, while in systems with memory the output words of l digits depend on previous parts of the input signal as well as on the present n digits. The analysis of the capabilities of various codes spans from the simple parity checking to the complexity of trellis diagrams and Viterbi techniques [29]. The only division in this brief presentation of codes for error detection and control is into memoryless codes and coding techniques which require memory. The main results revealing important theoretical limitations and examples illustrating typical coding structures interleave throughout the following two sections.

1.2.2.1 Memoryless Coding

Probably the simplest ways of detecting errors are symbol repetition and single-digit parity checks. In the first case each source symbol or group of symbols is repeated several times. At the receiving side the original signal is reconstructed by assuming that the majority of the repetitions have arrived correctly. This method requires much redundancy and is far from efficient. Single parity checks are used more often in practical systems, mainly because of their simplicity of implementation. Usually the output blocks contain an additional digit (0 or 1) so that the total sum of 1-s becomes even or odd. The code efficiency is better but no even number of errors can be detected and it is only possible to find out the occurrence of an odd number of errors without knowing whether it was one or more. The single-digit parity check coding is mostly used in systems with relatively low probability of error along the channel and where erroneous messages can be ignored or requested for repetition. In fact the probability of error is a very essential issue which often determines the suitability of codes for certain applications and makes the same codes unsuitable in other cases. In general, codes are designed for independent errors with uniform distribution, although special types have been developed for particular kinds of noise environment where errors may appear in burst patterns or have some correlation.

With many applications it is essential to detect the presence of more than one error and also to correct them. How many errors can be detected and eventually corrected depends on the amount of information added to the original signal. This amount is limited by the number of digits added to the blocks of input symbols but how close to this limit one can get depends on the pattern of coded words. In the case of binary-to-binary coding[6] the number of additional symbols, required for correction of any combination of k errors (or less), cannot be smaller than a quantity given by the following relation

$$l - n \geq \log_2 \sum_{r=0}^{k} \binom{l}{r}, \qquad \text{where } \binom{l}{r} = \frac{l!}{r!(l-r)!}$$

is the respective binomial coefficient. There is little doubt that with a sufficiently large number of additional symbols it will be possible to construct a code capable of correcting k errors. The interesting question is what is the code which does this with the minimum number $l - n$ given by the equality in the above relation. A remarkable answer to this question and a good example of solving the problems of stage B) defined above, are the Hamming codes which belong to a broader class called perfect codes.

The cyclic codes are another class which exhibit very useful features and have received considerable attention since they were first studied in 1957. The first of their important characteristics is evident from the definition which specifies that every code word of a cyclic code can be derived from another code word by shifting all symbols of the latter one position to the right and moving the last symbol to the first position. This indicates the possibility to implement cyclic codes in shift-register circuits which can easily produce a great variety of codes with not very large numbers of registers and many combinations of feedback connections. Another important feature of the cyclic codes is their inherent algebraic structure. This feature can easily be analysed and used by representing the symbols of the code blocks as coefficients of polynomials. Although the algebra

[6] Binary coding is assumed everywhere in the thesis without a special notice except if otherwise is mentioned explicitly.

of the code polynomials can be developed into very complex results, one essential outcome is the possibility of using generator polynomials in construction of codes. A disadvantage of the cyclic codes is that not all generator polynomials represent useful codes and it is not very easy to identify the ones which do. In spite of that cyclic codes still receive considerable attention due to the relatively simple use of shift registers for most operations. Finally, various designs of cyclic codes have proven successful in countering both random and burst types of errors.

An interesting class of cyclic codes are the BCH codes (named after Bose, Chaudhuri and Hocquenghem) known since 1960. Many coding theorist have studied and generalised the properties of the BCH codes and some of the significant contributions have been made by Peterson, 1960 and Berlekamp, 1968.

1.2.2.2 Codes with Memory

While the pair of integers (l,n) specifies a block code where any output word of length l depends only on the input word of length n at the same instant of time, the convolutional codes are specified with a set of three integers (l,n,m), where m denotes the memory order of the code. The numbers l and n specify the lengths of the output and the input words respectively, but the output blocks depend on m previous input blocks as well.

The convolutional coder can be viewed as a sequential machine. An important point regarding the approach adopted in the thesis is that most practical analyses of digital coding refer to finite element discrete devices. Systems based on such devices can only have a finite size of memory, therefore a finite number of possible states. As it will be shown in greater detail later, the Finite State Sequential Machine (FSSM) proves to be a very successful model of a digital coder with memory. There are three sets of symbols with a finite number of elements which are used in the digital coder model. These are:

- the set of all possible input blocks of symbols,
- the set of the allowed output blocks and
- the set of the coder states.

The complete description of the model requires also the rules which determine:

- the output words corresponding to the combinations of an input word and a coder state;
- the state into which the coder goes following a combination of an input word and a coder state.

These rules are usually given as suitably defined functions. The state transition diagram is another way of describing the coder rules. A convolutional coding scheme specified by (l, n, m) produces an output word of length l for every input word of length n according to the combination of m previous input words. All distinct combinations of m previous input words correspond to different contents of the coder memory, i.e. different coder states. When a convolutional code is represented as a set of parallel shift registers (one for every input-word symbol) with lengths $0 \leq M_i \leq m$, $i = 1, \ldots, n$ the total number of distinct coder states S_M is given by the product of all possible states of each shift register

$$S_M = 2^M = 2^{\left(\sum_{i=1}^{n} M_i\right)} \tag{1.2}$$

(The number $M = \sum_{i=1}^{n} M_i$ is usually called the coder memory.)

It is possible to have different coder configurations with the same memory M. All of these coders can be represented by an oriented graph with S_M nodes. Every node is a coder state and the transitions between any two states is indicated as a pointed connection labelled with the input/output word pair.

The coder state diagrams are very a useful tool for analysis of convolutional codes by investigating various paths (unbroken sequences of branches) describing the coder's behaviour. The sequential and the Viterbi algorithms are typical examples of such tools, [11]. They are based on an extension of the state diagram in the time domain and the resulting graph structures are called tree and trellis diagrams. The graphs for both diagrams start with a node representing some initial coder state. There are 2^n branches leaving the initial state - one for every possible input word. The branches arrive at nodes representing all coder states into which the coder can go after the initial state. The same applies to every next state. The branches are labelled with the output words corresponding to the input word producing the branch and the state it comes out from. The nodes of the graph arrived at after every branching correspond to the time instants where the input word changes. After m consecutive time intervals the resulting nodes will be all possible states of the coder. This follows from the fact that each node represents a path of branches corresponding to a distinct combination of $m + 1$ consecutive input words. Obviously all new branches will lead to an existing coder state. From this point onwards the tree and the trellis diagrams differ. In the sequential algorithm all new branches of the tree result in separate nodes whose number is $2^{M + kn}$ after $(m + k + n)$ input words. In the trellis diagram all new branches leading to the same coder state are connected to one node of the graph.

For any convolutional coder a particular input sequence results in a path comprising consecutive branches in the respective tree or trellis graph. The performance of the code depends essentially on the existence of efficient decoding algorithms. At the receiving side the coded sequence arrives with some symbols being in error. The received sequence has to be decoded so that the source digits are recovered as accurately as possible. Most methods are based on the principle of maximum likelihood decoding where a coded sequence is sought, which differs in the least number of symbols from the received sequence. To achieve this goal the coder graphs are presented in various ways which attribute probability measures to different paths. Then most likely paths are computed on receiving the coded sequence and the path (i.e. the suggested code sequence) which is closest to the received one, is accepted. Typical examples of decoding methods are the Fano and Viterbi algorithms, for sequential (tree) and trellis codes respectively [10]. The theory behind the estimation of various techniques and the construction of efficient convolutional coders is rather complicated and the major limitations in practical systems are the size of the coder memory and the number of computations.

1.2.2.3 Line Coding

The main strategy of digital transmission has been outlined in the previous sections. In summary it consists of adding more digits and/or correlating blocks of digits in order to detect and correct errors caused by noise in the transmission channel. For binary systems this is achieved at the expense of a higher signalling rate. Error control coding provides possibilities only for correcting a number of errors when they occur. Some codes are specially designed to counter particular types of errors but in general channel coding does not reduce the susceptibility of the signal to errors. This is why further processing of the digital signals before transmission is often employed. Although in general it does not require the precision of algebraic coding, the effects of line coding can contribute to the efficiency and reliability of communication. The sophistication of most methods for error control and the associated theory have made channel coding an area of communications theory in its own right. Only matters closely related to the theory of information are usually regarded as typical problems of this area. However, it is possible to define the communication channel with respect to various parts of a digital system for information transmission. The information at the input to the 'channel coding' block of Fig. 1.2 may be regarded as represented by a sequence of digits and everything before that point as a digital source. The physical propagation medium may be defined as the digital transmission channel. Therefore the purposes of channel coding may be viewed in a broader aspect:

Firstly *to enhance and modify the digital information sequence so that occurrence of up to a certain amount of errors does not affect the original information contents.*

Secondly *to restructure the digital sequence so that it is less susceptible to degradation due to the limitations of real transmission media.*

The second purpose is not directly related to error control techniques but it can be associated with channel coding and the motivation behind this view is briefly given below. Error control implies that digital information sequences are processed to counter unwanted effects produced by the transmission medium. These effects are defined as errors and their pattern of occurrence is what represents the communication channel in terms of error control coding. In other words, for a given transmission environment a certain level of deterioration is specified, which is evaluated by the amount of possible errors and their distribution in time. The error control coding provide information which is added to the signal in such a manner that when the deterioration during transmission is below the specified level the unaffected amount of information is sufficient to recover the original source sequence. The coded signals are not always suitable for direct transmission in a sense that the amount of deterioration caused by the propagation medium would be higher than the code can cope with. It is possible to apply additional coding to transform the digital sequence so that, when it is represented by a particular signal waveform which is transmitted over the physical channel, the deterioration would result in a manageable amount of errors. This type of processing is referred to as *line coding.*

An essential parameter of a digital signal is its frequency spectrum. The same parameter, attributed to the transmission channel, reflects how much distortion would be caused to signals with different frequency spectra. The essence of line coding is to add information to the digital sequence so that the frequency parameters of the resulting signal match those of the transmission medium. However, in some practical cases there are additional considerations for implementing a line code which may not be directly related to the frequency parameters of the signal. In summary:

Error-control coding *provides protection from loss of source information under certain level of distortion caused by nonideal characteristics of the transmission channel.*

Line coding	*adapts the parameters of the digital signals to those of the transmission channel so that overall distortion does not exceed the specified levels.*

In this respect both types of coding can be regarded as part of the channel coding function in the general model of communication system. As line coding is the main topic of this work, a more detailed presentation follows in subsequent chapters.

It should be noted that channel coding does not provide the complete conditioning of the digital signal for the purposes of transmission. The problem of adapting the signals to the transmission medium is also solved by selecting suitable waveforms to represent the different code symbols. Various types of pulse waveforms exist to match different propagation conditions. The system which produces the appropriate signal waveform representing the coded sequence is briefly discussed in the next section.

1.2.3 Digital Modulation

There are two major types of transmission: baseband and carrier-modulated. The results presented in the thesis are not directly related to any particular method of signalling, although baseband transmission is implied by the use of a pulse amplitude modulator in the line coder model, described in Chapter 3. This choice has been made for completeness of the analysis and it is a reasonable approximation to assume that many real systems employ this kind of modulation. For example, most fibre-optic communications are based on direct intensity modulation of the light source.

The problems of digital line coding and digital modulation can be discussed quite independently. Line coding alters the structure of the digital sequence to produce a desired redistribution of the frequency components or particular shape of the frequency spectrum of this sequence. Digital modulation results in generating a series of signal waveforms representing the symbols of the coded sequence. The waveforms are completely specified for the time interval of each symbol and have particular spectral characteristics. The latter modify the frequency characteristics of the coded sequence often to improve the adaptation of the signal to the transmission channel.

The problems of coding are analysed by models where information is represented by the statistical parameters of sequences of symbols. What matters to the analysis are the probability functions and the time intervals related to symbols and groups of symbols, while their meaning is generally irrelevant. After the final sequence of digits is produced for transmission the parameters of the real processes which represent the different symbols become important to the efficiency of communication.

The simplest modulation technique is to transmit a binary sequence as two different signal levels (voltage, power, etc.), whose values are constant for the time interval T_c of one symbol. These levels can take any two of the values $+L$, $-L$ or 0. In many real systems the duration of the symbol interval is much greater than the time for transition between the different signal levels which allows the latter to be ignored in a general model of the waveforms used in binary transmission. If 0 and $+L$ are the signal levels chosen to represent the two symbols 0 and 1 respectively, the signal waveform can be analysed as a rectangular pulse which is shown in Fig. 1.4. This model is often used as a common basis for analysing the spectral characteristics of the coded sequences as it is a close approximation for many practical systems.
The technique of signalling by a pulse waveform with a particular shape is referred to as Pulse-Amplitude Modulation (PAM), because it can be viewed as a sequence of pulses whose amplitude assumes different values for the different symbols of the sequence. If the encoded sequence of symbols at the input of the modulator is $x_r \in \{0,1\}$, then the modulated signal $y(t)$ can be expressed as

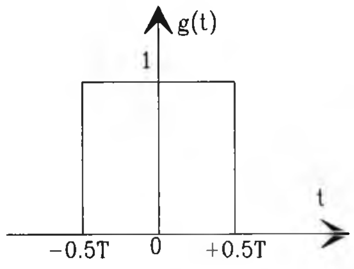


Fig. 1.4a Line pulse waveform

$$y(t) = \sum_{r=-\infty}^{+\infty} x_r g(t-rT) \tag{1.3a}$$

where $g(t-rT)$ is the time function of the basic pulse shape. When this function represents an ideal rectangular waveform it can be defined as

$$g(t-rT) = \begin{cases} 1, & \text{for } |\, t-rT \,| \le \frac{T}{2} \\ 0, & \text{for } |\, t-rT \,| > \frac{T}{2} \end{cases} \quad , r = 0, \pm 1, \pm 2, \dots \tag{1.3b}$$

The essential parameters of the basic pulse waveform are its duration, which is assumed to be the code-symbol period T and its frequency spectrum $G(f)$. For the purposes of normalised assessment of the spectral characteristics of coded sequences in later chapters the spectral density function of the ideal rectangular pulse have been used. $G(f)$ can be expressed as the Fourier transform as follows

$$G(f) = \int_{-\frac{1}{T}}^{+\frac{1}{T}} g(t-rT)\, \mathrm{e}^{-j2\pi ft} \mathrm{d}t = -\frac{1}{j2\pi f}\left[1 \times \mathrm{e}^{-j2\pi ft}\right]_{-\frac{1}{T}}^{+\frac{1}{T}} = -\frac{\left[\mathrm{e}^{-j\pi fT} - \mathrm{e}^{j\pi fT}\right]}{j2\pi f} =$$

$$= \frac{1}{\pi f}\left[\frac{e^{j\pi fT} - e^{-j\pi fT}}{j2}\right] = T\,\frac{\sin(\pi fT)}{(\pi fT)} = T\mathrm{sinc}(\pi fT) \qquad (1.3c)$$

The frequency function is shown in Fig. 1.4b. How exactly the spectrum of the basic pulse interacts with that of the coded sequences will be shown in greater detail in the presentation of the line coder model (Chapter 3).

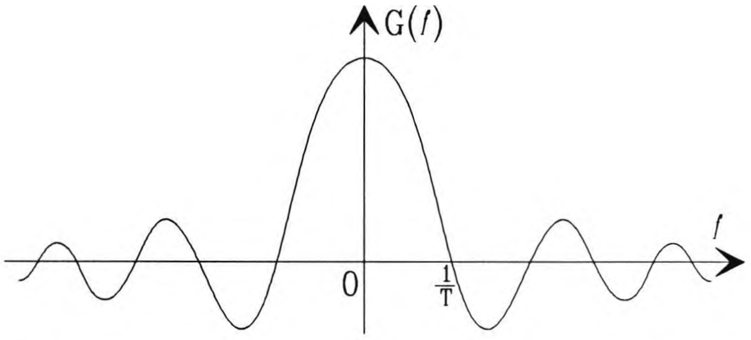


Fig. 1.4b The spectrum of a rectangular pulse waveform of duration T

In real systems the pulse waveforms undergo considerable changes during transmission over the communication channel. Due to the greater attenuation and time delays of various frequency components the original pulse shape is changed and is no longer confined to its time interval. The occurrence of intersymbol interference is very often causes errors and measures have to be taken to avoid it. Usually this is achieved through using pulse waveforms with particular shapes of their time function. The relevant theory has been extensively developed and a great variety of waveforms have been suggested for transmission channels with different frequency characteristics. The major limitations are determined by the Nyquist theorem and quantitative assessment is based mainly on the Fourier relation between the time/frequency functions.

The symmetry of the Fourier transform $g(t) \leftrightarrow G(f)$ illustrates the requirements which the pulse waveforms and their spectral characteristics should satisfy. If the channel is band-limited, then a waveform with an ideal 'rectangular' spectrum is the best possible choice. The inverse Fourier transform shows that the time function of such a pulse has the shape of the spectrum of a rectangular pulse waveform (see Fig. 1.5). Obviously the modulated sequence in this case will have components of every pulse outside its time interval. In practice various methods are used to produce pulses with particular shapes. Some of them involve combining the sinc-function waveform with its delayed versions, others use special types of filtering. In general the main results pursued by pulse shaping are to reduce intersymbol interference and to have a waveform with suitable frequency distribution.

1.3 Information Theory – Limitations to Coding

The Theory of Information does not provide means for direct assessment of the results achieved by line coding, which is the main subject of the forthcoming presentation. However, there are essential considerations about information capacity and bandwidth limitations which are applicable in every aspect of digital coding. This is why certain fundamental relations are outlined in this section, in order to provide the basis for some definitions and assumptions given later. An important aspect of digital communications is the probabilistic nature of most measures concerning the characteristics of discrete signals and sequences of symbols. A relevant basis for the analysis of digital systems for information transmission has proven to be the theory of markovian processes.

1.3.1 Quantity of Information

In digital communication it is very important to have a measure of the amount of information being processed at any stage. For a discrete system this measure is provided through a description of the sets of symbols the system can recognise and/or generate. The definition of information content of messages constructed as sequences of symbols is a crucial step in achieving quantitative assessment of information transmission. Many authors introduce this definition by giving various examples of situations with a number of possible outcomes. Most descriptions imply that people perceive likely outcomes as conveying little information, while the occurrence of events which are not very probable is considered very informative. In spite of their great variety, most examples used in the literature to introduce the definition of information, can be summarised, [9] as follows:

> *Information, from a communications point of view, represents the amount of uncertainty or the freedom of choice associated with a number of events and not the meaning or the significance of these events to an individual.*

The following assertions can be made in order to construct a meaningful definition of Information Quantity: (1) The amount of information is attributed to a particular situation with respect to the possible changes of this situation. (2) The changes are referred to as events and can be represented by sets of symbols. (3) The process of occurrence of events is represented as sequences of symbols and corresponds to variations in the amount of information. (4) The quantity of information conveyed can be a measure of the changes that have occurred.

However, there is a semantic problem in the definition of information as it is impossible to identify the entity to which the quantity of information is attributed. This entity could be referred to as an event or represented by a symbol but it cannot be defined, because the definition itself is an event and is represented by symbols which should have been defined. Clearly no definitive beginning can be used other than the intuitive assumption of what is an information source and the set of symbols representing that source.

At this point it is possible to introduce the probabilities $p_1, p_2, \ldots, p_n$ associated with a set of symbols denoted as $\{a_1, a_2, \ldots, a_n\}$. The probabilities are numbers which correspond to the symbols in the following manner:

> if a_i represents an event which is more likely to occur than the event represented by a_j, then $p_i > p_j$;
>
> if a_i represents an event which is certain to occur (i.e. all other symbols a_j, $j \neq i$ represent events which will not occur), then p_i=1 and p_j=0, $j \neq i$;
>
> if a_{ij} represents the joint occurrence of two events a_i and a_j, which are independent then the probability of a_{ij} is p_{ij}=$p_i p_j$.

Under the above assumptions the following assertions can be made regarding information quantity:

1) The amount of information (the possibility of change) I_i associated with a_i is greater than the amount of information I_j associated with a_j if $p_i < p_j$ (i.e. the event of a_i is less likely than the event of a_j).

2) Similarly I_i=I_j if p_i=p_j.

3) The amount of information I_i=0 if p_i=1, i.e. a_i is certain to occur (no freedom of choice or possibility of change).

4) The amount of information I_{ij} associated with a combination of the two symbols a_i and a_j is given by $I_{ij} = I_i + I_j$ when the corresponding events are independent[7].

[7] The combinations of two symbols are more than the single symbols, therefore the freedom of choice, hence the information is greater.

The assertions above suggest that the amount of information can be represented as a function of the symbol probabilities. From 1) and 3) it follows that information is inversely related to these probabilities. The most adequate functional relation between information quantity I and symbol probability p is determined as

$$I(p) = \log \frac{1}{p} = -\log p, \qquad \text{for } 0 \leq p \leq 1.$$

Base 2 is usually adopted for the logarithm above because it has been widely accepted as convenient for most applications. It also gives the common unit for a quantity of information - bit. This can be illustrated by assuming two possible events n=2 with equal probabilities of occurrence $p_1 = p_2 = \frac{1}{2}$. Then the amount of information associated with any of the two events a_1 and a_2 is

$$I_1 = I_2 = \log_2 2 = 1 \text{ bit}$$

The entropy is an important quantity which associates an integral measure of information with a set of symbols $\{a_1, a_2, \ldots, a_n\}$ and is given by

$$H(a) = \sum_{i=1}^{n} p_i \log_2 \left(\frac{1}{p_i}\right) \text{bits/symbol}$$

$H(a)$ gives an estimate of the average amount of information per symbol over the whole set of symbols. A careful observation of the last expression shows that for a given number of symbols n the entropy $H(a)$ varies with the probabilities of the symbols. It can be shown that its maximum value is attained only when all symbols have equal probabilities $p_i = \frac{1}{n}$ for $i = 1, \ldots, n$.

The great importance of the entropy function is the quantitative limitation it gives to source coding. In fact $H(a)$ determines the smallest average number of binary digits per symbol which can be achieved. Thus the entropy can be used to measure the efficiency of a code.

1.3.2 Channel Capacity

Information theory describes communications in two main aspects. The first aspect concerns the functional relation between information content and the probabilities of symbols representing messages. The quantity of information associated with different sets of symbols and probabilities can be evaluated. This allows for accurate estimation of what is achievable through coding. The second aspect is the actual transmission of information. It regards communication as a process which takes place in time and in conditions where loss of information is greater than zero.

In digital communications information is transmitted as sequences of symbols. Transmission cannot take place instantaneously as every symbol is represented as a signal waveform with a finite duration grater than zero. Therefore only a finite amount of information can be transmitted per unit of time. This limitation can be expressed as a number of symbols per unit time interval, which is uniquely related to the frequency of occurrences of the different symbols. The various frequencies related to a particular sequence of symbols comprise the spectrum of that sequence and determine its frequency bandwidth.

The second significant limitation is the inevitable loss of information in real transmission systems. This factor is usually represented as the signal-to-noise ratio (SNR) and its effect on digital signals can be evaluated through the probability of a number of symbols to be in error.

The rigorous mathematical model of information transmission over a channel with noise is beyond the scope of this presentation. Only a few of the main considerations will be given below in order to illustrate the fundamental limitation imposed by finite bandwidth and signal-to-noise ratio.

It is essential to note that the communication channel can be characterised completely by the set of conditional probabilities associated with the symbols as they have been changed by the noise of the channel with respect to the symbols prior to transmission. The information I which is transmitted over the channel can be defined as the information of the original set of symbols, given by their entropy H, less the entropy H_e of the same set of symbols, estimated from the symbols after transmission, i.e. $I = H - H_e$. (H_e is known as the equivocation.) Clearly I represents the average transmitted information per symbol. In general the transmission rate R is defined as the amount of information I per unit time interval T which can be denoted as $R = \left(\frac{I}{T}\right)$. This follows from the finite signalling rate which is defined as $\frac{1}{T}$ symbols per second. The maximum information rate achievable for a communication channel is defined as the channel capacity and can be expressed as

$$C = \max(R) = \frac{\max(I)}{T}, \left[\frac{\text{bits}}{\text{symbol}} \times \frac{\text{symbols}}{\text{second}}\right] = \frac{1}{T}\max(I), \left[\frac{\text{bit}}{\text{sec}}\right]. \qquad (1.4)$$

The channel capacity, C can be expressed through the bandwidth, B of a signal where the symbols are transmitted as pulses. As it has been shown in section 1.2.1 the maximum pulse transmission rate is determined as $\frac{1}{T} = 2B$. It is possible to apply some non-strict, reasoning in order to describe the fundamental relationship of channel capacity to bandwidth and signal-to-noise ratio, $\frac{S}{N}$. The complete proof of the well known Shannon's formula is rather complicated and requires substantial mathematical involvement. For the purpose of illustrating the major results of information theory for the case of transmission over noisy channels it is sufficient to make several assumptions which would allow expression (1.4) to be transformed so that it includes $\frac{S}{N}$.

The quantity of information $\max(I)$ can formally be represented as $\max(I) = \log(n)$. The number n can be viewed as a number of equiprobable symbols which can be represented as signal levels. These levels have to be separated at least by the amount of noise so that they are resolvable. If $S+N$ represents the total power corresponding to the amount of information after transmission and N represents the noise power, then n will correspond to the number of resolvable signal levels through the following relation, [Marshall, 1980]

$$n = \frac{\sqrt{S+N}}{\sqrt{N}}$$

This leads to the fundamental Shannon's formula relating the information capacity of a noisy channel to its bandwidth and signal-to-noise ratio:

$$C = 2B \log\left(\sqrt{\frac{S+N}{N}}\right) = \mathrm{B} \log\left(1 + \frac{S}{N}\right) \tag{1.5}$$

The interpretation of this result can be summarised as follows:

For any information transmission channel with capacity C it is possible to devise a code which allows for a source signal with an information rate of R bits/s to be transmitted over the channel at an arbitrarily low error probability, provided $R < C$ is fulfilled.

2. LINE CODING BASICS

The main principles of line coding are discussed in this chapter. The most essential objective in the development of line coding techniques is to achieve appropriate frequency distribution so that the characteristics of the line signal match as closely as possible those of the transmission channel. There are other important issues related to this type of coding, e. g. complexity of the coder circuit, methods of decoding and synchronisation, etc. Although some ideas concerning these topics are mentioned briefly throughout the presentation, it should be noted that most of the research work and the respective results revealed in subsequent sections concentrate on the construction and spectral analysis of various types of line codes.

This chapter serves as an introduction to the subject of line coding and is divided into two main parts. The first part is a general overview of many known types of line codes. Specific features and applications of various codes are given in the second part. In many publications outlining this area it is common to develop classifications of line codes. Some of these are mentioned below, although it is the author's opinion that a useful and sufficiently general classification is possible only after a set of appropriate criteria for assessment of line codes has been developed.

Before going into further details about the problems of line coding it is appropriate to summarise the description of the functional parts of the digital transmission system given in the previous chapter. This summary will help to define the general purpose and the specific role of line coding with respect to the other types of coding used in digital communications. The three major coding functions of the transmission side are given in Fig.2.1.

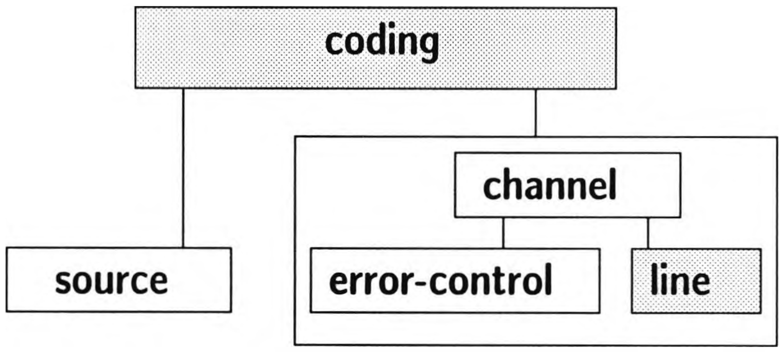


Fig.2.1 Coding for information transmission

There are two main design objectives which are common to all three types of coding: *efficiency* and *reliability.*

The first type, source coding, is required for efficient representation of information sources. The process of digitisation results in representing the source by a range of numbers. The source generates a sequence of symbols which correspond to these numbers. Usually the original sequence has to be transformed (coded) into a sequence of different symbols, suitable for transmission. Often the information measure associated with the coded symbols is higher than that of the source symbols. This redundancy is hard to avoid, but it is possible to be reduced. Source coding is applied to achieve this reduction, i.e. to increase efficiency of coding.

The second type is error-control coding. It is used to increase the reliability of transmission over a noisy channel. In digital communications the degrading effects of noise appear as errors. In this respect reliable transmission means having the capability to detect and correct errors. Coding for error control does not require specification of the parameters of the transmission media. It is concerned with the type of possible errors, their distribution and overall statistics. In summary: error-control coding ensures that a certain level of deterioration does not affect the original information content.

The third type, line coding, can improve on both – efficiency and reliability. While the first two types refer to the information content of signals, line coding is concerned with their frequency characteristics. Digital transmission is more efficient and reliable when the signal spectrum matches that of the channel. Line coding allows a suitable redistribution of frequency components to be achieved in order to minimise the degrading effects of the transmission media.

From the summaries given above it is clear that a common basis for the first two types is the Information Theory. However, it is quite appropriate to consider error-control and line coding as parts of a larger area which investigates the methods of countering the effects of channel deficiencies. This area is referred to as channel coding. Having identified the specifics and the interrelations between the types of coding, a definition of line coding is given below, which takes into consideration the objectives of coding in general [Cattermole, 1983].

Coding is the adaptation of digital signals to a transmission channel for the purpose of efficient and reliable communication

In order to specify this definition distinctively regarding line coding, the following statement is suggested: *Line codes convert digital sequences into coded line signals whose spectral characteristics provide for minimum distortion of the original signal by the restrictions of the transmission channel.*

2.1 Preliminary Notes

Historically there are many examples of actions in the direction of adapting the information source to the specifics of the transmission path [Ingram, 1981]. In a broad sense the very choice of appropriate signalling over some physical line can be considered line coding. However, this presentation is concerned with more contemporary communication systems which are based on pulse coded modulation type of digital signal transmission. In this respect it is useful to give an account of what are the practical objectives of line coding.

1) Frequency bandwidth:

Real transmission channels have a finite bandwidth. The effect of this limitation in general is that high frequencies are attenuated more than low frequencies. By means of line coding it is possible to redistribute the signal energy so that the significant part of it is contained in the lower range of frequencies with respect to the channel bandwidth.

2) Low frequency content:

Most often practical systems do not favour processing signals with large low-frequency components. Some of the reasons are the presence of a.c. coupling and significant capacitive and inductive components. Line codes are available to produce digital signals whose spectra have very small low (and/or high) frequency components without affecting the information content of the original signal.

3) Interference:

There are different kinds of interference.

a) Intersymbol interference can be dealt with by suitable pulse-waveform selection combined with special techniques of assigning the waveform to the original digital sequence. An example of this method is the partial response signalling.

b) Cross-channel interference is another type of transmission impairment. Signals with different spectra often share the same transmission channel. It is essential that frequency components of one signal do not appear in the bandwidth of another. Problems of this kind can be relaxed by the use of line codes which concentrate the frequency components in narrow spectra prior to distributing the signals over the channel bandwidth.

c) Electromagnetic interference is a general problem, which results in having unwanted frequency components within a particular range of the signal bandwidth. This problem can be solved by line coding which distributes the signal frequencies to parts of the bandwidth which are less affected by interference.

4) Bandwidth $\leftrightarrow$ SNR exchange:

A straightforward consideration shows that for a channel with a narrow bandwidth and high SNR a suitable multilevel line code could be applied to reduce the bandwidth of the original signal while utilising the low probability of errors caused by noise. Obviously the opposite transformation can be a valid line code which will make use of higher channel bandwidth to transmit a signal with fewer levels and thus reducing the degrading effects of a low SNR.

5) Timing content:

A very important point is the availability of information about the time interval of the code-symbols and/or the period of coded blocks of symbols. The knowledge of the duration of the pulse waveforms representing the code symbols is crucial for the correct recognition of the transmitted sequence. The process of acquiring this knowledge at the receiving side is called clock-synchronisation and on its precision depends the choice of best decision instants. These are the time instants where the received signal values are expected to be as close as possible to the values representing the original code symbols. Of similar importance is the recovery of the period of code words. Block synchronisation is essential for correct decoding of the received line signal.

The problems of the timing content of line-coded signals require special attention and have been well documented in the literature, [2,3,7]. It is sufficient to mention that the ability to extract the symbol repetition rate from the coded signal depends on the minimum number of transitions per unit time interval as well as the level of sophistication of the receiving equipment. Some aspects of spectral analysis related to the problems of synchronisation are addressed in an overview of typical line coding formats given in the next section.

To complete this preliminary part, the main requirements of the process of line coding are summarised below. Thus a properly designed line code should:

i) minimise vulnerability to interference and noise;
ii) enable extraction of timing information;
iii) introduce as little redundancy as possible.

It is possible to think that requirement ii) is a quite sufficient condition on the distribution of symbols in the line signal. Indeed, in order to have enough transitions as mentioned above, the code should provide for symbol variations in the coded sequence. However, the amount of transitions may be sufficient but the pattern of identical consecutive symbols to be inappropriate. The consequences of such a situation could be analysed through the low frequency content of the signal spectrum. One way to avoid problems of this kind is to add another requirement

iv) uniform distribution of the line coded symbols.

The last point to be mentioned here concerns requirements which bear no direct relation to the spectral shaping task of line coding. Sometimes the line signal is designed to provide additional information to support supplementary functions of the communication link and performance monitoring. These requirements are a consequence of two factors:

a) The line coder is part of a complex terminal equipment whose performance depends on ancillary channels for service and monitoring.
b) In many real systems specific frequency distribution is achieved with line codes which introduce high information redundancy, which can be utilised.

The factors outlined above allow for the possibility to require additional features of a line code, such as provision of supervisory and error monitoring channels.

2.2 Types of Digital Line Codes

Classification of line codes depends on the assessment criteria employed. One possibility is to distinguish various groups according to similarities in the coding rules. For example codes where a source symbol is represented by alternating code symbols in order to avoid long spells of constant-level line signal. Another classification divides line codes according to areas of application. Possible categories could be codes for metallic cables, radio links, satellite communications, fibre-optic lines. The classification which seems most appropriate for analytical purposes is based on the structure of the coded sequences. Finally it should be mentioned that for many categories of line coding schemes it is possible to introduce subclasses. For example the line codes of a particular class could be subdivided into binary and multilevel types.

In general it is a very difficult task to compile an exhaustive classification of the existing line codes. A major obstacle is the lack of a systematic approach in the practical design of line coding techniques. Most achievements in this area are predominantly application driven and the reported results are difficult to categorise. The spectral analysis technique based on the coder model developed by Cariolaro and Tronca, [19] and the design approach suggested in the thesis are an attempt to promote a unified method for construction and assessment of digital line codes. In this section various classes of codes will be described only to outline typical features and their relation to the spectral characteristics of the respective line signals.

2.2.1 Codes with Constraint on Disparity

It is known that for many applications (especially for metallic cable lines) transmission of unbalanced sequences of symbols[1] causes serious problems, like high levels of intersymbol interference. Unbalanced sequences are likely to contain long portions of consecutive identical symbols which also results in difficulties with the extraction of timing information. To overcome this problem methods for control of the digital sum of the coded sequences can be applied. The digital sum for binary sequences is defined as the difference between the number of ones and the number of zeros [Cattermole, 1983]. For a sequence of k symbols (where the zeros are represented as -1) this sum is given by

$$S_k = \sum_{i=1}^{k} a_i \tag{2.1}$$

for $a_i = +1$ or -1 and k is an arbitrary integer. The notion of digital sum can also be applied to blocks of symbols (words) and is usually called disparity.

It is possible to derive a general expression of the disparity for non-binary set of symbols. The interesting problem is to determine the number of words of equal disparity for a given word length. This can be done in various ways and an example of the results (taken from [17]) for binary words of length 5 to 12 is shown in Table 2.1.

Number of digits n	Number of words 2^n	Number of words with disparity 0	± 1	± 2
5	32		10	
6	64	20		15
7	128		35	
8	256	70		56
9	512		126	
10	1024	252		210
11	2048		462	
12	4096	924		792

Table 2.1 Numbers of binary words with a given disparity

[1] Unbalanced binary sequences exhibit portions where the number of ones and the number of zeros differ considerably.

By appropriate selection of a constraint on the variations of the digital sum or the disparity various codes can be constructed to solve the problems mentioned at the beginning of this section. A class of codes which have been widely implemented in digital communication system are those with zero mean disparity. Most of them are known as alternate codes a classical example being the AMI code[2]. A common strategy in implementing disparity constraints is to use a pair of code blocks with opposite disparities to substitute alternatively an input block. Typical representatives of this group are:

the binary nB(n+1)B codes, for n odd, e.g. 1B2B (CMI), 5B6B, 7B8B;
the ternary nBmT codes, e.g. AMI, 4B3T.

In general the effects of using coding schemes with low disparity or small digital sum variations result in modification of the signal spectrum which are favourable for most transmission channels. When the digital sum is bounded its long term average tends to zero and the DC component is eliminated. The same constraint limits the maximum number of consecutive identical digits which reduces the level of low frequency components and ensures the occurrence of transitions, i.e. the availability of timing information.

An important consequence of using disparity constraints, which is most relevant to the spectral analysis techniques presented in the thesis, is the possibility to use a finite state machine model of the line coder. By adopting limitations of the digital sum or the word disparity it is possible to view the coder as a system which operates according to a set of states, corresponding to particular values of disparity. At this stage it is possible to give a brief preliminary introduction to the line coder model adopted in the subsequent analysis. The idea of constructing codes, which restrict the variation of disparity related to symbol blocks of finite length, leads to the need of a device with a finite number of states. In other words the coder registers predefined values of the digital sum and moves into uniquely corresponding states. A coder is said to have more than one state if there is at least one input word which is transformed into two or more distinct output words. The essential features of a finite state machine model of such a coder can be specified as follows:

- the system has a finite number of possible states and operates over a finite number of input (and output) blocks;

[2] Alternate Mark Inversion. Detailed description of this code as well as most other line codes mentioned as typical representatives of different categories will be given in subsequent sections.

- the next state of the coder is determined from a combination of the present state and input.

- the output is determined from the current state and input of the coder;

The implementation of this model for spectral analysis of digital line codes requires more detailed consideration of some results from the probability theory and statistics of Markov chains. A complete description of the model is given in a separate section of Chapter 3.

2.2.1.1 Codes with Alphabetic Structure

The zero mean disparity codes are a subclass of a larger category known as alphabetic codes. These codes are based on the availability of output blocks whose number is larger than the number of input blocks. Two or more subsets of output words, called alphabets, correspond uniquely to different coder states. Each code alphabet is a unique mapping of all possible input words for a given state. In general an output word may belong to different alphabets. For example, a code block with zero disparity can correspond to the same input word for all coder states. At the same time code blocks with opposite disparities may belong to different alphabets and correspond to another input word. These are only some of the considerations in designing a particular line code. There are many examples of coding schemes with more than two alphabets which are typical for the ternary block codes. The latter employ three values for the output symbols, often ± 1 and 0. Typical examples of this group are MS43 and FOMOT, [25].

The considerations applied in constructing the code alphabets could be quite complicated and may include probabilistic constraints of the digital sum variations. In a few words, this means that the digital sum is allowed to attain larger values but the code ensures that the probability of this to happen is very low. Techniques involving probabilistic constraints are suitable for coding sequences with different probabilities of symbols and blocks of symbols. The 6B4T is an example of such code and some more considerations on the problems of probabilistic constraints are given in reference [17]. The same reference provides also an overview of two main categories based on areas of application, i.e. line codes for cable communications and radio links. The first category includes two distinct subgroups codes for metallic lines and codes for optical fibres.

2.2.1.2 Application Considerations

It has already been mentioned that a classification, based on structural features of the line codes is considered most appropriate for the purposes of this presentation. However, the typical arguments in forming categories application-wise are briefly presented below for the sake of completeness.

Metallic lines for digital transmission are divided in three types with respect to line code suitability: short, medium and long. The specific requirements on the parameters of the digital signals depend on the type of communication system. Short distance lines are typical for local networks carrying voice and data and reliability and simplicity have been often considered before efficiency. Coding schemes like AMI are very popular especially the improved alternating codes from the High Density Bipolar class (HDBn)[3], which remove long sequences of identical symbols. The HDB3 code, for example, causes predefined violations of the alternation rule to indicate four consecutive zeros. For long communication lines codes with higher level of efficiency are used. Such lines also require special attention to the utilisation of the channel bandwidth. High-capacity long-distance communication system usually employ binary or ternary coding schemes, like 3B4B, 6B4T and 10B7T. Codes with a number of symbol values higher than three are rarely considered suitable for cable lines mainly because of problems in intermediate regeneration of the signals.

It is more difficult to define special codes used in medium distance communications, but it will not be misleading to state that the considerations combine requirements from the other distance-types. HDB3 and 4B3T codes are not uncommon in medium range cable communications. Finally it should be mentioned that suitability of line codes depends on the type of metallic cable, i.e. copper pairs or coaxial cables. While only the latter are used in long-distance systems, both types are possible for short and medium lines.

The codes for optical fibre lines are mainly binary. This is due to the fact that light-intensity modulated signals are unipolar. There have been various approaches in designing line codes for optical fibres, [25]. In the early optical communication systems compatibility with existing digital communications has been very important. This has resulted in coding schemes converting ternary into two-level codes. Typical representatives of this group are the CMI and the two-level AMI. These codes exhibit many good properties but their efficiency is not always satisfactory. They have been used mainly in systems with excess of optical-fibre bandwidth and high reliability requirements. With the development of large

[3] n is the maximum number of consecutive zeros which are left unchanged.

capacity optical communication systems and the availability of fast components binary codes of much higher efficiency have been designed. They are generally known as nB$(n+1)$B codes. Usually two alphabets are constructed from the set of output words and the codes vary mainly in the constraint on the digital sum. Some codes with very good properties have found a wide spread application, typical examples being 5B6B and 7B8B. With respect to their spectral characteristics the nB$(n+1)$B codes have representatives which offer quite favourable line signal spectra. For example the two alphabet alternating types with small disparity values exhibit symmetrical frequency distribution with reduced low frequency content.

Most considerations mentioned above are mainly applicable to baseband digital transmission. Coding for carrier modulated communications is based on fairly different requirements, although, in principle it is possible to achieve acceptable results by direct transfer of a baseband signal upon a carrier through amplitude, frequency or phase modulation. However, there are problems and restrictions specific to carrier transmission and the related theory requires more involved discussion which is outside the scope of this work.

2.2.2 Other Types of Classification

It is possible to identify a group of line codes with features corresponding to particular system requirements. An example of such a group are the codes providing for error monitoring and ancillary channels. Codes with these features are very attractive for optical-fibre systems where additional service lines are not efficient. Although most manufacturers of optical cables offer metallic wires running alongside the optical fibres, it is much more efficient to provide a service channel integrated in the main digital signal.

In principle all coding schemes with some amount of redundancy have inherent error monitoring potential[4]. The simplest example is the the AMI code which provides for detection of errors violating the rule of mark alternation. However, the poor timing content of that code has made it less attractive for modern communication systems. More sophisticated error monitoring pattern is available from the HDB3 line code which has been designed to overcome the timing disadvantages of the AMI. This code has proven to be so successful in the early digital transmission systems that it has been included in the Recommendations of the CCITT (G.703).

[4] E.g., the number of valid code words is smaller than the total number of code blocks for a given set of code symbol values.

A further step in utilising the inherent redundancy of some line codes is the deliberate 'violation' of the coding rules at the transmitting side for the purpose of sending additional information (service channels). The pattern of occurrence of such violations is anticipatory to the receiving side and can be distinguished from channel errors. In order to illustrate some techniques for error monitoring and service channel insertion a few examples are described briefly below.

The 2-level AMI code, [30] has resulted from the adaptation of the original bipolar version to a binary signalling system. The coding for 3-level AMI and its transformation into a 2-level format are shown in Table 2.2. It should be noted that the transformation results in a coded sequence of binary digits at twice the original transmission rate.

Binary input	3-level bipolar AMI	Binary AMI
0	0 (preceded by 0)	10
	0 (preceded by 1)	01
1	+1 (if the last non-zero digit was −1)	11
	−1 (if the last non-zero digit was +1)	00

Table 2.2 The binary AMI code definition

It is not difficult to see that certain combinations of code symbols are invalid, e. g. 0000 and 1111. These two blocks could represent deliberate modifications of 0100 and 1011 respectively. If such changes are inserted at a predefined rate (lower than the information transmission rate and higher than the expected error rate), an additional channel could be established. Obviously the larger the redundancy of a line code the more freedom of choice in organising the ancillary channel. In the case of low-redundancy codes greater care should be exerted about possible changes in the spectral characteristics of the modified line signal.

The second example refers to codes with simple error detection capabilities. The 7B8B line code is a very popular scheme in this respect, although other binary block codes are also suitable for the purpose. One possibility to achieve a simple error monitoring structure is to control the code sequence parity. This can be done by assigning two code words with opposite disparity to a specially selected input word. In the process of transmission the digital sum is constantly evaluated. Every time the special input word appears the coder replaces that word with one of the two code blocks so that the digital sum always becomes even (or odd). Descriptions of in-service error monitoring schemes are given in [21,22].

The preceding sections are far from an exhaustive survey of the possible classifications of digital line codes. Only a few examples of the many possible codes have been given, mainly to illustrate the considerations in estimating the significant features of different coding schemes with respect to their particular category. As it was mentioned earlier, a complete classification of the existing line codes is not the prime goal of this chapter. It provides only the basis for developing the unified spectral analysis approach suggested in Chapter 4. To achieve this a number of known line codes are specified in the next section without introducing any particular order.

2.3 Specific Features of Typical Line Codes

The main reason for having this section is to provide a broad basis for comparing the conventional techniques of code definition to the generalised method presented in Chapter 4. The theoretical basis for this method was first suggested by Cariolaro and Tronca, [19] and part of this thesis is an attempt to extend the supporting arguments over a larger number of practical cases. Most of the line codes given below are defined in a way, commonly used in the specialised literature, i.e. by compiling a descriptive list of coder rules. There are certain advantages in such definitions, one of them being the ability to suggest a possible coder design, as well as any special features of the coded sequence. However, an obvious disadvantage is the lack of uniformity in stating the code rules, which has two main consequences:

a) Similarities between definitions are sometimes difficult to estimate (this may result in 'invention' of existing codes).
b) Systematic classification cannot be developed due to great diversity of coder specifications.

It should be noted that the presentation is restricted to the problems of baseband transmission, the practical examples, which are given below and further analysed in subsequent chapters, involve binary and ternary coding only. These are the coding schemes in predominant use for non-carrier systems and the examples chosen to illustrate the results of this work convey sufficiently well the ideas of the unified approach in spectral analysis of digital line codes. In order to preserve some consistency in the following description of line coding schemes common terminology and, where possible, notation is adopted. This is based on the following major assumption which does not restrict the general validity and applicability of the results.

Coding always refers to substitution of symbols or blocks of symbols at the input of the coder with output symbols or blocks of symbols.

With respect to the remark made above, binary symbols are represented as 0 and 1 or −1 and +1, while ternary symbols are taken from the set −1, 0, +1.

Another common convention, which is often used in the description of line codes, is to refer to the transformation of input signal levels into a combination of output signal levels. Thus the binary set consists of a low level (corresponds symbols to 0 or −1) and a high level (1 or +1) respectively. The only order to be followed in the descriptions below is from most basic and simple codes towards schemes of higher complexity.

2.3.1 'No Coding' or Basic Symbol Transformations

It is quite legitimate to include this case before any other code description as, clearly, a line signal is produced by the transmitting side of any digital communication system, regardless of whether a specific coding procedure is involved. Many authors prefer to present line coding as an integral part of the modulation process [Peebles, 1987; Killen, 1988]. This is due to the gradual evolution of line coding from the problems of digital signal modulation and eventually explains why in the literature line coding is often described as waveform formatting of digital signals.

Such an approach is based on the assumption that the line signal is formed by selection of particular pulse waveforms and then generating these waveforms to represent the symbols of a digital sequence. Sometimes combinations of waveforms (and periods of 'no-waveforms') are used to achieve specific properties of the line signal (mainly spectral characteristics suitable to the transmission media). The complete frequency distribution of the transmitted signal is considered predefined by the spectrum of the individual pulse waveform which is additionally modified by possible combinations of waveforms. This method of line signal analysis is applicable only in cases with relatively simple rules of substitution of a sequence of digits with a combination of waveforms. It is far less practical when blocks of two or more input symbols are to be represented with a combination of pulse waveforms and certainly inefficient in evaluating the statistical properties of line codes which are essential in achieving special features of the signal spectra prior the process of modulation.

In most cases, when 'signal-encoding formats' are described, it is assumed that the output signal is formed by a sequence of rectangular pulses. This assumption allows for relatively simple illustration of the resulting line signal as a time function and a degree of uniformity in deriving the expressions for the spectral density distribution. In summary, when line coding is presented as a part of the digital modulation process the various code structures are described as the transformation of a sequence of digits into a sequence of pulses (usually rectangular time-waveforms). This implies the use of a common terminology of signal levels and transitions between levels.

At this point it is possible to indicate the first significant concept of the unified approach in spectral analysis of digital line codes developed in the thesis. It is appropriate to represent the line coder as a separate system, transforming a sequence of input symbols into a sequence of output symbols[5], in order to analyse and design particular modifications of the line signal spectrum, which are accomplished through the specific patterns and combinations of waveforms, regardless of their shape. This representation, as it will be shown later, allows for complete evaluation of the statistics of the coded sequence, and therefore its spectral density distribution. It should be noted, however, that second order statistics are sufficient only in the case of subsequent symbol-by-symbol modulation with the line signal waveform. With more complex modulation formats higher order statistics are required [Cariolaro, 1983].

A signalling technique, known as partial response provides a specific example with respect to the above considerations. It has been mentioned already that the problems of intersymbol interference are reduced with particular pulse waveforms (e.g. a raised-cosine pulse). By using combinations of such pulses corresponding to symbols from a digital sequence and delayed versions of that sequence it is possible to achieve particular shaping of the line signal spectrum. This type of processing is referred to as partial response, because the resulting pulses occupy two adjacent symbol intervals. The frequency distribution, achieved through such a process, is attributed to the modulating technique following the usual way of computing the spectrum from the time function of the signal waveform. However it is possible to show that the special features of the signal spectrum are acquired through the type of combination of the symbols in the digital sequence regardless of the modulating pulse waveform which modifies that spectrum additionally and could be designed to accomplish other purposes like intersymbol interference. The analysis results for the Duobinary types of coding, given in Chapter 4, provide further evidence in support of this statement.

[5] Although in general the meaning of the symbols is immaterial to the coding process, they are usually represented as digits, which, together with a set of operations, are considered to belong to some number field, in order to facilitate the strict mathematical analysis.

Finally it can be seen why certain types of line signalling are regarded as codes without actually being a result of any encoding procedure. Clearly the reason for this is the presentation of the line coding process as an integral part of the modulation function, which is commonly referred to as waveform formatting. At this point it is possible to start the list of conventional definitions with the basic types 'encoding formats'. Some of the brief descriptions given below indicate alternative ways of code rules definition to facilitate the presentation of the encoder separated from the digital modulator.

NRZ (unipolar)

The nonreturn-to-zero line signal formatting is the straight forward transformation of a sequence of binary symbols into a pulse waveform which is ideally represented by a constant non-zero level for the time interval of a symbol one and zero level line signal for the period of in input symbol of zero. If this is viewed as the process of modulation only, then it can be described as modulating the 1-s from a coded 0/1 sequence with a rectangular pulse waveform which does not return to zero level for the whole time interval of a code symbol 1. Obviously the coded sequence in this case is the original digital sequence itself, therefore the line coder (which transfers symbols into symbols) can be presented as a fictitious system 'converting' 0 to 0 and 1 to 1. In the pattern of the forthcoming definitions the NRZ line code can formally be described as follows.

Coding rules: The input zero (space) is coded as output zero and the input symbol of one (mark) is coded as output one.

Main features: As this code or rather line signal format ***does not change the original information sequence***, it can be considered the basis for comparison of of all other coding schemes. The ***absence of any information redundancy*** means that ***the spectrum of the NRZ signal can be taken as a basis*** in the estimation of minimum channel capacity as well as channel bandwidth and transmission rate. The advantage of the NRZ signalling is the ultimate ***simplicity and efficiency.*** However, the ***presence of d.c. and a high level of low frequency components plus the small amount of timing information*** with long sequences of identical symbols, comprise the disadvantages which sometimes are quite unacceptable.

When the output symbols are +1 and −1, corresponding to positive and negative signal levels respectively, the NRZ is called a polar waveform. This type of signalling has the same features as the unipolar NRZ except for slightly higher noise resistance (under certain conditions).

Some modifications of the basic NRZ format exhibit a slightly higher level of coding complexity[6]. Although these types of line signalling do not contribute substantially to the variety of coding formats, the main reason for their inclusion as separate definitions is to illustrate the suggested new approach in presenting the transformation of a sequence of symbols into a coded sequence.

NRZ-M/NRZ-S

The NRZ-Mark and the NRZ-Space, known also as 'differential' formats, convert the original digital sequence into a sequence with a different structure without changing the information content of the signal. This operation is best illustrated by showing that it is equivalent in principle to symbol-by-symbol inversion of the digits in a suitably defined binary sequence. To avoid unnecessary wording the symbols of the original sequence are taken in pairs and each pair is substituted with a different one according to the respective rules.

Coding:	NRZ-M	NRZ-S
	00 and 10 become 00 or 11	00 and 10 become 01 or 10
	01 and 11 become 01 or 10	01 and 11 become 00 or 11

Main features: If long spells of identical digits are expected to appear regularly in the original sequence, these codes ***provide for increase in the amount of transitions, i.e. improvement of the timing content*** can be achieved.

It is important to note that, when the input symbols are of equal probability, nothing is gained from the use of NRZ-M and NRZ-S.

It is now possible to see that if the combinations 00/10 and 01/11 (for NRZ-M) are viewed as some composite symbols, then the combinations 00/11 and 01/10 can be considered to represent the inverse of the former. Obviously the effect is the same as the normal inversion of the original binary sequence. (Similar reasoning applies for NRZ-S). However, the difference is that the two codes change long sequences of consecutive 1-s or 0-s, respectively into sequences with transitions. This difference is a potential advantage. The reason for the remark,

[6] This term is discussed in detail in Chapter 4.

following the feature specification of the above line codes, becomes apparent by pointing out that for equiprobable input symbols the average density of transitions is the same as the density of consecutive identical symbols. The advantage of applying the two types of NRZ signalling becomes significant when the input symbol probabilities are different.

The software routine (presented in Chapter 3), which has been developed to provide parametrical computation of the spectral density function, offers significant power and precision in evaluating the resulting frequency distribution for various symbol probabilities. More detailed comment on these matters is given in Chapter 4.

Before closing this section one more 'no-coding' scheme is described mainly for completeness and also to affirm the argument in favour of replacing the conventional 'waveform formatting' approach with 'encoding of digital sequences' followed by modulation for the purpose of developing a unified evaluation technique in spectral analysis of digital line codes.

RZ

The return-to-zero signalling format is an example of a modulation procedure ensuring the presence of transitions in all symbol intervals corresponding to input one.

Coding rules: The input zero is transformed into a symbol period of zero level (no pulse). The input one is represented by a pulse waveform (high signal level) for half of the symbol period and zero level for the other half of the period[7].

Main features: *Good timing information* provided for high density of symbol 1 in the original sequence. Requires ***twice the transmission bandwidth of the NRZ format.***

It is obvious that the RZ wave formatting can also be represented as a modulator which generates a pulse waveform (of duration half the input symbol period) for a code symbol 1 and no-pulse for a code symbol 0 plus a coder which transforms an input 0 into 00 and an input 1 into 10. The inefficiency of this type of signalling in terms of line coding will certainly become evident through the subsequent analysis of other coding techniques, based on the same input/output

[7] Different proportions are also possible.

symbol sets. The presentation of conventional code definitions continuous in the next section with more complex code structures. No explicit notification will be made where alternative code rule descriptions are given to suggest a coding scheme independent from the modulation format.

2.3.2 Coding – Advanced Symbol Transformations

The basic coding formats from the previous section have a very restricted effect on reshaping the signal spectra without considerable increase of the bandwidth. Most of them achieve very specific purposes, often under certain probabilistic conditions only. More significant results in adapting the signals to the transmission channel can be accomplished through line coding schemes with higher level of sophistication. Binary and ternary block codes are used predominantly in digital systems for baseband transmission. Some of the most popular versions employed in modern communication systems are described below.

AMI

The Alternate Mark Inversion is a frequently used line coding scheme. It is also known as 'bipolar format' or 'pseudoternary' code.

Coding rules: The input symbol of zero is coded always as output zero. The input symbol of one is coded alternatively as +1 and −1.

Main features: ***No d.c. component. Errors can be detected*** from violations of the alternation rule. ***No timing information*** in long sequences of consecutive zeros. ***Low transmission efficiency*** (≈ 0.6)

CMI

The Coded Mark Inversion (also known as biphase-space) results in a two level format. It has been designed to improve on the timing contents and to avoid the necessity of three signalling levels.

Coding rules: The input zero is always coded as output low and high level, each of duration half the period of the input symbols. This can also be denoted as $0 \rightarrow 01$. The input one is coded as a low level (00), if the previous one was coded as a high level (11) and vice versa.

Main features: ***The amount of transitions is high*** (at least one per input symbol interval). The two levels are ***suitable for both polar and unipolar transmission***. Requires ***twice the bandwidth of the input sequence***.

Manchester

This code is also called biphase-level. Together with CMI and several other modifications, the Manchester code is a very popular representative of a group known as biphase coding formats.

Coding rules: The input zero is always represented as output low and high levels for the first and the second half of the input symbol period respectively, i.e. $0 \rightarrow 01$. The input one is represented by the same combination in reverse order, i.e. $1 \rightarrow 10$.

Main features: *Sufficient timing information* is available due to the presence of transitions for every input symbol interval. The *doubled bandwidth* is a disadvantage, common to the whole group of biphase coding formats.

A common characteristic of the biphase coding schemes is the substitution of input zero or input one or both with two different output symbols. The names of the different versions are suggestive of the type of transformation. Thus the biphase-space (CMI) converts every space (0) into 01, while the marks (1) retain a constant value during the input symbol period (i.e. 00 or 11 alternatively). The biphase-mark is the inverse of the CMI code and will not be described separately. The meaning of biphase-level for the Manchester code can be explained by noting that both input levels (0 and 1) are replaced with a pair of different symbols, 01 and 10 respectively.

Differential Manchester

Unlike the Manchester code, which creates two transitions for every input symbol in long uniform sequences while removing the transitions in the original sequence, the Differential Manchester introduces two transitions for one of the input symbols and one transition for the other input symbol.

Coding rules: The input zero is coded with two different symbols, the first of which is always different from the previous symbol. The input one is also coded with two different symbols, but the first of them is always the same as the preceding one. The rule can be given alternatively as follows

$$0 \rightarrow \begin{cases} 01 \text{ if the last code digit is } 1 \\ 10 \text{ if the last code digit is } 0 \end{cases} ; 1 \rightarrow \begin{cases} 01 \text{ if the last code digit is } 0 \\ 10 \text{ if the last code digit is } 1 \end{cases}$$

Main features: The advantage of this code is the same as for the whole Manchester (biphase) group, i.e. ***abundance of timing information***, due to the availability of at least one transition in every input symbol interval. The disadvantage is again the requirement for ***higher transmission bandwidth.***

The Differential Manchester code derives from the biphase family in exactly the same manner as the differential techniques (NRZ-M/NRZ-S) derive from the basic signalling format, NRZ.

Miller

The Miller code is also called delay modulation, the reason for which becomes apparent when the symbol transformation is described in the following manner: The input sequence is converted into transitions between two levels. A transition for input one comes half a period after the start of the symbol interval. A transition for input zero comes after a full symbol period, provided the next input digit is also a zero.

Coding rules: Input one is always coded as two different output symbols of duration half the input-symbol time interval, while an input zero is coded as output one or zero of length equal to the input symbol period, so that it is the same as the last code digit, if it represents an input one and opposite to the last code digit, if it represents an input zero. The alternative definition can be given as follows

$$0 \rightarrow \begin{cases} 00 \text{ if the last two code digits are } 10 \text{ or } 11 \\ 11 \text{ if the last two code digits are } 00 \text{ or } 01 \end{cases} ;$$

$$1 \rightarrow \begin{cases} 01 \text{ if the last two code digits are } 00 \text{ or } 10 \\ 10 \text{ if the last two code digits are } 01 \text{ or } 11 \end{cases} ;$$

Main features: The Miller code produces a sequence with ***very small low-frequency content*** and a ***possibility of signalling within the bandwidth of the original digital sequence.***

All coding schemes presented so far have more or less straight forward definitions in the sense that no algebraic operations are required to describe the coder rules. It should be pointed out, however, that virtually every coding operation can be presented formally as an expression which determines the algebraic relation between the input sequence and the coded sequence of symbols. Some good examples of such presentations in the terms of waveform formatting can be found in [Peebles, Jr., 1987]. In the simplest case of NRZ-mark (differential) encoding format the output sequence b_k can be produced from the input sequence a_k through modulo 2 addition as shown below

$$b_k = b_{k-1} \oplus a_k \; (k = 0, \pm 1, \pm 2, \ldots) \tag{2.2}$$

where b_{k-1} is the preceding output symbol. With some line codes the algebraic presentation provides better understanding of the transformation procedure and often a scope for further development and improvement of the underlying coding structure. The two coding schemes described below are often given as a good illustration of simple algebraic definitions.

Duobinary

This type of line waveform formatting is another example of a ternary code. The input binary symbols are converted into a three-level signal similar to the AMI format with one very essential difference - the positive and the negative symbols correspond to the input ones and zeros respectively, while the middle level (output zero) represents both. A more accurate specification is given below.

Coding rules: A binary zero at the input is changed into a zero or a middle level if the preceding input symbol was a one and into a low level if the preceding input symbol was a zero. The inverse applies for a binary one, i.e. it is coded as a zero or a middle level if the preceding input symbol was a zero and into a high level if the preceding input symbol was a one. Alternatively

$$0 \rightarrow \begin{cases} 0 \text{ to code the second digit of consecutive 10 at the input} \\ -1 \text{ to code the second digit of consecutive 00 at the input} \end{cases}$$

$$1 \rightarrow \begin{cases} 0 \text{ to code the second digit of consecutive 01 at the input} \\ +1 \text{ to code the second digit of consecutive 11 at the input} \end{cases}$$

Main features: The output symbols are of the same duration as the input symbols. The resulting line signal requires only ***half the bandwidth of the original binary sequence.*** The ***efficiency in terms of*** *information transmission rate is not optimal. Low frequency components are not reduced* and a sequence of alternating input symbols causes a constant zero (middle) level at the output with *no timing information.* Typical disadvantage of this type of coding is the *propagation of errors* and the need of precoding to avoid it.

Modified Duobinary

This code is an improved version of the formatting technique described above. It employs the same principle as the Duobinary code, except that each code digit depends on the input symbol preceding the last one. More strictly this can be stated as follows.

Coding rules: The output symbol is a zero or a middle level when the input digit is the same as the input digit before the previous one. The output symbol is a low level (−1) or a high level (+1) when the input is a binary 0 or 1, respectively and is different from the input digit before the previous one. Alternatively

$$0 \rightarrow \begin{cases} 0 \text{ to code the third digit of consecutive 000 or 010 at the input} \\ -1 \text{ to code the third digit of consecutive 100 or 110 at the input} \end{cases}$$

$$1 \rightarrow \begin{cases} 0 \text{ to code the third digit of consecutive 101 or 111 at the input} \\ +1 \text{ to code the third digit of consecutive 001 or 011 at the input} \end{cases}$$

Main features: The Modified Duobinary code exhibits *no d.c. and small low frequency components*, while preserving the *narrow bandwidth* requirement of the Duobinary. *Propagation of transmission errors* can be countered with special precoding.

The rules of the last two coding schemes can be specified more concisely through algebraic expressions relating the sequence of input symbols a_k to the output sequence of symbols b_k.

$$b_k = a_k + a_{k-1} - 1 \text{ (for Duobinary)} \tag{2.3a}$$

$$b_k = a_k - a_{k-2} \text{ (for Modified Duobinary).} \tag{2.3b}$$

It is important to note the gradual increase in complexity in the line code definitions given so far. Obviously the coder rules become more tedious to specify and difficult to understand when their implementation requires memory. In spite of the relative simplicity of the algebraic relations producing the code symbols, the dependence of the coder output on preceding input and output symbols involves evaluation of many combinations, whose number increases exponentially with the size of memory required. It should also be noted that in the cases of alternative definitions of the coder rules it is much clearer to state the transformations of symbols into symbols than to describe the resulting line waveform in terms of signal levels. A few more examples are given below to complete this section with code definitions which are closest to the form suggested for the unified approach of the general spectral analysis model. These are examples of alphabetic codes, mentioned earlier as a particular type, also known as block codes.

All coding schemes with a block structure are defined as transformation of blocks (or words) of n input symbols into blocks of l output symbols. The set of input symbols and that of the output symbols may be the same or different and in the case of both being the binary digits 0 and 1, the block codes are generally denoted as nBlB. Another more or less conventional notation refers to coding binary words into blocks of ternary digits $(-1, 0, +1)$, which is usually denoted as nBlT. There two important points to be made about codes with a block structure in order to specify the boundaries of applicability of the forthcoming arguments.

1) The numbers n and l can be either variable or fixed. The theory of variable length block coding is rather complicated and requires special treatment. The spectral analysis model adopted in this presentation and the results are related to fixed-length block codes.

2) The line coding schemes described so far are predominantly of symbol-by-symbol type. However it is clear that the latter can be represented as 1BlB or 1BlT block coding which may be viewed as the first step towards the unified method of line-coder definition.

In general the methods of line coding are not restricted to two or three level signalling. In case of multi-level line signal formatting the corresponding coding scheme is often referred to as m-ary, where m is the number of signal levels. It is possible to show that for a fixed rate of information transmission binary and multi-level transmission are equivalent if the channel is noiseless and not band-limited. However, for real transmission conditions binary signalling is more suitable to noisy environment with unrestricted bandwidth while m-ary line formats are preferable for low symbol transmission rates in channels with low levels of noise.

Two typical examples of block codes are considered below illustrating the binary and the ternary types respectively.

3B4B

This code transforms blocks of three binary digits into blocks of four binary digits. The output words are selected from the complete set of 16 possible combinations of four binary digits in such a way that the digital sum is limited and the average numbers of zeros and ones are equal.

Coding rules: The four words - 0000, 0011, 1100 and 1111 - are not used and provide for an error monitoring condition. Four of the remaining 12 words - 0101, 0110,1001 and 1010 - have equal number of zeros and ones. They are used to represent uniquely four input words. The other 8 output words are divided into two sets as shown in the table below

input words	output words		
0 0 0		0 1 0 1	
0 0 1		0 1 1 0	
0 1 0	0 0 0 1		1 0 1 1
0 1 1	0 0 1 0		0 1 1 1
1 0 0	0 1 0 0		1 1 1 0
1 0 1	1 0 0 0		1 1 0 1
1 1 0		1 0 0 1	
1 1 1		1 0 1 0	

Table 2.3 The 3B4B code definition

The coder selects an output word from the two sets alternatively to transform the respective input words.

Main features: The 3B4B line code is ***balanced***, i.e. the digital sum is limited and the average disparity is zero. ***The low-frequency components of the signal spectrum are small.*** The code ***efficiency is moderate*** (0.75).

The next description is one of the many versions of the 4B3T line code. It transforms the input binary words into a set of ternary words.

MS43

The use of three levels in this code allows for reduction in the signalling rate by transmission of 3 code symbols for every 4 input symbols. There are three code-word sets designed to ensure a good balance as well as disparity constraints.

Coding rules: Every four-digit input binary word is substituted with a three-digit ternary word from the table given below, according to the accumulated disparity.

input words	output words used with disparity −2	−1 or 0	+1
0 0 0 0	+1 +1 +1	−1 +1 −1	−1 +1 −1
0 0 0 1	+1 +1 0	0 0 −1	0 0 −1
0 0 1 0	+1 0 +1	0 −1 0	0 −1 0
0 0 1 1	0 −1 +1	0 −1 +1	0 −1 +1
0 1 0 0	0 +1 +1	−1 0 0	−1 0 0
0 1 0 1	−1 0 +1	−1 0 +1	−1 0 +1
0 1 1 0	−1 +1 0	−1 +1 0	−1 +1 0
0 1 1 1	−1 +1 +1	−1 +1 +1	−1 −1 +1
1 0 0 0	+1 −1 +1	+1 −1 +1	−1 −1 −1
1 0 0 1	0 0 +1	0 0 +1	−1 −1 0
1 0 1 0	0 +1 0	0 +1 0	−1 0 −1
1 0 1 1	0 +1 −1	0 +1 −1	0 +1 −1
1 1 0 0	+1 0 0	+1 0 0	0 −1 −1
1 1 0 1	+1 0 −1	+1 0 −1	+1 0 −1
1 1 1 0	+1 −1 0	+1 −1 0	+1 −1 0
1 1 1 1	+1 +1 −1	+1 −1 −1	+1 −1 −1

Table 2.4 The MS43 code definition

Main features: The MS43 line code is ***well balanced*** and the restricted digital sum provides for a ***small low-frequency content.*** The need for a ***look-up table*** requires ***higher coder complexity.***

Most practical line coding schemes require $l > n$ for a binary block structure nBlB and usually $l < n$ for ternary block codes. This may suggest that the prime reason for implementing an nBlT code is to reduce transmission rate and possibly to achieve a particular frequency distribution. While this is so in many cases, the fact that a large number of 1B1T codes exists (some have been described above) indicates that initially three-level signalling has been introduced mainly to achieve a particular spectral distribution and timing content.

Most of the known codes, as well as many newly developed ones, are specified by the usual descriptive definition of the coder rules. Unfortunately this approach has been adopted even by some influential standardising organisations. Typical examples are the code definitions of HDB3 and CMI given in Recommendation G.703 of CCITT. The descriptive specifications of line codes given in this section indicate some weaknesses of this type of presentation. The definition of code rules becomes too complicated with the increase of the number of possible coder states. Sometimes similarities as well as differences are disguised by particular wordings which could prevent the derivation of more general methods and schemes. It has also been indicated that, if the modulating process is separated from the line coding, the latter can be defined in a more uniform manner as transformations of symbols instead of waveform formatting. Thus it becomes possible to perform spectral analysis based on the structure of the coded sequence regardless of the characteristics of the pulse waveform used for transmission. Such an approach provides much greater flexibility in achieving special features of line signal spectra and considerable freedom in developing new coding schemes.

Various examples of coding techniques have been given in this section to illustrate the conventional descriptive definition of coder rules. Some possibilities for alternative concise and uniform definitions have been suggested. The line coder presentation as a system transforming input symbols into a coded sequence of output symbols is further developed in the next chapter. A complete statistical model of the coding process is introduced after a brief overview of the spectral analysis concepts. Based on this model an unified approach to code definition is fully developed in Chapter 4 and some of the coding formats described above are redefined through their table and matrix presentation.

3. THE LINE CODER MODEL – THEORY AND IMPLEMENTATION

The general spectral analysis model is defined in this chapter. It has been developed by G. L. Cariolaro et. al. and published in the Int. J. Electronics, 1983. The model is based on the concept of representing a digital line coder as separate systems. This allows the evaluation of the complete statistics of the coded signal independently of the line waveform modulator. Most line codes can be specified as transformations of a sequence of symbols into a coded sequence of symbols. The adaptation of the line signal to the transmission channel is achieved mainly through appropriate shaping of the spectrum of the coded digital sequence which requires adequate frequency analysis of the latter. In order to evaluate the spectral density of the symbol sequence at the output of the coder, it is necessary to determine the statistics of this sequence. The adopted line coder model provides a general solution to this problem for all codes which transform blocks of digits into blocks of code-symbols[1]. It allows the statistics of the output digital sequence to be evaluated from the statistics of the input sequence (which is assumed to be stationary and memoryless) by defining the process of transformation through the transitional probabilities of a finite-state system.

By assuming some conventional linear digital modulator (e.g. one which produces a rectangular pulse waveform for every nonzero code symbol) the line coder model can be used to specify uniquely any fixed-length block code. This allows accurate comparative assessment of various coding schemes together with complete and detailed spectral characterisation of the coded line signal. The analysis results can provide any required degree of numerical precision[2] as well as highly informative graphic output. The latter can be appreciated through the analysis examples presented in Chapter 4. Finally, the possibility of a systematic assessment of the characteristics of line coding schemes and the generality of the method, have created the basis for some constructive feedback, which has allowed a new classification to be suggested and an unrestricted variety of new codes to be generated.

This chapter begins with a brief introduction to the problems of spectral analysis in digital line coding. This is followed by a detailed definition of the general line coder model which leads to the analytical expressions for the frequency distribution used in the computational procedure. The latter has been implemented in a software routine whose structure is outlined in Section 3.3.

[1] Symbol-by-symbol coding can be viewed as transformation of blocks of length one.

[2] The only real limitations are the number of variables and the sizes of mathematical structures the package can handle.

3.1 Spectral Analysis – General Considerations

How important is it to analyse the spectra of digital line signals and what degree of precision should be pursued, are questions whose answers determine the degree of involvement and the capabilities of various methods. The types of signals transmitted over digital communication lines can be divided in two main categories: signals which require error-free transmission and signals which are considered acceptable above a certain level of deterioration. In the first case error-correcting techniques are applied and usually transmission systems operate with considerable margins with respect to channel capacity to ensure that errors do not exceed the number which a code is capable to correct. The second category of signals allow for a certain amount of errors to be considered a tolerable impairment of the received information (voice, video, etc.). Most practical digital systems, however, operate at very low error rates (far below one in a million) and very close to the capacity limit of the transmission channel. (The requirement for efficiency is often imperative.) The second condition implies that signal spectra should match almost perfectly the frequency characteristics of the transmission media. This means that even small deviation from the required spectral density distribution can result in an unacceptable increase in the amount of errors.

Therefore, to be able to perform detailed and accurate spectral analysis of digital line codes is considered of crucial importance to modern communications, where the degree of efficiency and precision are constantly pushed closer and closer to their limits. It is also important to create powerful design techniques that can combine a sufficient productivity in the development of new solutions with the ability to make an uniform assessment of the analysis results. The results presented in subsequent sections demonstrate a possible approach in response to the above requirements.

The spectral density function of deterministic signals can easily be found through the Fourier series theory. The frequency analysis of truly random signals, by evaluating the autocorrelation function is also more or less straightforward. The problems become complicated in the spectral analysis of signals which are not deterministic, but not completely random either. When a system transforms a random sequence of digits, so that the output sequence depends on various states of that system, the result is a signal which has some added structure, i.e. the symbols of the output sequence are no longer independent. In order to evaluate the spectrum of the transformed sequence it is possible to use the autocorrelation function which can be derived from the statistics of that sequence. However, the problem is to determine how the statistics of the input sequence have been modified in the process of transformation. Clearly the changes are functionally related to the sequence of transitions between the states of the system. Such sequences can be described most adequately by the theory of Markov chains through their transitional probabilities [Winsten, 1981; O'Reilly, 1984] and in many practical cases of interest this is sufficient to evaluate the complete statistics of the coded sequence.

The spectral analysis in digital transmission is based on the second order statistics of the respective signals. This can be illustrated with a general introductory definition of the fundamental relationship between the time and the frequency functions representing a random signal. The autocorrelation of some sequence $a(t)$ is given by

$$R_a(\tau) = E\langle a(t)\ a(t+\tau)\rangle$$

where $a(t+\tau)$ is a delayed version of that sequence. The spectral density of $a(t)$ is defined as the Fourier transform of its autocorrelation as follows

$$S_a(f) = \int_{-\infty}^{+\infty} R_a(\tau)\ exp(-j2\pi f\tau)\ d\tau$$

Various techniques exist for evaluation of the frequency distribution of coded digital signals. Most elaborate theoretical treatment of the related problems can be found in [Cattermole and O'Reilly, 1984] and [Tröndle and Söder, 1987].

The complete definition of the model is presented in the next section by adopting the conventional discrete signal notation $x(kT)$. Some of the algebraic transformations, which are not given in the original work, have been made independently to trace the key steps in deriving the final expressions used in the software computational routine.

3.2. Description of the Line Coder Model

The spectral analysis model, described in the following subsections, is presented in three main stages. First the components comprising the model are appropriately defined. The second stage involves specification of a finite-state system which describes the core of a coder, transforming input symbols into code symbols according to a set of possible states. A suitable definition of the input/output variables and functional relations of the finite-state system allows to evaluate the complete statistics of the coded signal from the input symbol probabilities. Finally the analytical expressions for the spectral density distribution are derived from the second order statistics of the coded sequence which are evaluated through appropriately defined autocorrelation function. The theoretical basis for spectral analysis in linear and non-linear systems is briefly introduced in order to determine the frequency characteristics of the line signal by using the input/output relations for some systems of the model.

An arbitrary line coder can be described through the systems and the signals shown in Fig.3.1, which can be viewed as components of the model.

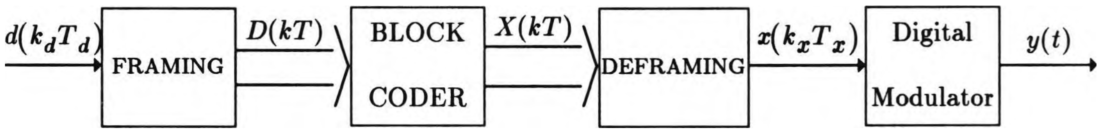


Fig.3.1 Components of the line coder model

The two types of components are defined in their general form as follows.

Signals There are two categories of signals — deterministic and random. A mapping of the time-domain into a set of values is considered a deterministic signal. A collection of such mappings is assumed to represent a random signal where every mapping is a realisation of the random signal.

The present analysis is concerned mainly with signals described as random sequences of symbols. Two types of signals represent synchronous transmission — continuous-time signals and discrete-time signals. In the first case both the time-domain and the signal values are some continuous subset of the real numbers and this is usually denoted as $y(t)$. In the second case the time-domain consists of a discrete set of values which are mapped into the set of discrete signal values. For equally spaced time-domain values $\{0, \pm T, \pm 2\mathrm{T}, \dots\}$ the signal can be denoted as $d(kT)$, where $k = 0, \pm 1, \pm 2, \dots$. Here T is the period of time between two consecutive signal values and $F = \frac{1}{T}$ is the signal rate.

Systems These are components of the model which can be represented by their input/output relation and possibly a set of states. Two types of systems are used in the representation of the coder – linear and nonlinear.

The spectral analysis theory for liner systems allows the frequency characteristics of the output signal to be determined directly from those of the input signal through the relation between the respective correlation functions. For nonlinear systems the evaluation of the spectral density of the output signals requires the second order statistics to be estimated. The block coder is a nonlinear system and so is in general the digital modulator. For the purposes of the present analysis it has been assumed that the line signal is produced by a linear function, such as Pulse Amplitude Modulation (PAM), which is implemented in many real systems.

3.2.1. Definition of the Components in the Line-Coder Model

In the discrete time-domain signals are often referred to as sequences of symbols or sequences of words (blocks of symbols). The signals in the line coder model can now be defined as follows:

$d(k_dT_d)$ is the input-symbol sequence where $k_d = 0, \pm 1, \pm 2, \ldots$. The period of the input symbols is T_d and they are selected from the set of possible values (alphabet) denoted as $d = \{d_1, d_2, \ldots, d_\alpha\}$.

$D(kT) = \left[d_1(kT), d_2(kT), \ldots, d_N(kT)\right]$

is the input-word sequence where $k = 0, \pm 1, \pm 2, \ldots$. The input word is a row vector with elements $d_n(kT)$, $n = 1, \ldots, N$ and period $T = NT_d$. $D(kT)$ is selected from the set of possible blocks of symbols (dictionary) denoted as $D = \{D_1, D_2, \ldots, D_M\}$. The size of the input dictionary is $M = \alpha^N$.

$X(kT) = \left[x_1(kT), x_2(kT), \ldots, x_L(kT)\right]$

is the output-word sequence where $k = 0, \pm 1, \pm 2, \ldots$ The output word is a row vector with elements $x_l(kT)$, $l = 1, \ldots, L$ which are the output symbols of the coder as defined below. The period of the output word T is assumed equal to the input-word period. $X(kT)$ is selected from the set of possible blocks of symbols (dictionary) denoted as $X = \{X_1, X_2, \ldots, X_J\}$. The size of the output dictionary is $J \leq \beta^L$, where β is the number of output symbols.

$x(k_xT_x)$ is the output-symbol sequence where $k_x = 0, \pm 1, \pm 2, \ldots$ The period of the output symbols is $T_x = \frac{T}{L}$ and they are selected from the set of possible values (alphabet) denoted as $x = \{x_1, x_2, \ldots, x_\beta\}$.

$y(t)$ is the line signal which is sent through the transmission media. It is produced by modulating the output-symbol sequence with a particular signal waveform. (In the case of PAM $y(t)$ is a continuous-time pulse waveform.)

The model of the line coder consists of four distinct systems. The framing and the deframing systems are linear while the block coder and the digital modulator are non-linear in general. In the case of PAM the digital modulator is a linear system. The functions characterising the relations between the input and the output of each system are introduced below. Linear systems are described in brief as their mathematical representation is straightforward. Also, a linear function can be used to describe certain types of modulating systems which allows the digital modulator to be assumed a linear system without substantial reduction in the generality of the present analysis. The main definition of the block coder is given briefly as more detailed presentation follows in a separate subsection. This is necessary as the description of the block-coder model is of greatest importance to the development of adequate statistical and spectral analysis of the coded signal.

The input-output relation which represents the linear systems of the model can be derived in general form. It is known that for an N-input, L-output system this relation is given by

$$b(t) = \int_{\tau \in \Phi} a(\tau) g(t-\tau) \tag{3.1a}$$

where

$a(\tau)$ input signal – a row vector with N components;
$$\tau \in \Phi = \{0, \pm T_a, \pm 2T_a, \ldots\}$$
$b(t)$ output signal – a row vector with L components;
$$t \in \Psi = \{0, \pm T_b, \pm 2T_b, \ldots\}$$
$g(\theta)$ composite characterising function of the system – an N by L matrix;
$$\theta \in \Gamma = \{\theta = t - \tau \colon\ t \in \Psi;\ \tau \in \Phi\}.$$

The relations between the time-domains Φ, Ψ and Γ determine the type of the system — a filter, an interpolating filter or a decimating filter. When $\tau = kT$ the integral becomes a summation:

$$b(t) = \sum_{k=-\infty}^{+\infty} T a(kT) g(t-kT) \tag{3.1b}$$

The linear systems of the line coder model can now be defined as follows:

FRAMING

This system relates the input-word sequence $D(kT)$ to the input-symbol sequence $d(k_dT_d)$ through the expression

$$d_n(kT) = d\Big((kN+n-1)T_d\Big)$$

where $n = 1,\ldots,N$ for every $k = 0, \pm 1, \pm 2,\ldots$. This is a decimating filter with one input and N outputs where $\Phi \supset \Psi$ and $\Gamma \equiv \Phi$, because

$$\Phi = \{0, \pm T_d, \pm 2T_d,\ldots\} \text{ and } \Psi = \{0, \pm T, \pm 2T,\ldots\}.$$

The characterising function of this system is given by a row vector of delta-functions defined as

$$g_d\Big((kN-k_d)T_d\Big) =$$

$$= \Big[\delta\Big((kN-k_d)T_d\Big), \delta\Big((kN-k_d+1)T_d\Big),\ldots, \delta\Big((kN-k_d+N-1)T_d\Big)\Big] \quad (3.2a)$$

where $\delta\Big((kN-k_d)T_d\Big) = \begin{cases} \frac{1}{T_d}, & \text{for } kN-k_d = 0 \\ 0, & \text{for } kN-k_d \neq 0 \end{cases}, \quad \text{k}, k_d = 0, \pm 1, \pm 2,\ldots$

DEFRAMING

This system relates the output-word sequence $X(kT)$ to the output-symbol sequence $x(k_xT_x)$ through the expression

$$x((kL+l-1)T_x) = x_l(kT)$$

where $l = 1,\ldots,L$ for every $k = 0, \pm 1, \pm 2,\ldots$. This is an interpolating filter with L inputs and one output where $\Phi \subset \Psi$ and $\Gamma \equiv \Psi$, because

$$\Phi = \{0, \pm T, \pm 2T,\ldots\} \text{ and } \Psi = \{0, \pm T_x, \pm 2T_x,\ldots\}.$$

This system has a characterising function given by a column vector of delta-functions defined as

$$g_x((k_x-kL)T_x) = \frac{1}{L}\begin{bmatrix} \delta((k_x-kL)T_x) \\ \delta((k_x-kL-1)T_x) \\ \vdots \\ \delta((k_x-kL-L+1)T_x) \end{bmatrix} \quad (3.2b)$$

where $\delta((k_x-kL)T_x) = \begin{cases} \frac{1}{T_x}, & \text{for } k_x-kL = 0 \\ 0, & \text{for } k_x-kL \neq 0 \end{cases}, \quad k, k_x = 0, \pm 1, \pm 2,\ldots$

The system to be described next transforms the symbol sequence into a continuous waveform which is transmitted over the communication channel. As it has been already mentioned, one of the main objectives of line coding is to achieve spectral shaping which is best suited to the particular transmission media. It has been noted earlier that additional modification of line-signal spectrum can be accomplished through special modulating techniques. The choice of a special modulator function has received considerable attention in the literature [Peebles, 1987]. Consequently, it is not discussed in this presentation and the most simple case of a linear system, i.e. a PAM modulator, has been adopted for completeness of the analysis. Although some types of waveforms may appear to be more appropriate for certain coded sequences, in general a comparative assessment and analysis of the various line codes should be performed on the spectral density distribution achieved through the very process of coding itself.

Digital Modulator

This system represents the functional relationship between the line signal $y(t)$ and the output-symbol sequence $x(k_xT_x)$ by

$$y(t) = \sum_{k_x=-\infty}^{+\infty} \gamma(t - k_xT_x, x(k_xT_x))$$

In general $\gamma(*, x_b)$ denotes the continuous-time waveform sent to the transmission line for the output symbol $x_b \in x$. The process of digital modulation for a PAM system can be expressed through its input/output relation which is given by

$$y(t) = \sum_{k_x=-\infty}^{+\infty} x(k_xT_x)\gamma(t - k_xT_x)$$

where $\gamma(*)$ is a particular pulse waveform. A pulse-amplitude modulator can be viewed as an interpolating filter with one input and one output where $\Phi \subset \Psi$ and $\Gamma \equiv \Psi$, because $\Phi = \{0, \pm T_x, \pm 2T_x, \ldots\}$ and Ψ is the domain of the real numbers. In this case the modulator transfer function is easily determined through the characterising pulse waveform as follows:

$$g_y(t - k_xT_x) = \frac{1}{T_x}\,\gamma(t - k_xT_x) \tag{3.3}$$

The last to be defined, but the most important of all four systems is the block coder. This system is represented by a nonlinear model which can be avoided only for a very limited number of cases. The number of possible states the block coder can be in is the major difference between this system and the other three defined above. It is necessary to evaluate the statistics of the block coder in order to determine the statistics of the coded sequence. This can be achieved if the probabilities of transition between the coder states are found. The output function and the state-transition functions are defined below to specify the sequence of coder sates and output blocks of symbols .

BLOCK CODER

This system relates its next state and the output-word sequence to the input-word sequence and the present state through the expressions:

$$s((k+1)T) = A[\![s(kT), D(kT)]\!] \tag{3.4a}$$

$$X(kT) = B[\![s(kT), D(kT)]\!] \tag{3.4b}$$

where $s(kT)$ denotes the sequence of coder-states taken from the set $s = \{s_1, s_2, \ldots, s_I\}$.

The function $A[\![*,*]\!]$ determines the next coder state $s((k+1)T)$ from the state $s(kT)$ and the input word $D(kT)$ at the previous instant of time. The function $B[\![*,*]\!]$ maps the combination of an input word and a coder state into the set of output words. Thus, the output $X(kT)$ is the response to an input $D(kT)$ and the state $s(kT)$ at the same instant of time.

As the block coder is in general a non-linear system it requires special attention in modelling its input-output relations. This is necessary because higher order statistics have to be evaluated for the subsequent spectral analysis. It has been established that the Finite State Sequential Machine (FSSM) is a very adequate model and a powerful tool for the complete statistical analysis of a fixed-length line coder, [Cariolaro, 1983].

3.2.1.1. Presentation of the Block Coder as an FSSM

The general model of a finite state system represents the two functions A and B relating the three sets D, X and s whose elements are defined as follows:

$D_m \in \{D_1, D_2, \ldots, D_M\}$	is the m-th input word from the set D;
$X_j \in \{X_1, X_2, \ldots, X_J\}$	is the j-th output word from the set X;
$s_i \in \{s_1, s_2, \ldots, s_I\}$	is the i-th coder state from the set s.

The state transition function, $A[\![*,*]\!]$ represents the mapping $s \times D \rightarrow s$ of the two sets s and D into s and defines the next state of the FSSM as

$$s((k+1)T) = A[s(kT), D(kT)].$$

The output function, $B[*,*]$ represents the mapping $s \times D \rightarrow X$ of the two sets s and D into X and defines the output of the FSSM as

$$X(kT) = B[s(kT), D(kT)].$$

The general structure of this type of FSSM is shown in Fig.3.2. The two blocks labelled $A[*,*]$ and $B[*,*]$ are memoryless. They assign the next state $s((k+1)T)$ and the output word $X(kT)$ respectively in terms of the present state $s(kT)$ and the input word $D(kT)$. The third block represents the memory of the coder which holds the output of block $A[*,*]$ for the duration T of a block of symbols.

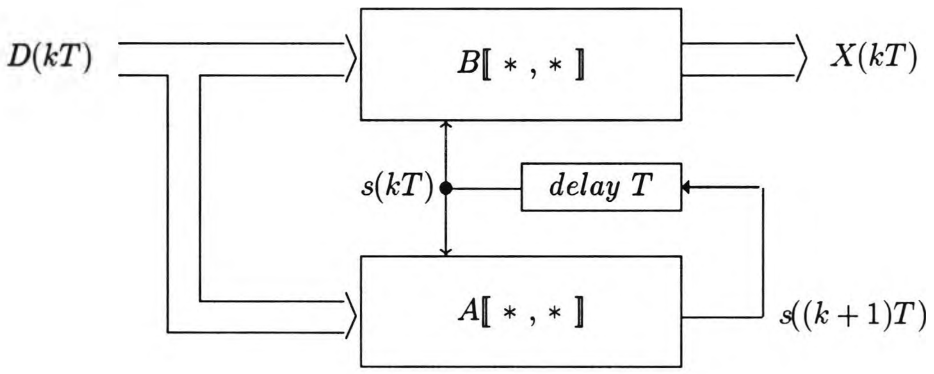


Fig. 3.2 Structure of the FSSM

The FSSM representing a line coder is specified by the nature of the three sets D, s and X. The input and the output words are usually represented as row vectors of the respective symbols. The three sets related by the characterising functions A and B determine the Operational Space[3] of the FSSM. Within this space every output word is assigned to at least one pair of an input word and a state, which is denoted as $B[s_i, D_m] \Rightarrow X_{mi}$, where $X_{mi} \in \{X_1, X_2, \ldots, X_J\}$. In most practical systems the condition for unique decodability is usually applied. In general this condition can be stated as follows: every output word corresponds to only one input word and one or more output words correspond to every input word. The requirement for unique decodability ensures the possibility to recover the original sequence of words without knowledge of the coder state sequence.

[3] The ideas about the Coder Operational Space are discussed in greater detail in Chapter 6.

State-dependent decoding is also employed in some real systems. The main problems in such systems are to restrict error propagation as well as to maintain circuit complexity within reasonable limits. The efforts, however, could be well rewarded with higher code efficiency and greater flexibility in spectrum shaping.

Three methods are mainly used to specify the functional relations between the three sets of the FSSM :

Transitional diagrams These are oriented graphs with I vertices which represent the states $s_1, s_2, \ldots, s_I$. An oriented branch of the graph corresponds to every possible transition from state s_i to state s_{mi} indicated by the function $A[\![s_i, D_m]\!] \Rightarrow s_{mi}$, where $s_{mi} = \{s_1, s_2, \ldots, s_I\}$.

The condition for the FSSM to be strongly connected, can be verified by checking whether the graph is strongly connected. This means there is a sequence of input words $D_{m_1}, D_{m_2}, \ldots, D_{m_r}$, for every ordered pair of states $(s_i, s_{m_r i})$, which changes the state of the block coder from s_i into $s_{m_r i}$.

Table presentation A table is constructed to reflect the characterising functions A and B. The rows of this table correspond to all input words and the columns correspond to all possible states. The intersection of row D_m with column s_i corresponds to the output word X_{mi} and the next state s_{mi}.

The general form of table presentation is shown in Fig. 3.3.

D_m/s_i	s_1	s_2	$\cdots$	s_I
D_1	X_{11} , s_{11}	X_{12} , s_{12}	$\cdots$, $\cdots$	X_{1I} , s_{1I}
D_2	X_{21} , s_{21}	X_{22} , s_{22}	$\cdots$, $\cdots$	X_{2I} , s_{2I}
$\vdots$	$\vdots$, $\vdots$	$\vdots$, $\vdots$	$\vdots$, $\vdots$	$\vdots$, $\vdots$
D_M	X_{M1} , s_{M1}	X_{M2} , s_{M2}	$\cdots$, $\cdots$	X_{MI} , s_{MI}

Fig. 3.3 Table presentation of the block coder as a FSSM

The statistical analysis of the line coder model is most conveniently carried out by means of matrix algebra. Therefore two types of matrices are defined to represent the characterising functions of the respective FSSM as follows:

Matrix presentation An *output matrix* is defined for every input word D_m, $(m = 1,\ldots,M)$

$$Z_m = \begin{bmatrix} X_{m1} \\ X_{m2} \\ \vdots \\ X_{mI} \end{bmatrix} = \begin{bmatrix} X_m(1,1) & X_m(1,2) & \ldots & X_m(1,L) \\ X_m(2,1) & X_m(2,2) & \ldots & X_m(2,L) \\ \vdots & \vdots & \vdots & \vdots \\ X_m(I,1) & X_m(I,2) & \ldots & X_m(I,L) \end{bmatrix} \tag{3.5a}$$

The size of the output matrices is $I \times L$. Every row of an output matrix Z_m is an output word $X_{mi} = \{X_1, X_2, \ldots, X_J\}$ with elements $X_m(i,l) = \{x_1, x_2, \ldots, x_\beta\}$, for $i = 1,\ldots,I$ and $l = 1,\ldots,L$. In other words an output matrix is a column vector of all output words for a given input word, ordered according to the ordering of the states.

A *state transition matrix* is defined for every input word D_m as follows

$$S_m = \begin{bmatrix} S_m(1,1) & S_m(1,2) & \ldots & S_m(1,I) \\ S_m(2,1) & S_m(2,2) & \ldots & S_m(2,I) \\ \vdots & \vdots & \vdots & \vdots \\ S_m(I,1) & S_m(I,2) & \ldots & S_m(I,I) \end{bmatrix} \tag{3.5b}$$

The size of the matrices is $I \times I$. There is one state transition matrix for every input word and its entries are determined by

$$S_m(i,j) = \begin{cases} 1, & \text{if } A[s_i, D_m] = s_j \\ 0, & \text{otherwise} \end{cases}$$

In words — the elements of S_m with indices (i,j) are equal to 1 only when the FSSM changes from state s_i into s_j while the input word is D_m. Otherwise the elements of S_m are 0. A very important feature of the state transition matrix is that all of its rows have exactly one non-zero element.

3.2.1.2 The 'Mealy' and 'Moore' FSSM Models

The matrix presentation of the block coder as a FSSM is used for the subsequent statistical and spectral analysis. Any practically implemented line coder can be modelled into different equivalent FSSM-s. This statement is supported at least by the fact that all real line coders are designed and built on memory and logic components with a finite number of states. It is probably possible to analyse a particular coder on the basis of any relevant FSSM model, but the level of computational complexity may differ widely. Therefore it is important to select the model which is easiest to perform the analysis with. This can be achieved through recognising equivalent FSSM-s and identifying the minimal FSSM within the class of equivalence that machine belongs to. For this purpose the following definitions should be applied:

- Two equivalent FSSM-s have the same input and output sets and for every state of one at least one state of the other exists, such that the output sequences of the two machines, resulting from the same input sequences, are identical when the corresponding states are the initial ones.

- FSSM-s can be grouped into classes of equivalence and there is a unique minimal FSSM in every equivalence class. The minimal machine has a minimal number of states and can be obtained from any other of the same class through a transformation procedure.

The two characterising functions (A, B) and the three sets (D, s, X) used to describe the line coder model comprise the so called ‘Mealy machine’. It has been established,[18] that evaluation of the statistics of the coder states is easier to perform for a slightly simpler model called the ‘Moore machine’. The latter is equivalent to the Mealy machine and is derived through assuming new characterising functions, $\mathcal{A}$ and $\mathfrak{B}$ and a set of states, σ as shown below. (The $\rightarrow$ symbols indicate transformations.)

$$\boxed{\begin{array}{ll} \mathrm{D}((k+1)T) & \rightarrow \mathfrak{D}(kT) \\ & \\ \{s(kT), D(kT)\} & \rightarrow \sigma(kT) \end{array}} \Rightarrow \boxed{\begin{array}{rcl} \sigma((k+1)T) = & \mathcal{A}[\![\sigma(kT), \mathfrak{D}(kT)]\!] \\ & = & \{A[\![s(kT), D(kT)]\!], \mathrm{D}((k+1)T)\} \\ X(kT) & = & \mathfrak{B}[\![\sigma(kT)]\!] = B[\![s(kT), D(kT)]\!] \end{array}}$$

The simplification is due to the new output function $\mathfrak{B}[\![*]\!]$ which makes the output words dependent only on a set of states and also to the new output matrices which are equal, $\mathfrak{Z}_m = \mathfrak{Z}$ for $m = 1, \ldots, M$. The simpler presentation of the Moore model refers to the relations between the elements of the model and is achieved at the expense of their size. This can be shown by relating the new state-transition and output matrices to those of the Mealy model. If the new states are ordered by $\sigma_{(m-1)I+i} = \{s_i, D_m\}$ and the relation between the output words and the new states is $X_{i_m} = \mathfrak{B}[\![\sigma_{i_m}]\!]$, $i_m = 1,\ldots,(I \times M)$, then the number of new states is $I \times M$ and the new state-transition and output matrices are given by:

$$\mathcal{S}_1 = \begin{bmatrix} S_1 & 0 & \ldots & 0 \\ S_2 & 0 & \ldots & 0 \\ \vdots & \vdots & \vdots & \vdots \\ S_M & 0 & \ldots & 0 \end{bmatrix}, \ldots, \mathcal{S}_M = \begin{bmatrix} 0 & 0 & \ldots & S_1 \\ 0 & 0 & \ldots & S_2 \\ \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \ldots & S_M \end{bmatrix}; \mathfrak{Z} = \begin{bmatrix} Z_1 \\ Z_2 \\ \ldots \\ Z_M \end{bmatrix},$$

where the size of $\mathfrak{Z}$ is $(I \times M)$ by L and the size of $\mathcal{S}_m$, $m = 1,\ldots,M$ is $(I \times M)$ by $(I \times M)$.

The equivalence of the Moore and Mealy models will be used in following sections where the statistical and the spectral analyses are developed. This equivalence allows the statistics of the coded sequence and the spectral density of the signals in the Mealy model to be derived from the results obtained for the Moore model.

3.2.2 Statistical Analysis of the Sequences in the Line-Coder Model

The evaluation of the statistics of the sequences of symbols and words in the coder model is based on the assumption that the input symbols are independent and identically distributed (i.e. they are defined as a Stationary Memoryless Source (SMS)). The input-symbol probabilities are given by

$$q(d_\nu) = p\Big\{d(k_d T_d) = d_\nu\Big\}, \quad d_\nu \in d$$

When the input-symbol sequence is a SMS so is the input-word sequence whose probabilities are

$$Q_m = p\{D(kT) = D_m\} = \prod_{n=1}^{N} q(d_{mn}) \tag{3.6}$$

where d_{mn} is the n-th symbol in the input word D_m and $D_m \in D$. Having defined the probabilities of the input-word sequence $D(kT)$ as an SMS it becomes possible to determine the statistics of the state sequence $s(kT)$ which is related to the input-word sequence through $s((k+1)T) = A[\![s(kT), D(kT)]\!]$. It can be proven [17] that the state sequence of a FSSM driven by a SMS is a homogeneous Markov chain with Transition Probability Matrix (TPM) given by

$$S = \sum_{m=1}^{M} Q_m S_m \tag{3.7}$$

This result is fundamental because it allows the evaluation of second order statistics of the output-word sequence, which is used to derive the expressions for the spectral density.

3.2.2.1. Evaluation of State Sequence Probabilities

The statistics of the coder-state sequence, which is stationary, are completely specified by the TPM because it is a stochastic matrix (its elements are non-negative and their sum in every row is one) and $s(kT)$ is ergodic and regular. The features of ergodicity (every state can be reached from every other) and regularity ($s(kT)$ has an aperiodic structure) follow from the assumption of an FSSM model which is strongly connected and input-word probabilities which are strictly positive. Three important results are based on the assumptions made above:

- The first order state probabilities are the elements of a row vector $P = [P(1), P(2), \ldots, P(I)]$, where $P(i) = p\{s(kT) = s_i\}$. This vector is uniquely determined as the solution of the system of matrix equations

$$P = PS \tag{3.8a}$$

$$\sum_{i=1}^{I} P(i) = 1 \tag{3.8b}$$

- The probability of the states after r transitions are the elements of the r-step TPM $S^{(r)}$, which are given by

$$S^{(r)}(i,j) = p\{s((k+r)T) = s_j \mid s(kT) = s_i\}.$$

  The r-step TPM for $r > 0$ can be found from

$$S^{(r)} = S^r \tag{3.9}$$

- Higher order probabilities can be evaluated from the following expression

$$p\left\{s(kT) = s_{i_0}, s((k+r_1)T) = s_{i_1}, \ldots, s\big((k+r_1+r_2+\ldots+r_\xi)T\big) = s_{i_\xi}\right\} =$$

$$= p\left\{s(kT) = s_{i_0}\right\} \times p\left\{s((k+r_1)T) = s_{i_1} \mid s(kT) = s_{i_0}\right\} \times \ldots \times$$

$$\times\, p\left\{s\big((k+r_1+r_2+\ldots+r_\xi)T\big) = s_{i_\xi} \mid s\big((k+r_1+r_2+\ldots+r_{\xi-1})T\big) = s_{i_{\xi-1}}\right\} =$$

$$= P(i_0) S^{(r_1)}(i_0, i_1) S^{(r_2)}(i_1, i_2) \times \ldots \times S^{(r_\xi)}(i_{\xi-1}, i_\xi) \tag{3.10}$$

where $\xi > 0$ and $r_1, r_2, \ldots, r_\xi \geq 0$ are integers.

3.2.2.2. Probabilities of the Output Sequences

The evaluation of the statistics of the output-word sequence, $X(kT)$ is based on expression (3.10) and on a 'careful' transition from the statistics of the Moore model to those of the Mealy FSSM. The final result as well as the results of some intermediate steps are outlined below. Most of these results can be stated as theorems and proven strictly. For the purposes of this presentation it is sufficient to specify the essential conditions which determine the main steps in the transition between the two models. The intention is to present the general idea of applying the functional relations of the FSSM model in deriving the final expressions for the statistics of the coded sequence.

The statistical analysis is based on the input-word probabilities Q_m determined through (3.6) and the specification of the state transition matrices S_m given by (3.5b). These values allow to determine the probability matrix S and the vector P containing the first order probabilities of the sequence of coder states, expressions (3.7) and (3.8) respectively. The generalisation of this result has led to (3.10), which gives the state probabilities of arbitrary high order. At this stage appropriate conditions are specified so that a relatively straightforward transition can be made from the statistics of the state sequence to the output word probabilities in a Moore FSSM model. The relation between the two models, described in section 3.2.1.2, allows for the same conditions to be applied to the characterising functions in a Mealy model, given by (3.4). The important consequence is the possibility to determine the statistics of the sequence $X(kT)$ from the probabilities of the state sequence $\sigma(kT)$. The TPM, $\mathcal{S}$ and the state-probability vector $\mathcal{P}$ corresponding to $\sigma(kT)$ can be evaluated through the definition of the matrices $\mathcal{S}_m$, therefore through S and P respectively. This leads to the main result – the expressions relating the probabilities of the output word sequence to the statistics of the input sequences, Q_m and the respective state-transition matrices S_m.

The most essential stages of the transition described briefly in the last paragraph are defined below together with the corresponding analytical expressions. However, no strict proofs have been given as the main purpose of the theoretical part of this presentation is to provide the minimal complete and consistent set of formulae relating the initial specification parameters of a coder to the final expression for the spectral density function. The four main stages are as follows.

1) The statistics of the output-word sequence in the Moore model can be determined through the special definition of the output-word function $X(kT) = \mathfrak{B}[\![\sigma(kT)]\!]$ which provides for the output words to be state dependent only. The Moore FSSM can be considered specified independent of a Mealy model and it is possible to prove the following important result:

The output-word sequence of a Moore machine driven by a SMS is a memoryless time-invariant function of a homogeneous Markov chain.

As an output word X_{i_μ} is assigned to a particular state s_{i_μ} through the functional relation $X_{i_\mu} = B[\![\sigma_{i_\mu}]\!]$ which indicates the identity of the events $\{X(kT) = X_{i_\mu}\} \equiv \{s(kT) = s_{i_\mu}\}$, the statistics of the output-word sequence are given by

$$p\Big\{X(kT) = X_{i_0}, X\big((k+r_1)T\big) = X_{i_1}, \ldots, X\big((k+r_1+r_2+\ldots+r_\xi)T\big) = X_{i_\xi}\Big\} =$$

$$= P(i_0)S^{(r_1)}(i_0,i_1)S^{(r_2)}(i_1,i_2) \times \cdots \times S^{(r_\xi)}(i_{\xi-1},i_\xi) \qquad (3.11)$$

2) The considerations which allow the statistics of the output-word sequence to be found, still hold when the Moore model is derived from a Mealy FSSM. In particular the following two statements can be proven:

a) In a Mealy machine driven by the sequence $\mathfrak{D}(kT) = D((k+1)T)$, which is stationary and memoryless, the combined sequence of states and input words $\sigma(kT) = \{s(kT), D(kT)\}$, given recursively by

$$\sigma((k+1)T) = \mathcal{A}[\![\sigma(kT), \mathfrak{D}(kT)]\!] = A[\![s(kT), D(kT), D((k+1)T)]\!],$$

is a homogeneous Markov chain[4].

b) The sequence of the output words $X(kT) = \mathfrak{B}[\![\sigma(kT)]\!] = B[\![s(kT), D(kT)]\!]$ is a memoryless time-invariant function of a homogeneous Markov chain.

The above statements specify essentially the same as in 1) but also reveal the relations which allow the output-words probabilities in a Mealy model to be derived from the statistics of the sequence $\sigma(kT)$. The TPM of $\sigma(kT)$ is given by

---

[4] $A[\![s(kT), D(kT)]\!]$ is in fact $s((k+1)T)$, therefore $\sigma((k+1)T)$ can be considered independent of an input from previous instants.

$$\mathcal{S} = \sum_{\mu=1}^{M} Q_\mu \mathcal{S}_\mu = \| \mathcal{S}_{mh} \| = \| Q_h S_m \| \qquad (3.12a)$$

where $\| \mathcal{S}_{mh} \|$ is a partitioning of $\mathcal{S}$ defined as follows: the element of $\mathcal{S}_{mh}$ with indices (i, j) is the element of $\mathcal{S}$ with indices $((m-1)I + i, (h-1) + j)$. The state probabilities of $\sigma(kT)$ are the elements of the row vector

$$\mathcal{P} = \| \mathcal{P}_h \| = \| Q_h P \| \qquad (3.12b)$$

The r-step TPM is given by

$$\mathcal{S}^r = \left\| \mathcal{S}_{mh}^{(r)} \right\| = \left\| Q_h S_m \mathcal{S}^{r-1} \right\|, \text{ for } r \geq 1 \qquad (3.12c)$$

3) It is useful to summarise the significant results which allow for the final expression of the output-word probabilities to be produced.

i) The input-symbol sequence $d(kT)$ and the input-word sequences $D(kT)$ are SMS-s and their probabilities have been defined.

ii) The sequence of states $s(kT)$ of an FSSM driven by the input-word sequence defined as a SMS is a homogeneous Markov chain whose statistics are specified by the TPM S. The first order probabilities $P(i)$ of the coder states and the r-th order TPM $S^{(r)}$ are given by (3.8) and (3.9).

iii) The general expression for higher order probabilities of the coder state sequence $s(kT)$ is given by (3.10).

iv) In an FSSM specified directly as a Moore machine and driven by a SMS the output-word sequence $X(kT)$ is a memoryless time-invariant function of a homogeneous Markov chain and is state dependent only. This allows to determine the expression for the output-word probabilities, (3.11).

v) The statistics of the state sequence $\sigma(kT)$ and the output-word sequence $X(kT)$ are defined for a Moore FSSM derived from a Mealy model. The expressions (3.12) relate the TPM $\mathcal{S}$ and the state-probability vector $\mathcal{P}$ to S and P respectively.

vi) The final transition to the output-word probabilities in a Mealy FSSM is based on the functional relation $X_{mi} = B[\![s_i, D_m]\!]$ and the identity of the events

$$\left\{X(kT) = X_{mi}\right\} \equiv \left\{s(kT) = s_i, D(kT) = D_m\right\} \equiv \left\{\sigma(kT) = \sigma_{(m-1)I+i}\right\}$$

Therefore the first, the second and the ξ-th order statistics of the code-word sequence are given respectively by:

$$p\left\{X(kT) = X_{mi}\right\} = \mathcal{P}_m(i) = Q_m P(i) \tag{3.13a}$$

$$p\left\{X(kT) = X_{mi}, X((k+r)T) = X_{hj}\right\} =$$
$$= \mathcal{P}_m(i)\mathcal{I}_{mh}^{(r)}(i,j) = Q_m P(i)\sum_{\epsilon=1}^{I} Q_h S_m(i,\epsilon) S^{(r-1)}(\epsilon, j) \tag{3.13b}$$

$$p\left\{X(kT) = X_{m_0 i_0}, X((k+r_1)T) = X_{m_1 i_1}, \dots, X\left((k+r_1+\dots+r_\xi)T\right) = X_{m_\xi i_\xi}\right\} =$$
$$= \mathcal{P}_{m_0}(i_0)\mathcal{I}_{m_0 m_1}^{(r_1)}(i_0, i_1) \times \dots \times \mathcal{I}_{m_{\xi-1} m_\xi}^{(r_\xi)}(i_{\xi-1}, i_\xi) =$$
$$= Q_{m_0} P(i_0) \sum_{\epsilon_1=1}^{I} Q_{m_1} S_{m_0}(i_0, \epsilon_1) S^{(r_1-1)}(\epsilon_1, i_1) \times \sum_{\epsilon_2=1}^{I} Q_{m_2} S_{m_1}(i_1, \epsilon_2) S^{(r_2-1)}(\epsilon_2, i_2) \times \dots$$
$$\dots \times \sum_{\epsilon_\xi=1}^{I} Q_{m_\xi} S_{m_{\xi-1}}(i_{\xi-1}, \epsilon_\xi) S^{(r_\xi-1)}(\epsilon_\xi, i_\xi) \tag{3.13c}$$

4) Finally, it is possible to obtain the probabilities of the output-symbol sequence by referring to the relation $x((kL+l-1)T_x) = x_l(kT)$ which indicates the identity of the events

$$\{x((kL+l-1)T_x) = X_m(i,l)\} \equiv \left\{X(kT) = X_{mi}\right\}.$$

Therefore the output-symbol probabilities are given by

$$p\{x((kL+l-1)T_x) = X_m(i,l)\} = p\left\{X(kT) = X_{mi}\right\}$$

where $k = 0, \pm 1, \pm 2, \dots$; $l = 1, \dots, L$ and $m = 1, \dots, M$. It is a matter of some conventional matrix algebra to find the expression for the higher order statistics of the output-symbol sequence.

3.2.3. Spectral Analysis of the Line-Coder Signals

The frequency analysis theory, presented briefly in this section, applies to discrete-time signals in general and will be used to determine the spectral density functions of the sequences defined in the line-coder model. The same results can be transformed straight-forwardly into a more general form so that their validity extends to continuous-time signals as well. The signals under consideration in the present analysis are random sequences. To evaluate a random process in the frequency domain the mean value and the correlation are used. The validity of the results depends on the stationarity of the signals. In general the signals involved in the process of fixed-length line coding are cyclo-stationary. It is possible to show that under certain conditions the frequency analysis of stationary processes applies to cyclo-stationary signals as well.

The spectral analysis of the line-coded signal requires very involved theoretical assessment. As already mentioned, the spectral density of the output signals cannot be obtained through linear system analysis from the frequency functions of the input signals. The FSSM model of a coder has been used to derive the statistics of the output word sequence from the probabilities of the input signal and the coder state sequence. This allows for the frequency analysis of the output word sequence, $X(kT)$ to be performed. The Deframing part and the Modulator have been defined as linear systems and the spectral density function of the coded line signal can be obtained accordingly.

Some of the main results from the theory of spectral analysis are discussed very briefly in the following subsections. The most essential general expressions are given in order to trace the stages of their transformation into the final formulae for the spectral density functions of the coded signal. This transformation can be summarised in the following three steps:

- Firstly the expression for the spectral density function is derived in a general form.

- The next step involves using the results from section 3.2.3.2 to define the correlation function through the first and second order probabilities of the output-word sequence and determine the continuous and the discrete parts of its spectral distribution.

- Finally, as the output-symbol sequence and the line signal result from linear transformations, their spectral densities are expressed through the frequency functions of the respective input signals.

3.2.3.1. Spectral Density Function

The mean value vector V_a and the correlation matrix R_a of a stationary random process, which is defined as a row vector $a(kT) = [a_1(kT), a_2(kT), \ldots, a_N(kT)]$, $k = 0, \pm 1, \pm 2, \ldots$, are given by

$$V_a = E\langle a(kT) \rangle$$

$$R_a(\rho T) = E\langle a'(kT) a((k+\rho)T) \rangle, \; \rho = 0, \pm 1, \pm 2, \ldots$$

Both V_a, a vector with N components and $R_a(\rho \mathrm{T})$, which is an N by N matrix, are independent of the reference time kT. In the case of discrete signals the usual definition of the spectral density as the Fourier transform of the correlation is given by

$$W_a(f) = \sum_{\rho=-\infty}^{+\infty} T R_a(\rho T) \exp(-j2\pi f \rho T)$$

The spectral density function matrix, $W_a(f)$ has a variety of properties which allow for the frequency analysis to be presented in a more convenient form. One of them is the possibility, under certain conditions to decompose $W_a(f)$ into a discrete component $W_a^{(d)}(f)$ and a continuous component $W_a^{(c)}(f)$. Such conditions are assumed to exist for most cases of interest in digital signal transmission.

A rigorous mathematical analysis is required for the decomposition of the spectral density function. A thorough treatment of the underlying theory has not been included in this presentation, as analytical results only are given to indicate how each expression follows from the previous ones.

The correlation of the random process $a(kT)$ can be expressed as the sum of its covariance, $J_a(\rho T)$ and the squared mean value[5], i.e.

$$R_a(\rho T) = J_a(\rho T) + V_a' V_a.$$

It is possible to view the term $V_a' V_a$ as the discrete part of the correlation, which derives from the last expression[6] for $\rho \rightarrow \infty$:

[5] The mathematical operations should be interpreted with respect to the definition of $a(kT)$ as a vector.

[6] Assuming the random process exhibits asymptotic uncorrelation.

$$R_a(\infty) = \lim_{\rho\to\infty} R_a(\rho T) = V'_a V_a$$

Consequently $J_a(\rho T)$ is assumed to represent the continuous part of the correlation. This allows for the spectral density decomposition to be defined as the Fourier transform of the discrete and continuous parts of $R_a(\rho T)$.

$$\boxed{\begin{array}{l} R_a^{(d)}(\rho T) = V'_a V_a \\ \\ R_a^{(c)}(\rho T) = R_a(\rho T) - V'_a V_a \end{array}} \Rightarrow \boxed{\begin{array}{l} W_a^{(d)}(f) = V'_a V_a \sum\limits_{\rho=-\infty}^{+\infty} \delta(f - \rho F) \\ W_a^{(c)}(f) = \sum\limits_{\rho=-\infty}^{+\infty} T J_a(\rho T)\, z^{-\rho} \end{array}} \qquad (D1)$$

where $F = \frac{1}{T}$ and $z = \exp(j2\pi fT)$. It is possible to represent the last expression as a one-sided series by making use of the equality $J_a(-\rho T) = J'_a(\rho T)$. In the case of discrete-time random processes the continuous part of the spectral density can be presented as the two-sided z-transform of the covariance. The continuous part of the spectral density then takes the form

$$W_a^{(c)}(f) = T\Big[\mathrm{W}(z) + W'(z^{-1})\Big] \qquad (D2)$$

where $W(z) = 0.5J_a(0) + \sum\limits_{\rho=1}^{+\infty} T J_a(\rho T) z^{-\rho}$

The main results from the spectral analysis theory presented so far refer to stationary random signals. When the random sequence $a(kT)$ is cyclo-stationary, the mean value and the correlation are not time-invariant, i.e. they depend not only on the time displacement ρT, but are also periodic functions of kT. This can be denoted as:

$$V_a(kT) = E\langle a(kT)\rangle \text{ and } R_a(kT, \rho T) = E\Big\langle a'(kT) a((k+\rho)T)\Big\rangle$$

The period of cyclostationarity can be any multiple of T, say ηT and the information about the mean value and the correlation with respect to their dependence of kT is given by their averages over that period, namely

$$\overline{V}_a = \frac{1}{\eta T} \sum_{k=0}^{\eta - 1} T V_a(kT)$$

$$\overline{R}_a(\rho T) = \frac{1}{\eta T} \sum_{k=0}^{\eta - 1} T R_a(kT, \rho T),\ \rho = 0, \pm 1, \pm 2, \ldots$$

The average spectral density for the discrete-time processes is defined accordingly as

$$\overline{W}_a(f) = \sum_{\rho=-\infty}^{+\infty} T\overline{R}_a(\rho T)\exp(-j2\pi f\rho T)$$

At this point it is important to note that the properties of the average parameters of a cyclostationary signal are the same as those of the corresponding parameters of a stationary signal. Proofs of this can be found in the appropriate literature [Gardner, 1975] and similar results can also be achieved through a variety of ways, including phase randomisation and spectral analysis involving two argument correlation. In developing the frequency analysis of line coding it is quite sufficient for most cases of interest to use the average parameters as defined above. For convenience the 'average' sign will be omitted in the subsequent presentation.

3.2.3.2. Frequency Analysis of the Line-Coder Output-Word Sequence

It has been mentioned earlier that the output-word spectral density cannot be obtained through a linear relationship with the corresponding parameters of the input-word sequence, because the block coder is a non-linear system. Therefore the spectral analysis can only be developed by using the second order probabilities of the sequence of output words. The general expressions for the probabilities of any order have been introduced in section 3.2.2.2 for both the Moore and the Mealy FSSM models. These expressions are used to evaluate the mean and the correlation through their statistical dependence on the input-word probabilities, the state probabilities and the output matrices. For a Mealy model the mean can be determined as follows

$$V_X = E\langle X(kT)\rangle =$$

$$= \sum_{m=1}^{M}\sum_{i=1}^{I} X_{mi}\boxed{p\left\{X(kT) = X_{mi}\right\}} = \sum_{m=1}^{M}\sum_{i=1}^{I} X_{mi}\boxed{Q_m P(i)} \qquad (3.14a)$$

By using second order probabilities and the equality

$$p\left\{X(kT) = X_{mi}, X((k+0)T) = X_{hj}\right\} = Q_m P(i)$$

the correlation of the output word sequence in a Mealy model is presented as two parts, corresponding to $r = 0$ and $r \geq 1$, respectively.

$$R_X(rT) = E\langle X'(kT)X((k+r)T)\rangle =$$

$$= \sum_{m,h=1}^{M} \sum_{i,j=1}^{I} X'_{mi} X_{hj} \boxed{p\{X(kT) = X_{mi}, X((k+r)T) = X_{hj}\}} =$$

$$= \begin{cases} \sum_{m=1}^{M} \sum_{i=1}^{I} X'_{mi} X_{mi} Q_m P(i) & \text{for } r = 0 \\ \sum_{m,h=1}^{M} \sum_{i,j=1}^{I} X'_{mi} X_{hj} \boxed{Q_m P(i) \sum_{\epsilon=1}^{I} Q_h S_m(i,\epsilon) S^{(r-1)}(\epsilon, j)} & \text{for } r \geq 1 \end{cases} \tag{3.14b}$$

The two pairs of framed expressions, shown in (3.14a) and (3.14b) indicate the substitutions which follow from (3.13a) and (3.13b) respectively. Some conventional matrix-algebra transformations of the above expressions allow for much better presentable notation to be used for V_X and $R_X(rT)$. In this respect the following rearrangements have been used:

$$\sum_{i=1}^{I} X'_{mi} X_{mi} Q_m P(i) = Q_m \sum_{i=1}^{I} X'_{mi} P(i) X_{mi} = Q_m Z'_m v Z_m$$

$$\sum_{i,j=1}^{I} X'_{mi} X_{hj} Q_m P(i) \sum_{\epsilon=1}^{I} Q_h S_m(i,\epsilon) \overset{(r-1)}{S(\epsilon, j)} =$$

$$= Q_m Q_h \sum_{i,j=1}^{I} X'_{mi} \underbrace{\left(P(i) \sum_{\epsilon=1}^{I} S_m(i,\epsilon) \overset{(r-1)}{S(\epsilon, j)} \right)}_{S_m S^{r-1}} X_{hj} = Q_m Q_h Z'_m v S_m S^{r-1} Z_h$$

where Z_m and Z_h belong to the set of output matrices; $v = \mathrm{diag}[P]$ is a diagonal matrix whose non-zero elements are the components of the state-probabilities vector P. When the above expressions are substituted in (3.14a) and (3.14b) the mean and the correlation become

$$V_X = \sum_{m=1}^{M} Q_m P Z_m \tag{3.15a}$$

$$R_X(rT) = \begin{cases} \sum_{m=1}^{M} Q_m Z'_m v Z_m & \text{for } r = 0 \\ \sum_{m,h=1}^{M} Q_m Q_h Z'_m v S_m S^{r-1} Z_h & \text{for } r \geq 1 \end{cases} \tag{3.15b}$$

The evaluation of the spectral density can now be presented in a very convenient and elegant matrix form. To do this it is necessary to examine the behaviour of the higher order TPM, S^r for the limiting case $r \to \infty$. This would allow to determine the discrete and the continuous part in the decomposition of $W_X(f)$ by deriving expressions for the 'infinity' correlation $R_X(\infty)$ and the covariance $J_X(rT)$.

It has been assumed that the sequence of coder states $s(kT)$ is an ergodic and regular Markov chain. The theory describing this type of random sequences has been extensively developed, [7] and some of the important results, relevant to the present analysis are shown below. If S_∞ denotes the limiting TPM of $S^{(r)}$ when r is infinitely large and $u = [1, 1, \ldots, 1]'$ is a column vector of ones then the following can be proven

$$S_\infty = \lim_{r \to \infty} S^r = uP \tag{3.16}$$

This allows the limiting value of the correlation to be determined as

$$R_X(\infty) = \lim_{r \to \infty} R_X(rT) =$$

$$= \sum_{m,h=1}^{M} Q_m Q_h Z'_m v \underbrace{S_m S_\infty}_{S_\infty} Z_h = \sum_{m,h=1}^{M} Q_m Q_h Z'_m \underbrace{vu}_{P'} P Z_h =$$

$$= \sum_{m,h=1}^{M} Q_m Q_h Z'_m P' P Z_h = V'_X V_X$$

In the algebraic transformations of the last expression the following three relations have been used: $S_m S_\infty = S_\infty$; $vu = P'$ and (3.15a). At this stage the discrete part of the output-word spectral density can be determined by combining (3.15a) and definition ($D1$) given in section 3.2.3.1, i.e.

$$W_X^{(d)}(f) = \boxed{V'_X V_X \sum_{r=-\infty}^{+\infty} \delta(f - rF)}, \text{ where } F = \frac{1}{T} \tag{3.17}$$

It is worth noting an essential feature of the output-word spectral density, namely that its discrete part represents spectral lines at multiples of the word rate F which are related to the asymptotic behaviour of the correlation. In order to apply the definition for the continuous part of the spectral density it is necessary to determine the covariance with respect to (3.15b), i. e.

$$J_X(rT) = R_X(rT) - R_X(\infty) =$$

$$= \begin{cases} \left(\sum_{m=1}^{M} Q_m Z'_m v Z_m\right) - V'_X V_X & \text{for } r = 0 \\ \sum_{m,h=1}^{M} Q_m Q_h Z'_m v S_m\left(S^{r-1} - S_\infty\right) Z_h & \text{for } r \geq 1 \end{cases}$$

By substitution in ($D2$) the complete expression for $W_X^{(c)}(f)$ takes the form

$$W_X^{(c)}(f) = \left(\sum_{m=1}^{M} Q_m Z'_m v Z_m\right) - V'_X V_X +$$

$$+ \sum_{m=1}^{M} Q_m Z'_m v S_m\left(\sum_{r=1}^{\infty}\left(S^{r-1} - S_\infty\right) z^{-r}\right)\sum_{h=1}^{M} Q_h Z_h +$$

$$+ \left(\sum_{m=1}^{M} Q_m Z'_m v S_m\left(\sum_{r=1}^{\infty}\left(S^{r-1} - S_\infty\right) z^{r}\right)\sum_{h=1}^{M} Q_h Z_h\right)' \qquad (3.18a)$$

The special properties of the terms comprising the summation over r in the last expression are well studied in the theory of the ergodic and regular Markov chains [27]. The two important results given below are based on the ergodicity and the regularity of the TPM, S and the limiting TPM, S_∞.

- ▹ $S^r - S_\infty = (S - S_\infty)^r$, for all $r > 0$

- ▹ $(S - S_\infty)^r$ is absolutely summable

They allow for $W(z)$, as defined in ($D2$), to be transformed into a finite sum, using the following equality

$$z^{-1}\sum_{r=0}^{\infty}(S - S_\infty)^r z^{-r} = (zU - S + S_\infty)^{-1} \qquad (3.18b)$$

With some conventional matrix-algebra the term representing summation over r in (3.18a) is transformed as shown below:

$$\sum_{r=1}^{\infty}\left(S^{r-1} - S_\infty\right) z^{-r} = (U - S_\infty) z^{-1} \sum_{r=0}^{\infty}(S - S_\infty)^r z^{-r}$$

After applying equality (3.18b) to the last result and adopting the following notation

$$H(z) = (U - S_\infty)(zU - S + S_\infty)^{-1} \tag{3.19a}$$

$$H_0 = \sum_{m=1}^{M} Q_m Z'_m v Z_m, \qquad H_1 = \sum_{m=1}^{M} Q_m Z'_m v S_m, \qquad H_2 = \sum_{h=1}^{M} Q_h Z_h \tag{3.19b}$$

the final expression for the continuous part of the spectral density is given by

$$\boxed{W_X^{(c)}(f) = T\left[W(z) + W'(z^{-1})\right]} \tag{3.20a}$$

$$\text{where } W(z) = \tfrac{1}{2}\left(H_0 - V'_X V_X\right) + H_1 H(z) H_2 \tag{3.20b}$$

The frequency analysis of the block coder, modelled as a non-linear system results in two expressions giving the discrete and the continuous parts of the spectral density distribution of the output-word sequence. The final expressions for $W_X^{(d)}(f)$ and $W_X^{(c)}(f)$, given by (3.17) and (3.20a), are of great importance to the spectral analysis of the line coder model for two main reasons. The first is the broad validity of the results which is based on the general statistical presentation of the model as a FSSM. The second reason is the possibility to determine the spectral functions for the output-symbol sequence and the line signal from the frequency distribution of the input signals to the respective systems when the latter are linear.

3.2.3.3 Linear-System Spectral Analysis

To present a linear system in the frequency domain the Fourier transform of the characterising function $g(*)$ given in (3.1) is used. This results in the following expression for the frequency response of a linear system

$$\boxed{G(f) = \int_{\theta \in \Gamma} g(\theta)\exp(-j2\pi f\theta)\, d\theta} \tag{D3}$$

In the general case of an N-input/ L-output linear system the second order statistics of the input and the output signals are related through

$$E\langle b'(t)b(t+\psi)\rangle =$$

$$= \int_{\tau\in\Phi}\int_{\phi\in\Phi} g'(t-\tau)\; E\langle a'(\tau)a(\tau+\phi)\rangle\; g(t+\psi-\tau-\phi)\; d(\tau+\phi)\; d\tau$$

where $t,\ \psi \in \Psi$. The last expression relates the correlation functions of the input and the output signals given by

$$R_a(\phi,\tau) = E\langle a'(\tau)a(\tau+\phi)\rangle \quad \text{and} \quad R_b(\psi,t) = E\langle b'(t)b(t+\psi)\rangle.$$

At this stage the generality of the analysis is reduced to two types of linear systems which can be adopted in the line coder model for most cases of interest. In particular these are the filter and the interpolating filter which are defined through the relations between the domains of the input signal, the output signal and the characterising function in the following manner

$$\begin{aligned} \textit{filter}:&\ \ \Phi \equiv \Psi \equiv \Gamma \\ \textit{interpolating filter}:&\ \ \Psi \equiv \Gamma \supset \Phi \end{aligned}$$

The relation of the input and the output correlation has been examined in [Cariolaro, 1983]. Although in the general case the output is not stationary in spite of the stationarity of the input, the results show that for a filter and an interpolating filter the correlation of the input signal and that of the output signal are related through

$$R_b(\psi) = \int_{\Phi}\left(\int_{\Phi} g'(t-\tau)R_a(\phi)g(t+\psi-\tau-\phi)\; d\tau\right) d(t-\tau)$$

and

$$R_b(\psi) = \int_{\Phi}\left(\int_{\Psi} g'(t-\tau)R_a(\phi)g(t+\psi-\tau-\phi)\; d(t-\tau)\right) d\tau$$

respectively. Finally, by applying the Fourier transform to both sides of the last two expressions, the relation between the spectral density function of the output to that of the input in a linear system is given by

$$W_b(f) = G^{\mathrm{T}}(f)W_a(f)G(f) \tag{3.21}$$

where $G(f)$ is the Fourier transform of the characterising function of the system determined from $(D3)$ and $G^{\mathrm{T}}(f)$ is its conjugate transpose.

The process of deframing in the line coder model has been defined as a linear system. In particular – the conversion of the output blocks of symbols of period T into a sequence of symbols of period T_x is performed by an L-input/1-output interpolating filter. The time domains of the input and the output signals of the deframing system are $\Phi = \{0, \pm T, \pm 2T, \ldots\}$ and $\Psi = \{0, \pm T_x, \pm 2T_x, \ldots\}$ respectively. In the frequency domain the functional relation of this system is represented by the Fourier transform of the characterising function (3.2b) given by

$$\mathrm{G}_x(f) = \frac{1}{L}\begin{bmatrix} 1 \\ \exp(-j2\pi f T_x) \\ \exp(-\mathrm{j}2\pi 2T_x) \\ \vdots \\ \exp(-\mathrm{j}2\pi(L-1)T_x) \end{bmatrix} \tag{3.22}$$

This allows for a very convenient transition from the results obtained for the spectral density of the output-word sequence to that of the output-symbol sequence. By using (3.22) for the frequency response of the deframing system and (3.20a) in the left-hand side of (3.21), the continuous spectral density of the output-symbol sequence is determined as

$$w_x^{(c)}(f) = G_x^{\mathrm{T}}(f) W_X^{(c)}(f) G_x(f) \tag{3.23a}$$

In a similar way the discrete spectral density of the same sequence is found to be

$$w_x^{(\mathrm{d})}(f) = G_x^{\mathrm{T}}(f) W_X^{(d)}(f) G_x(f) = \sum_{r=-\infty}^{+\infty} G_x^{\mathrm{T}}(rF) V'_X \, V_X \, G_x(rF)\, \delta(f - rF) =$$

$$= \sum_{r=-\infty}^{+\infty} \left\| \frac{1}{L} \sum_{l=1}^{L} V_X(l) \exp\!\left(-j2\pi(l-1)\frac{r}{F}\right) \right\|^2 \delta(f - rF) \tag{3.23b}$$

where $W_X^{(d)}(f)$ and $G_x(f)$ are determined from (3.17) and (3.22) respectively.

The availability of the last two formulae is a significant achievement in the general spectral analysis of a broad class coders. The current presentation regards the case of discrete signals in particular but its generality can be extended straightforwardly to most types of random signals.

There is one last step before the determination of the complete expression for the Power Spectral Density (PSD) of the line signal. The spectral analysis so far provides means of evaluating the frequency distribution of a symbol sequence which is produced by fixed-length block coding of a random sequence of digits. The latter is assumed to be stationary and memoryless.

Finally it is possible to derive the expressions characterising the PSD of the line signal by relating the output of the digital modulator to its input which is presented in the frequency domain by (3.23a) and (3.23b). For this purpose it is essential to identify the type of system used for modulation. In many real line coders the digital modulator is adequately modelled as a linear system. This is so for a PAM which includes, for example, coders like those used in optical-fibre transmission systems based on direct light-intensity modulation.

By identifying the time domains of $x(rT)$ and $y(t)$ as $\Phi = \{0, \pm T_x, \pm 2T_x, \ldots\}$ and $\Psi \equiv \{\textit{the real numbers}\}$ respectively, the digital modulator can be described as an interpolating filter with a characterising function $g_y(*)$ defined as in (3.3). The frequency response of that function is given by

$$G_y(f) = \frac{1}{T_x} G_\gamma(f) \tag{3.24}$$

where $G_\gamma(f)$ is the Fourier transform of $\gamma(t - k_x T_x)$. This leads to the final form of the expressions evaluating the continuous and the discrete parts of the PSD of the line signal as:

$$Y^{(c)}(f) = G_y^{\mathrm{T}}(f) w_x^{(c)}(f) G_y(f) = \left\| G_y(f) \right\|^2 w_x^{(c)}(f) \tag{3.25a}$$

$$Y^{(d)}(f) = \sum_{r=-\infty}^{+\infty} \left\| G_y(rF) \frac{1}{L} \sum_{l=1}^{L} V_X(l) \exp\left(-j2\pi(l-1)\frac{r}{F}\right) \right\|^2 \delta(f - rF) \tag{3.25b}$$

3.2.4 A Summary of the Main Stages in Constructing the Analysis Model

The last two expressions, (3.25a) and (3.25b), comprise the complete functional representation of the coded-signal spectral density. The two variables $w_x^{(c)}$ and $V_X(l)$ can be substituted with the expressions (3.23a) and (3.15a), derived in the preceding stages of the model definition, but it is unnecessary to write the complete mathematical formulae in explicit form. The structure of the analysis model allows, by starting from the initial coder definition (the sets of symbols and coder states), the derivation of the final results to be traced through several stages where the intermediate expressions can be evaluated independently. Thus it is possible to identify the following main steps of the analysis:

1) Specification of the initial conditions through –

   a) the set of output matrices Z_m (3.5a);
   b) the set of state transition matrices S_m (3.5b);
   c) the input word probabilities Q_m (3.6).

2) Computation of the main probability variables as follows –

   a) the transitional probability matrix S from (3.7);
   b) the first order state-probabilities vector P, (3.8);
   c) the limiting TPM S_∞, (3.16).

3) Evaluation of the main variables for the first and second order statistics of the output word sequence –

   a) the mean value V_X, (3.15a);
   b) the components of the covariance $W_X(z)$, given by the expressions (3.19).

The expression for the continuous part of the spectral density $W_X^{(c)}$ of the coded word sequence accumulates the results from all essential stages as outlined above. This leads to the final two steps:

4) Evaluation of the spectral density (3.23a) of the output symbol sequence through the frequency response $G_x(f)$ of the deframing transfer function, (3.22).

5) Finally, the PSD of the coded line signal is computed by combining the results of step 4) with an appropriate specification of the frequency response $G_y(f)$ of the modulator transfer function, through the expressions (3.25).

The five steps identified above have been successfully implemented in a computational procedure which is described in the next section.

3.3 Software Implementation of the Spectral Analysis Model

The general model described in the previous section is a very powerful tool for performing spectral analysis on digital signals involved in the process of line coding. The main advantage of this model is its validity for any coder as long as the FSSM, modelling the transformation of the input words into output words, is adequately specified. This asserts the possibility to apply the present frequency analysis method to all existing schemes for fixed-length block-coding because their realisation implies that they are finite systems. Full utilisation of the generality and the precision of the mathematical model presented in section 3.2 can be accomplished through a suitable computer implementation. The software routine, described in the following subsection, has been devised to achieve this goal. Some preliminary considerations are given below as an introduction to the main parts of the program.

An important feature of the adopted theoretical model is the compactness and the relative simplicity of the matrix expressions evaluating the statistics of the output-word sequences (3.15) and the two parts of the spectral distribution (3.17) and (3.20). The elegancy of the mathematical representation is due to the convenient definition of the state transition and output-word matrices which contain the whole information about the input/state conditions generating a particular output.

The explicit definition of the components of the line coder model allows for a convenient software implementation of the analysis stages outlined in the summary at the end of the previous section. Thus the specification of the sets of input symbols $d = \{d_1, d_2, \ldots, d_\alpha\}$ and input blocks of symbols $D = \{D_1, D_2, \ldots, D_M\}$ provides for the evaluation of the statistics of the input sequences Q_m. The most essential part of the definition of the initial conditions for the computational procedure is the construction of the state transition matrices S_m and the output word matrices Z_m (3.5) which are derived from the set of coder states $s = \{s_1, s_2, \ldots, s_I\}$ and the output word set $X = \{X_1, X_2, \ldots, X_J\}$ on the basis of the table presentation of the coder rules[7]. Finally the evaluation of the statistics of the output sequences is performed and the spectral density distribution is computed by appropriate specification of the frequency response of the deframing and the modulating transfer functions, $G_x(f)$ and $G_y(f)$.

[7] The state transition diagram could be used, although the table form of coder specification has been preferred for ease of presentation.

The sets of variables and the functions mentioned in the previous paragraph require independent specification in the computing procedure. The main body of the software routine involves the evaluation of the overall statistics of the signals (stages 2 and 3 from the summary given in subsection 3.2.4) and computing the continuous and the discrete PSD functions in graphical form[8]. An additional feature of the procedure allows for parametric evaluation of the line signal spectra if a range of values for the input symbol probabilities q_u $(u = 1,\ldots,\alpha)$, is also specified.

The application of the line coder model for practical computation of the PSD of coded signals in digital transmission requires adequate programming to perform conventional matrix operations. An efficient software routine has been created within the Matlab environment (see Appendix). This is a high-performance interactive package for scientific and engineering numeric computation [386-Matlab, User's Guide, 1989]. The main advantage of working in its environment is the capability of the package to perform a great variety of mathematical operations over a comprehensive set of elements expressed in a straight forward matrix form. At the same time a sufficient level of freedom is provided to develop software modules based on most common programming principles.

3.3.1 Description of the PSD Computational Routine

The procedure of computing PSD is an application oriented tool. It can be used for calculation and analysis of the spectral distribution of signals produced by a digital line coder. The results are presented in graphical form and can be displayed as two- and three-dimensional plots revealing interesting features and tendencies in parametric simulations. The software routine can be accessed through either a C/Fortran program or as a specific Matlab file. The coder definition values can be assigned interactively or by direct modification of the respective file. The second method gives greater freedom to experiment with unconventional coder specifications.

The overall structure of the computing algorithm comprises three conventional parts – input of the initial values; main computational body; graphics and numeric output. The description, given in this section refers to direct modification of the initial values in the computing file for two main reasons:

1) This allows for more detailed presentation of the developed software.

[8] The respective numerical data may be saved, if required.

2) The parts of the algorithm, which would require greater cautiousness when being altered, are pointed out.

The first part of the software is where the parameters of the line-coder model are specified. The only parameters required for specifying the input signals are:

- the length of the input words N;
- the number of input words M and
- the probabilities of the input symbols q_u.

Based on these three values the software generates all possible binary words of N symbols assuming the input-symbol set is $d = \{0,\ 1\}$. If a number of symbols $\alpha \neq 2$ is required, their values have to be specified explicitly.

The main initial values are determined next. From the mathematical model of the coder it can be seen that the statistics of the output sequences depend on the probabilities of the input words Q_m. To compute these probabilities the complete set of input words should be specified and the following expression has to be evaluated:

$$Q_m = q_1^{N_{m1}}\, q_2^{N_{m2}} \ldots q_\alpha^{N_{m\alpha}}$$

where $q_1 = p\{d_1\}$, $q_2 = p\{d_2\}$, ... , $q_\alpha = p\{d_\alpha\}$ are the probabilities of the input symbols; $N_{m1}, N_{m2}, \ldots, N_{m\alpha}$ are the respective numbers of symbols $d_1, d_2, \ldots, d_\alpha$ in the m-th input word such that

$$\sum_{a=1}^{\alpha} N_{ma} = N \quad \text{and} \quad N_{ma} = 0,\ 1,\ \ldots,\ N$$

Although the software is easily adaptable for any number of input symbols (no changes are necessary to the main body), in its present form it computes the PSD for binary input only. This is done for three reasons:

- the line coding theory based on the transformation of binary digit streams into signals suitable for particular transmission media is adequate to most practical cases;

- the conventional and the new coding formats, dealt with in the present analysis, are applied to binary sequences;

- the computing algorithm is developed to support the investigation of the general binary presentation of digital line codes discussed in a later chapter.

The above considerations confine the structure of the simulation routine to two versions. The simpler one assumes equiprobable binary symbols, i.e. $q_0 = q_1 = 0.5$ are the probabilities of the input symbols 0 and 1 respectively. In this case the input-word probabilities are equal and their value is given by

$$Q_m = q_1^{N_{m1}} q_0^{N-N_{m1}} = \frac{1}{2^N}$$

for all possible binary words of length N. The second version evaluates the frequency response for different probabilities of the input symbols. This requires the whole cycle of computations to be executed as many times as the number of initial values of the probabilities for input 0 and 1. If the set of values for q_0 and q_1 is not specified explicitly as an argument of the outer-most loop in the program, the working version assumes $q_0 = 0.1$ through 0.9 as the range with an increment of 0.1, while $q_1 = 1 - q_0$. As a result of using parametric input probabilities 3-D plots of families of spectral distribution curves are produced. Their importance to the analysis is discussed in section 4.1.2.

The next stage in the specification of the initial values requires understanding of the relation between the table and the matrix presentation of the line coder model. Indeed it is quite sufficient to provide the necessary numeric data in matrix form only but it is much easier to translate the coder rules into the respective table presentation which uniquely specifies the line coder and contains all the necessary information about the functional relations between the input words, the output words and the set of states. These relations are worth a more detailed description as they will be used extensively in the forthcoming chapters, especially for the enhanced algorithm which is applied in generation of the PSD graphs for classes of line codes.

The table representing the general FSSM model is shown in Fig. 3.1. It consists of M rows and I columns, where M is the number of possible input words and I is the number of states of the system. Every row corresponds uniquely to an input word and every column represents uniquely a coder state. In the crossing of a row with a column the two symbols X_{mi} and s_{mj} are placed. They represent the output word and the next state of the coder which result from the current state being s_i and the current input word being D_m for $i, j = 1,\ldots, I$ and $m = 1,\ldots, M$. To illustrate the functional relations between the input/output symbols and the coder states the table presentation of a differential coding scheme is given as an example in Table 3.1 below. This code has been introduced as NRZ-M in Chapter 2. The sets of symbols and coder states[9] for the present example of NRZ-M code are defined as follows:

$d = \{0,\ 1\}$	the input symbol set; the number of input symbols is $\alpha = 2$;
$x = \{-1, +1\}$	the output symbol set; the number of output symbols $\beta = 2$;
$D = \{0,\ 1\}$	the input word set for $N = 1$; the number of input words is $M = \alpha^N = 2$;
$X = \{-1, +1\}$	the output word set for $L = 1$; the number of output words $J = \beta^L = 2$;
$s = \{s_1, s_2\}$	the set of coder states; the number of states is $I = 2$;

The last three sets are combined to produce the table form definition of the code which is shown below.

m	D_m	s_1	s_2
1	0	$-1,\ s_1$	$+1,\ s_2$
2	1	$+1,\ s_2$	$-1,\ s_1$

Table 3.1 The NRZ-M code specification

The functional relations between the sets involved in the Table 3.1 apply for this example in the following manner:

1) When the input word is 0 and the coder is in state s_1 the output word -1 is produced and the coder moves (remains) into state s_1.

2) . . . 3) . . .

4) When the input word is 1 and the coder is in state s_2 the output word -1 is produced the coder moves into state s_1.

The process of constructing the table for the NRZ-M code is not described at this point to avoid distraction from the presentation of the software routine. The table-form definitions of this and other codes are given in greater detail in the next chapter.

It is straight forward to derive the state-transition matrices S_m and the output-word matrices Z_m directly from the coder-table. The pair (S_m, Z_m) corresponds to the m-th word D_m and represents the information from the m-th

[9] The problem of how to determine the set of states is given a more detailed consideration in the next chapter.

row of the coder table. The matrix S_m is of size $I \times I$ and its rows correspond to the columns of the table representing the states s_i. Then 1 is assigned to that element $S_m(i,j)$ of every row of S_m whose column index j indicates the state which the system moves into, i.e. $\{s_i \rightarrow s_{mj}\} \Rightarrow \{S_m(i,j) = 1\}$. All other elements of the same row of S_m are 0. The matrix Z_m is a collection of the output words from the m-th row of the transition table so that every row of the matrix is one word. The rows are indexed $i = 1, \dots, I$, where i is the second index of the output words X_{mi}. The columns of the output matrices are indexed $1, \dots, L$, for output words of length L.

The same example of the differential coding scheme NRZ-M is used again to illustrate the formation of the state-transition and the output matrices. There are two matrices of each type, i.e. S_1, S_2 and Z_1, Z_2 and their numerical values are determined as follows:

$$S_1 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \quad S_2 = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}, \quad Z_1 = \begin{bmatrix} -1 \\ +1 \end{bmatrix}, \quad Z_2 = \begin{bmatrix} +1 \\ -1 \end{bmatrix}$$

The matrix S_1 corresponds to the first row of Table 3.1. The first row of S_1 corresponds to state s_1 and the first element of that row indicates that s_1 moves (remains) into s_1. . . . The second row of S_2 corresponds to state s_2 and the first element of that row indicates that s_2 moves into s_1. The matrix Z_1 represents input word $D_m = 0$ in Table 3.1 and its two rows correspond to the output words -1 and $+1$ produced for this input when the coder is in states s_1 and s_2 respectively. Similarly for the matrix Z_2.

Finally, it should be noted that, if the values of the two sets of matrices described above are made available they can be set in the respective file by directly modifying several lines of the program. The matrices are assigned to variables which are labelled S1, ... , SM and Z1, ... , ZM. For more routine applications of the developed computing algorithm an interactive mode may be used where the values of the elements are requested separately for every row of a matrix. In this case the number of states I is required additionally.

The main part of the software comprises four sections in which the computation of the matrix terms of the expressions evaluating the PSD as a function of the normalised frequency is performed. In the first section the TPM S is determined from the input-word probabilities Q_m. This allows for the vector P containing the first order probabilities of the state sequence to be evaluated next.

There are at least two possible ways of evaluating P by using the linear system of equations (3.8) from Section 3.2.2.1. The choice depends entirely on the computing environment and on the level of precision required. A relatively fast method is to calculate the product $P_{k+1} = P_k S$ iteratively, starting from conveniently chosen initial vector P_1, so that the sum of its components is one. (E.g. $P(i) = \frac{1}{I}$, for $i = 1, \ldots, I$.) The process is likely to converge in most practical cases of interest by virtue of definition of the TPM and the state-probabilities vector. Simple conditions can be specified to prevent from entering an endless loop. The speed[10] of getting a satisfactory result for P depends on the specified error tolerance ϵ which ends the iterations when $\epsilon > \| P_k - P_{k-1} \|$.

The above technique is applicable mostly when conventional and well studied codes are evaluated where the specification of proper TPM-s eliminates the risk of having a divergent series $P_k S$. The more reliable method of computing the vector P is to solve the system of equations directly. This can be achieved easily depending on the mathematical tools available in the computing environment. For the software algorithm presented here this operation is simplified by the use of predefined matrix division. The latter allows for computing the matrix $X = \mathrm{B}/\mathrm{A}$ as a solution to the equation $X{*}\mathrm{A} = \mathrm{B}$.

In this case the system of equations $P = PS$, augmented by the condition $\sum_{i=1}^{I} P(i) = 1$ is solved by applying the following matrix transformations:

$$PS - PU + P\mathcal{E} = u'$$

$$P = u'(S + \mathcal{E} - U)^{-1}$$

where U is the identity matrix, $u = [\,1,1,\ldots,1]'$ is a column vector of size I by 1 and $\mathcal{E}$ is a square matrix with all its elements equal to one. The size of $\mathcal{E}$ is I by I. The last expression gives a meaningful result, in terms of computing the state-probability vector, if $(S + \mathcal{E} - U)$ is non-singular. It should be noted that the condition $\sum_{i=1}^{I} P(i) = 1$ is implemented by using the equality $P\mathcal{E} = u'$. Also, in order to achieve matrix conformity in distributive multiplication, the identity $PU \equiv P$ has been used.

The next step in the main body of the computing algorithm evaluates the matrix components V_X, H_0, H_1 and H_2 comprising the final expressions for the continuous and the discrete parts of the PSD. These expressions are given by (3.19) and they do not depend on the frequency argument f. It should be pointed

[10] The main reason for this method to be relatively fast is that it does not require matrix division which involves the computation of adjoint matrices.

out that when the input symbol probabilities are not equal the evaluation of these matrices is achieved through accumulative summation over $m = 1,\ldots, M$ when the number of input words M is big.

Two separate loops comprise the last section in the main part of the algorithm. The continuous and the discrete components of the spectral distribution are evaluated over the argument f representing the normalised frequency. The option to interact with the computational routine exists at this point as well. It is possible to specify two parameters *nf* and *nFx* which determine the resolution and the frequency range of numerical results respectively. The first value determines the total number of equally spaced frequency samples for which the PSD is computed. The higher this number, the greater the accuracy of the frequency response. At the same time *nf* is proportional to the duration of the overall computing and its practical limit can be determined with respect to the resolution of the graphics output. The other value *nFx* determines the total length of the working frequency range as the number of output-symbol bit rate units. For example, if $nFx = 3$ and the normalised output-symbol rate is $Fx = 1$, then the spectral distribution is computed over the range of $f = 1$ to 3. The values of the discrete part of the PSD are determined in a similar way by calculating the magnitude of the spectral lines at multiples of the word rate F.

As has been mentioned already, the analysis presented here is mainly concerned with the frequency response produced by various line coding techniques. Practical systems may require implementation of suitable pulse shaping and filtering to achieve better results in reducing intersymbol interference and other performance degrading effects encountered in a particular application. The simplest pulse shaping modulator function has been used in the computing procedure, in order to achieve a unified approach in the assessment and the comparison of different line coding schemes. The frequency response of the basic rectangular pulse waveform, as derived in Section 1.2.3, is given by

$$G_\gamma(f) = T_x \frac{\sin(\pi f T_x)}{\pi f T_x} = T_x \operatorname{sinc}(\pi f T_x)$$

For the case of a pulse-amplitude digital modulator the frequency response of the transfer function given by (3.24) becomes

$$G_y(f) = \frac{1}{T_x} G_\gamma(f) = \operatorname{sinc}(\pi f T_x)$$

The software routine described here can be amended with very little effort to accommodate various expressions for the modulator spectral response.

Finally, the last part of the algorithm produces the graphical output of the computation as plots of the spectral density distribution for the output-symbol sequence and the coded line signal. An essential point regarding the type of the output is that 3-D graphs are produced only with parametrical analysis when different values of the input-symbol probabilities have been used. This will be discussed in greater detail in a subsequent chapter.

In summary, the software routine described in this section has been designed to accommodate the full power and the generality of the theoretical model of the line coder. High degree of flexibility and universality have been achieved through a modular structure which allows the spectral analysis technique to be applied to virtually all digital coding schemes with a fixed-length block structure. This provides a common basis for estimation and comparison of codes and their frequency characteristics. Another advantage of the software algorithm is the possibility to modify the initial specifications easily and to adapt the procedure to the requirements of any coding scheme as long as the initial conditions are correctly specified. The practical results from applying the computational procedure, which illustrate the power and the generality of the spectral analysis method are presented in the next chapter.

4. THE UNIFORM ANALYSIS AND DESIGN OF DIGITAL LINE CODES

The discussion of coding in digital transmission and especially of the problems related to line coding has revealed the importance of practical and accurate spectral analysis methods. It has been indicated (Chapter 2) that a general and systematic approach has rarely been applied to the design and analysis of line codes. Results in this field are often presented in an application-specific form. The absence of common structure in the definition of various codes still poses difficulties in applying general assessment and comparison techniques.

The main goal of line coding is to shape appropriately the spectrum of the transmitted signal. In most practical applications, this involves a number of specific requirements, the most essential of which are as follows:

- small low frequency components (eventually no d.c.);
- sufficient amount of timing information (high density of transitions);
- narrow signal bandwidth (small increase in transmission rate).

A number of secondary requirements, like multi-level signalling, ancillary channels etc., are also common. Many similar problems and solutions in the area of line coding have been regarded as completely different, simply because the particular designs and results have been presented with respect to a specific requirement, rather than from a general assessment point of view.

The lack of a systematic approach to the problems of line coding was recognised a long time ago, [18]. Most publications in this field[1] show that some authors still resort predominantly to the descriptive form of code definition. Clearly this contributes very little towards the ability to categorise the proposed coding structures and even less to the assessment and estimation of the results in comparison with existing schemes.

With only a few exceptions, like [Poo, 1981; Cattermole et al., 1984], the majority of publications in the area of line coding have failed to promote general analysis methods and possibilities of unified classification of line coding techniques. The spectral analysis model presented in Chapter 3 is an excellent basis for a considerable improvement in this respect.

The computational procedure described in the previous chapter has been developed to provide suitable and powerful implementation of the line coder model. The results from practical application of the software routine, as well as its

[1] References [4,31-33] are just a fraction of all which could be given as an example.

potential to facilitate further theoretical investigations are presented in this chapter.

The main task in constructing the programming algorithm has been to incorporate the generality and the level of precision provided by the mathematical model. The accomplishment of this task is demonstrated through a considerable variety of examples. Some of them are the conventional line coding schemes, given in Chapter 2. The detailed analysis of these codes is presented for several purposes:

1) To prove that the results of evaluating PSD, produced by the developed computational procedure are correct and successfully compare with the best published to date.

2) To demonstrate the capability of the software routine to provide detailed and highly informative numerical and graphical analysis results.

3) To show the advantages of the unified coder presentation method over the conventional descriptive definition of coding rules.

4) To provide the basis for the suggested new classification structure and for generalised comparative assessment of line coding schemes.

In Chapter 5 the power of the analysis method is fully revealed through the further development of the software routine into a simple and convenient designer tool for generating completely new coding structures. A classification of line coding schemes, based on this enhancement of the spectral analysis routine is also suggested. Finally, a new method for evaluation of the 'coding capacity' of a coder, based on a special definition of its 'operational space'[2], is briefly outlined in Chapter 6.

4.1 Comparative Analysis of Line-Coding Techniques

The results from the spectral analysis of coded digital signals are usually presented in the form of graphics. It is common to assess the frequency characteristics of different coding schemes on the basis of the shape of the signal spectra, as well as on relative changes of the magnitude of particular frequency components. Evaluation of real numeric parameters is required in the practical

[2] The idea of the Coder Operational Space has been introduced in section 3.2.1 regarding the FSSM presentation of the block coder.

design of a coding system, especially when the spectral characteristics of the channel are specified precisely, and also for small changes of the spectra, which are difficult to estimate through direct examination of the corresponding graphics.

Various coding techniques are analysed in this chapter. The spectral densities of the resulting digital signals are given mainly in graphics form. However, it should be noted that the accuracy of the mathematical model and the powerful computational environment, [15] provide for the availability of highest precision numerical data corresponding to every graphical result and accessible at any stage of the evaluation process. A few explanatory notes are made below to facilitate the proper reading of the graphics format.

The working frequency range for the spectral density function has been determined over the normalised argument $f_n = \frac{f}{F_x}$, where $f \in [f_0, f_{max}]$[3] and $F_x = \frac{1}{T_x}$ is the code symbol repetition rate. The time interval of a code symbol, T_x has been defined in the description of the coder model. It relates to the input/output word period, T and the duration of the input symbols, T_d as follows: $T_x = \frac{T}{L} = \frac{NT_d}{L}$. The integers N and L are the numbers of symbols in the input and the output words respectively. Their values are determined by the program from the specified sets of input and output symbols. Two parameters, labelled *nf* and *nFx*, are used in the analysis procedure to determine the range and the resolution of the computation in the following way:

nf determines the number of numeric values of the frequency argument for which the continuous part of the PSD, $Y^{(c)}(f)$ is calculated.

nFx determines the upper limit of the normalised frequency range with respect to the code symbol transmission rate, from the relation $nFx = \frac{f_{max}}{F_x} = (f_n)_{max}$.

Thus the resolution, which also defines the size of the increment for the normalised frequency, is given by $f_{n+1} - f_n = \frac{nFx}{nf}$, where $n = 1, \ldots, nf$. The default values for the two parameters are $nf = 100$ and $nFx = 1$. By assuming $f_0 = 0$ the analysis range becomes $f \in [0, F_x]$. The values of the normalised frequency, $f_n = 0, \ldots, nFx$ are used for the numeric notation along the frequency axis, mainly for the purposes of convenience in producing the graphics. The program allows for unlimited freedom of choice in selecting a frequency range and resolution of interest[4].

[3] f_0 and $f_{\max}$ are real numbers specifying the frequency analysis domain.

[4] It is possible to do this interactively, if preferred.

The meaning of the numerical values for the plots of the spectral density functions, $W_X(f)$ and $Y(f)$, can be illustrated through the opening examples of the analysis. Before applying the unified structure of code definition, the basic waveform format, NRZ (unipolar and polar) is used to specify the reference PSD for the subsequent analysis.

As it has been already mentioned, the NRZ pulse waveform can be viewed as transformation of the symbols 0 and 1 into output symbols 0 and 1 or -1 and $+1$ for unipolar or polar formats respectively. Subsequently the digital modulator uses positive or negative pulses for $+1$ or -1 respectively and zero-level (no pulse) for 0. Without going into too many details and by tracing the main computational stages in the coder model, some of the numerical values of the spectral density functions for the NRZ line format are shown in Table 4.1 below. Common for both types (unipolar and polar) is the possibility to view the coder as a single state system, which results in having equivalent values for the following variables (as defined in Chapter 3): $S = 1$; $P = 1$; $S^r = 1$; $S_\infty = 1$. The use of the basic modulator function with a frequency response $G_y(f) = \text{sinc}(\pi f T_x)$ for all analysis examples has been discussed in the previous chapter. In the case of NRZ $T_x = T = T_d$.

NRZ	Z_1, Z_2	V_x	H_0	$J(z), H_1, H_2$	$W_X^{(c)}(f)$	$G_x(f)$	$W_x^{(c)}(f)$	$Y^{(c)}(f)$
unipolar	0, 1	.5	.5	0	.25 T	1	.25T	$\frac{1}{4}T\text{sinc}(\pi f T)$
polar	-1, $+1$	0	1	0	T	1	T	$T\text{sinc}(\pi f T)$

Table 4.1 Computational results for the normalised PSD of the NRZ

The examples given above indicate that it is convenient to use normalised numerical values for the spectral density functions. Having assumed a code word repetition rate of unity, the PSD plots are given for unity period of word repetition. In the case of NRZ, for example, this leads to $W_X(f) = W_x(f) = 0.25$ or 1 and $Y_x(f) = 0.25\text{sinc}(\pi f)$ or $\text{sinc}(\pi f)$ for unipolar or polar signal respectively. In general, it should be noted that the 2D graphics shown in this chapter represent the spectral density functions $W_x(f)$ and $Y_x(f)$ of the output symbol sequence and the coded line signal respectively. The results from their numerical evaluation are normalised by the factor $F_x = FL$, where $F = \frac{1}{T}$ is the code word rate[5]. In other words the actual plots are of the functions

[5] The normalising factor derives from the multipliers T and $\frac{1}{L}$ in the expressions for $W_X^{(c)}(f)$ and $G_x(f)$ given in the definition of the coder model.

$$\frac{W_x(f)}{T_x} = \left(\frac{L}{T}\right) W_x(f) \text{ and } \frac{Y_x(f)}{T_x} = \left(\frac{L}{T}\right) Y_x(f)$$

Finally, the 3D plots, which are given for most of the spectral analysis examples in this chapter, represent evaluation of the function $Y_x^{(c)}(f)$ along the vertical, z axis over the range of frequencies, f and the various sets of input symbol probabilities, $q(0)$ and $q(1)$ along the horizontal, x and y axes respectively.

4.1.1 The Unified Description of 'Coder-Rules'

The common approach in the definition of fixed-length block coding schemes has already been mentioned in previous chapters. It derives from the spectral analysis model presented in Chapter 3. The suggested coder definition method is summarised in this section in its complete form. It is described in a suitable format to facilitate the practical application of the developed frequency analysis technique. The specification of a code is divided in two parts which are presented below in their general form.

1) Specification of the symbol sets

The presentation of a coding scheme usually starts with specification of the input and the output sets of symbols. Following the discussion in Chapter 2, about the variety of ways to describe the rules of existing line codes, it has been assumed that all 'encoding formats', 'line-wave formatting techniques', etc. can be represented as two separate functions: transformation of symbols and digital modulation. Therefore, the cases where line codes have been given as a substitution of a digital sequence with signal levels (high, low, positive, negative, etc.) are suitably redefined according to the components of the line coder model (Fig. 3.1). This is achieved through converting the most commonly used signal levels into digits, representing output symbols, according to Table 4.2.

A careful examination of the mathematical expressions in the line coder model shows that the shape of signal spectra does not depend on the numerical values of the output symbols. It can be proven that changes of the symbol values result only in scaling and offsetting the graphics without altering the functional relation with respect to the frequency argument.

level	symbol
high, positive	1
middle, zero	0
low	0 or −1
negative	−1

Table 4.2 Symbol presentation of signal levels

However, for the sake of uniformity of the analysis results, the binary set of symbols is assumed as $[0,1]$ or $[-1,+1]$ and the set of ternary symbols as $[-1,0,+1]$.

By using the notation adopted in the definition of the components in the coder model, the table shown in Fig.4.1 is introduced for the specification of the input and the output sets of symbols.

Symbols

INPUT		OUTPUT
$d = [d_1, d_2, \ldots, d_\alpha]$ $q = [q_1, q_2, \ldots, q_\alpha]$	Q_m	$x = [x_1, x_2, \ldots, x_\beta]$
$D = \begin{bmatrix} d_{11}\, d_{12} \ldots d_{1N} \\ d_{21}\, d_{22} \ldots d_{2N} \\ \vdots \;\; \vdots \; \ldots \; \vdots \\ d_{M1} d_{M2} \ldots d_{MN} \end{bmatrix} = \begin{bmatrix} D_1 \\ D_2 \\ \vdots \\ D_M \end{bmatrix}$	$\begin{matrix} Q_1 \\ Q_2 \\ \vdots \\ Q_M \end{matrix}$	$X = \begin{bmatrix} x_{11}\, x_{12} \ldots x_{1L} \\ x_{21}\, x_{22} \ldots x_{2L} \\ \vdots \;\; \vdots \; \ldots \; \vdots \\ x_{J1}\, x_{J2} \ldots x_{JL} \end{bmatrix} = \begin{bmatrix} X_1 \\ X_2 \\ \vdots \\ X_J \end{bmatrix}$
$d_{mn} \in d; m = 1{:}M, n = 1{:}N$		$x_{jl} \in x; j = 1{:}J, l = 1{:}L$

Fig. 4.1 The symbol specification table

The table from Fig. 4.1 will be generally referred to as the 'Symbols' specification. It corresponds to the representation of the symbol sets for the computational procedure described in Chapter 3. The software routine provides several possibilities for the specification of the input and output symbol sets. Some of them, which are most likely to be used are summarised below:

i) It is sufficient to specify only the length of the input and the output words (N and L), if both the input and the output sets are binary. When the input symbol probabilities are not given explicitly the program assumes they are equal and automatically produces the three sets[6] D, X and Q.

It should be noted that the set of output words, X at this stage contains all K possible binary blocks of length L. The J valid output words are selected in the next stage when the coder table is compiled.

ii) If the symbol sets are not binary (which is more likely for the output symbols), their values have to be specified explicitly in d and/or x. The program again computes the matrices D, X and the

[6] These sets in fact are specified as matrices exactly as shown in the table of Fig. 4.1.

code word probabilities Q assuming that the complete sets are required. When this is not the case the input and the output word matrices have to be constructed manually.

iii) The last of the likely specification possibilities occurs when it is not necessary to compute all symbol combinations for given block lengths N and L. In this case the matrices D and X have to be specified explicitly and particular values to be assigned to the input word probabilities Q, if required[7].

Further particulars about the initial set-up of the computational procedure can be found through a direct examination of the program and use of the software.

2) Specification of the code table

The table presentation of the coder has been defined in the presentation of the spectral analysis model (Chapter 3). A detailed description of the relation between the code table and the matrix presentation has been given in Section 3.3 where the main parts of the software routine have been introduced. The table presentation is shown in this section only for the purpose of completeness of the general coder specification and for developing a smooth transition to its practical implementation.

Clearly it is more or less straightforward to determine the sets of input and output words (D, X) for many codes of practical interest, including the examples given in Chapter 2. However, it is not always so with the set of coder states, s. The FSSM model implies an infinite number of equivalent system realisations of a particular code. The one which is of practical interest is the minimal FSSM and the System Theory provides methods to prove that it is unique within a given class of equivalences, as well as procedures allowing for the minimal machine to be determined from any equivalent FSSM. The theoretical model of the line coder does not provide a general method for deriving the set of states, although a useful technique is suggested in reference [18], which allows the states to be determined on the basis of an implementation scheme.

The problem of identification of the coder states is approached in several ways in the presented spectral analysis. For most of the examples with relatively simple coding rules the coder states are determined from purely practical considerations based on particular conditions and/or features of the digital

[7] When the values for the input word probabilities are not specified they are computed as usual, on the assumption of equiprobable input symbols.

sequences. The NRZ-M code can be used as an illustration. It is not difficult to show that, if the coder states are assumed to correspond to the sign of the last output symbol, i.e. $s_1 = `-'$ and $s_2 = `+'$, then the coder rules represented by Table 3.1 apply. It is now possible to appreciate that the dependence of a coded symbol on the previous output symbol for this differential coding scheme makes its definition, given in Chapter 2, easier to state as transformation of pairs of symbols, than as a description of level transitions.

In the cases of other examples of conventional or new codes examined and analysed in forthcoming sections the sets of coder states are given without detailed comment on the method of their definition. This is done to avoid too lengthy descriptions or too detailed specifications of line codes which are well known.

An alternative possibility is suggested in Section 4.2, which involves a new technique for deriving the set of coder states together with the specification of the coder rules in a table-ready form. At this stage it is assumed that the sets of states are available for any of the codes whose spectral analysis is presented in the following sections. Special comments are made in the cases where the definition of the states requires explanation. By suitable reference to the coder-rules descriptions it is possible to combine the three sets D, X and s so that for every pair of an input word and a state the respective output word and next state are found. This naturally leads to the table presentation of the code, which is given in Fig. 4.2 combined with the corresponding matrix presentation. The construction of the output and the state-transition matrices has been discussed in detail in the previous chapter.

Code

D_m, $[Q_m]$	s_1	…	s_I	S_m	Z_m
D_1, $[Q_1]$	X_{11} s_{11}		X_{1I} s_{1I}	$\mathrm{SM} = \begin{bmatrix} S_1 \\ \vdots \\ S_M \end{bmatrix}$	$\mathrm{ZM} = \begin{bmatrix} Z_1 \\ \vdots \\ Z_M \end{bmatrix}$
⋮	⋮	…	⋮		
D_M, $[Q_M]$	X_{M1} s_{M1}		X_{MI} s_{MI}		

Fig. 4.2 Specification of the code in table/matrix form

The 'Symbols' and 'Code' tables defined in their general form above completely specify a fixed length block-coding scheme. The usual table presentation of the FSSM model of a coder is augmented with the input word probabilities[8] $[Q_m]$ and its matrix presentation. Formally a code is uniquely

[8] The values Q_m are given in square brackets to indicate that specification is optional and the input words are assumed equiprobable if these values are not given explicitly.

specified by the three sets D, X and s and their interrelation, given through the table originally presented in Fig. 3.3. Indeed, the ability to specify the FSSM model of any coder in table form presumes the knowledge of all the information contained in the table from Fig. 4.1. Also, the output and the state-transition matrices derive entirely from the table presentation of the coder, therefore they convey the information from the coder table, only in a different form.

The use of the two types of tables has been introduced mainly for the purposes of completeness in the suggested uniform specification of codes and convenience in the assessment and the practical analysis of their spectral characteristics. Finally, it should be noted that the matrices SM and ZM, collecting the output and the transitional matrices are implemented directly in the software routine as an efficient way of computing the cumulative summation over the index of the input words, m.

Before the systematic presentation of the frequency analysis results in the next section an example is shown below to illustrate the general coder specification and the graphics output produced by the computational procedure.

3B2T-RBS Symbols

INPUT		OUTPUT	
$d = [0,1]$ (word probabilities specified directly)	Q_m	$x = [0,1]$	
$D_1 = 0\ 0\ 0$	.0938	$X_1 = 0001$	$X_9 = 1001$
$D_2 = 0\ 0\ 1$	.1406	$X_2 = 0010$	$X_{10} = 1010$
$D_3 = 0\ 1\ 0$	.0938	$X_3 = 0011$	$X_{11} = 1011$
$D_4 = 0\ 1\ 1$	.1406	$X_4 = 0100$	$X_{12} = 1100$
$D_5 = 1\ 0\ 0$	.1406	$X_5 = 0101$	$X_{13} = 1101$
$D_6 = 1\ 0\ 1$	.0938	$X_6 = 0110$	$X_{14} = 1110$
$D_7 = 1\ 1\ 0$	.1406	$X_7 = 0111$	$X_{15} = 0000$(not used)
$D_8 = 1\ 1\ 1$	.0938	$X_8 = 1000$	$X_{16} = 1111$(not used)
$N = 3,\ M = 8$		$L = 4,\ J = 14,\ K = 16$	

Table 4.3a

This code is based on a description given in [26], where it is presented as a method of Relative Bipulse Signalling (RBS) which transforms a 3B2T ternary line code into a two-level format (3B2T-RBS). Table 4.3a proves to be a most adequate specification of the code in a binary form and the spectral analysis is performed straightforwardly by using the 3B2T-RBS definition from Table 4.3b.

3B2T-RBS Code

D_m	s_1	s_2	s_3	s_4	S_m	Z_m
000	1001, s_3	1001, s_4	0110, s_1	0110, s_2	$S_1=\begin{bmatrix}0010\\0001\\1000\\0100\end{bmatrix}, S_2=\begin{bmatrix}0100\\0010\\0100\\0010\end{bmatrix}$	$Z_1=\begin{bmatrix}1001\\1001\\0110\\0110\end{bmatrix}, Z_2=\begin{bmatrix}1110\\0001\\1110\\0001\end{bmatrix}$
001	1110, s_2	0001, s_3	1110, s_2	0001, s_3		
010	1011, s_4	1000, s_1	0111, s_4	0100, s_1	$S_3=\begin{bmatrix}0001\\1000\\0001\\1000\end{bmatrix}, S_4=\begin{bmatrix}1000\\0100\\0010\\0001\end{bmatrix}$	$Z_3=\begin{bmatrix}1011\\1000\\0111\\0100\end{bmatrix}, Z_4=\begin{bmatrix}0110\\0110\\1001\\1001\end{bmatrix}$
011	0110, s_1	0110, s_2	1001, s_3	1001, s_4		
100	1100, s_1	0011, s_4	1100, s_1	0011, s_4	$S_5=\begin{bmatrix}1000\\0001\\1000\\0001\end{bmatrix}, \quad S_6=S_3$	$Z_5=\begin{bmatrix}1100\\0011\\1100\\0011\end{bmatrix}, Z_6=\begin{bmatrix}1101\\0010\\1101\\0010\end{bmatrix}$
101	1101, s_4	0010, s_1	1101, s_4	0010, s_1		
110	0111, s_4	0100, s_1	1011, s_4	1000, s_1		
111	0101, s_3	0101, s_4	1010, s_1	1010, s_2	$S_7=S_3\,, \quad S_8=S_1$	$Z_7=\begin{bmatrix}0111\\0100\\1011\\1000\end{bmatrix}, Z_8=\begin{bmatrix}0101\\0101\\1010\\1010\end{bmatrix}$

Table 4.3b

The computational matrices SM and ZM are given below only to avoid drawing an excessively large table.

$$\mathrm{SM}=\begin{bmatrix}S_1\\S_2\\S_3\\S_4\\S_5\\S_6\\S_7\\S_8\end{bmatrix} \qquad \mathrm{ZM}=\begin{bmatrix}Z_1\\Z_2\\Z_3\\Z_4\\Z_5\\Z_6\\Z_7\\Z_8\end{bmatrix}$$

More detailed discussion of the 3B2T-RBS code is given later in this chapter. At this point the illustration of the spectral analysis procedure is completed with the graphical presentation of the results, which are typical for most cases of PSD computation. The plots in Fig. 4.3a represent the discrete and the continuous parts of the spectral density for the specified version of the 3B2T-RBS code. The PSD of the coded sequence of symbols and the line signal[9] are given as $w^{(c)}$ and $y^{(c)}$, respectively. The discrete frequency components (spectral lines) are shown as vertical bars, $y^{(d)}$ for better readability. For the example of the 3B2T-RBS code, in particular, $y^{(d)}$ indicates the presence of a d.c. component. As has been explained earlier in this chapter, the graphical results represent the PSD computed over the range of normalised frequencies, where $f_n = 1$ corresponds to the code-symbol repetition rate, $f = F_x$.

[9] PAM with a rectangular basic pulse shape is assumed, as defined in Chapter 3.

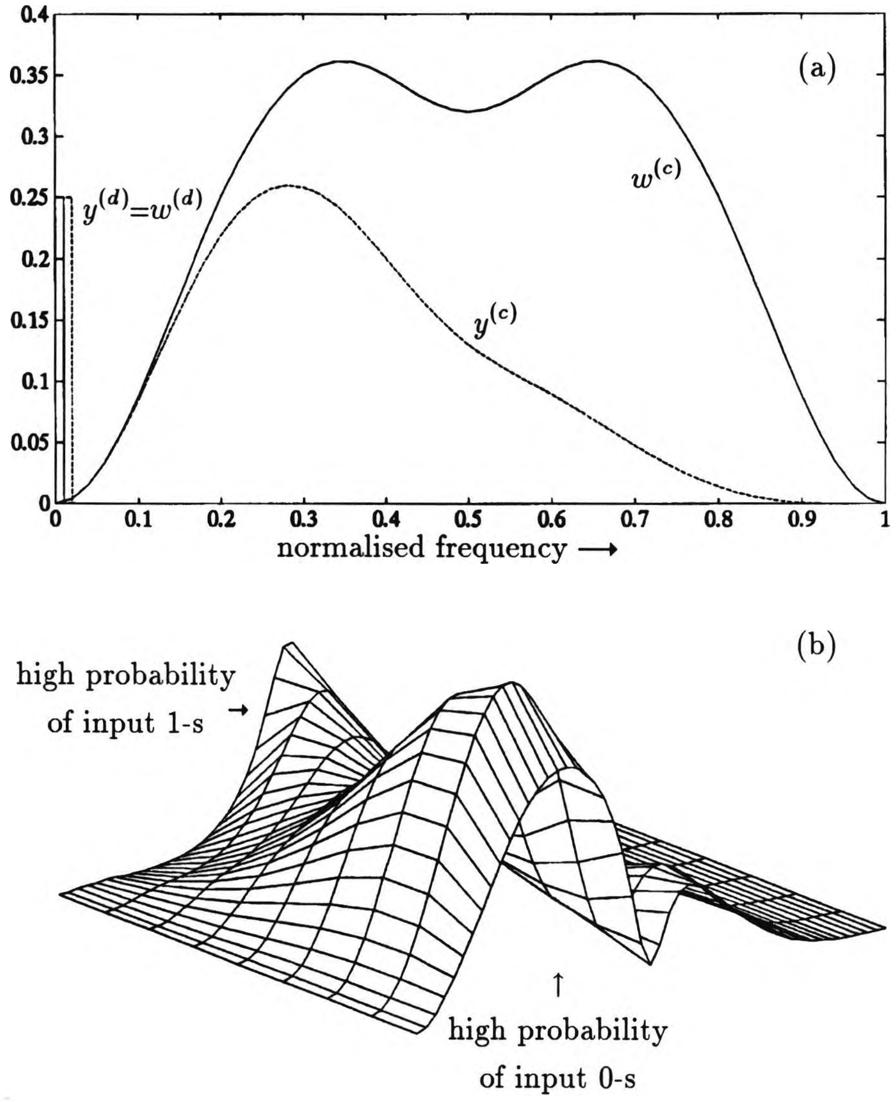


Fig. 4.3 Conventional (a) and parametrical (b) PSD plots of the 3B2T-RSB code

The 3D plot of Fig. 4.3b represents a family of spectral density curves of the line signal for ten sets of values of the input symbol probabilities, $q=[q_0, 1-q_0]$, where $q_0=0.3,\ 0.35,\ \dots\ ,\ 0.75$. It shows the expected shift of the signal power from the low frequency range for high probability of zeros in the input sequence towards the range of higher frequencies for high probability of input ones. The example clearly illustrates the possibility to evaluate the changes in the signal spectrum produced by variations of the input symbol probabilities. The parametrical PSD plots can be used to predict the deterioration of the communication channel performance for signals with large statistical variations, like real time transmission of speech or other burst type of information. More comments are given for the examples to follow in the sections below.

4.1.2 Practical Implementation of the General Spectral Analysis Model

The results from complete frequency analysis of digital line codes are presented in this section. The examples are mainly of existing coding techniques including most of the popular 'formats', already discussed in Chapter 2. Some codes, which are not amongst the popular ones are also analysed to demonstrate specific features which have not been exhibited by better known examples.

An important goal in producing the analysis results shown below has been to compile a comprehensive basis for general assessment of coding techniques. An initial step in this direction is the precise computation of the spectral density plots through the general coder model for many known codes. Only a few of them are usually given in the literature sources. Often PSD plots of the same code may differ considerably due to the various methods for their computation. In addition, the power and the flexibility of the computational algorithm have provided more information to be made available through a variety of parametric simulations. The latter have been presented as 3D spectral density plots, which are convenient for comparative assessment.

4.1.2.1 One-State Coding Schemes

The NRZ transmission format has been used twice so far: first to introduce the examples of conventional code description methods in Chapter 2 and second to specify the numerical reference basis in the presentation of the spectral analysis results (at the beginning of Section 4.1). The appropriate table form specification of the Unipolar and Polar NRZ formats, formally presented as 'coding' techniques, and the respective PSD graphics are given next in order to initiate the discussion of the results from the unified frequency analysis of various digital codes.

NRZ - Symbols

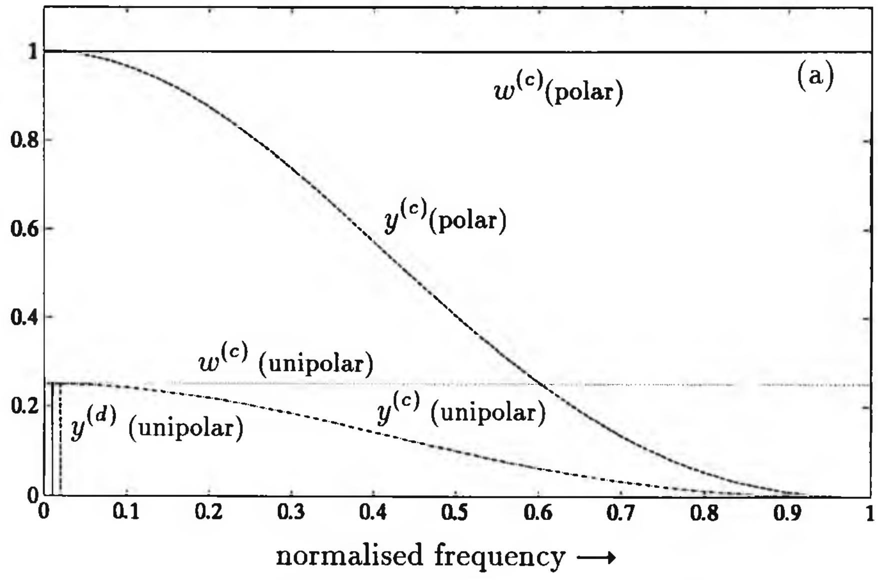
INPUT		OUTPUT	
$d = [0,\ 1]$ $q = [q_0, 1 - q_0]$ $q_0 = 0.1 - 0.9$	Q_m step $= 0.1$	polar $x = [-1, +1]$	unipolar $x = [0,\ 1]$
$D_1 = 0$	$0.1 - 0.9$	$X_1 = -1$	$X_1 = 0$
$D_2 = 1$	$0.9 - 0.1$	$X_2 = +1$	$X_2 = 1$
$N = 1, M = 2$		$L = 1, J = K = 2$	

Table 4.4a

NRZ - Code

D_m	polar			unipolar		
	s_1	S_m	Z_m	s_1	S_m	Z_m
0 1	$-1 \quad s_1$ $+1 \quad s_1$	$\text{SM} = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$	$\text{ZM} = \begin{bmatrix} -1 \\ +1 \end{bmatrix}$	$0 \quad s_1$ $1 \quad s_1$	$\text{SM} = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$	$\text{ZM} = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$

Table 4.4b



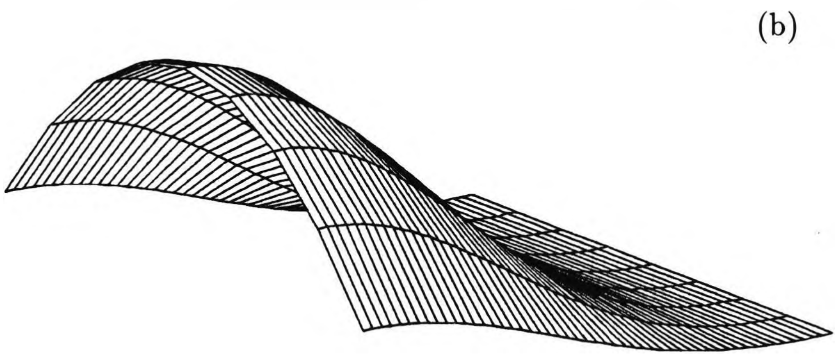


Fig. 4.4 Conventional (a) and parametrical (b) PSD plots of the NRZ code

The specification of the code denoted as NRZ in Tables 4.4a and 4.4b includes the parametric values for the input probabilities. It is not necessary to present and discuss separately the trivial case of the spectral density of a binary sequence of rectangular (non-return to zero) pulses with q = [0.5, 0.5]. The PSD graphs corresponding to this case are given in Fig. 4.4a only to illustrate the numerical results from Table 4.1. As expected $\left(y^{(c)}(0)\,/\,T_x\right)$ equals 1 for polar NRZ and 0.25 for unipolar. Another obvious result is $w_x^{(c)} =$ constant which is exactly the spectral density of a random sequence of delta functions representing the output symbols[10] before modulation with the rectangular pulse waveform. This results in the well-known 'sinc' function for the PSD of the NRZ 'coded' line signal, $y^{(c)}$.

The input probabilities are given in the specification tables of the above examples mainly for completeness and also to indicate that their values may be essential to the shape of the signal spectra. However, all spectral analysis results presented below are based on the two cases, equiprobable input symbols or parametric evaluation[11] of the PSD over the range of probabilities 0.05, 0.15, ... , 0.95 (except when explicitly specified otherwise), corresponding to the 2D and 3D plots respectively. For this reason q and Q_m will be omitted in the coder tables of the following examples, while remembering their significance for every case where the input probabilities have special values. Fig. 4.4b is an example of a 3D plot, representing the family of NRZ curves for variable input symbol probabilities.

The spectral analysis of other line codes is presented below in a certain order which provides a preliminary idea about the general classification structure defined completely in Chapter 5. When the first examples of line coding techniques were initially described, an order, according to code complexity, was suggested. This feature on its own is clearly insufficient for a stricter classification as it depends on several components of the code specification. Without going into further details at this stage, it is assumed that within the suggested ordering, complexity, due to higher number of code words, comes before the complexity resulting from higher number of coder states.

RZ - Symbols

input	output
$d = [0,1]$	$x = [0,1]$
$D_1 = 0$ $D_2 = 1$	$X_1 = 0\ 0$ $X_2 = 1\ 0$
$N = 1, M = 2$	$L = 2, J = 2$

a)

RZ - Code

D_m	s_1	S_m	Z_m
0	0 0 s_1	1	0 0
1	1 0 s_1	1	1 0

Table 4.5 b)

By keeping the number of states equal to one, as in the opening example, it is possible find a variety of codes with a different number of input and output words. The simplest case being the code equivalent to the RZ signalling format whose specification is shown in Tables 4.5a,b. The results from the spectral analysis of this code are identical to those of the NRZ with the essential remark that $(F_x)_{\mathrm{RZ}} = 2(F_x)_{\mathrm{NRZ}}$.

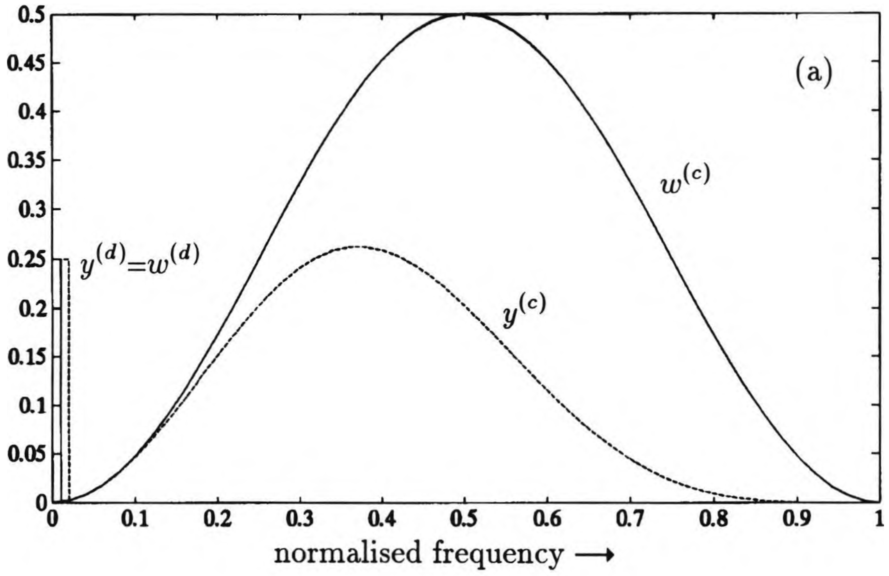
The Manchester code is another popular technique which does not require memory (the coder has one state only).

[10] The output sequence is identical to the input random sequence of symbols.

[11] For these cases q_0 denotes the probability of a binary input 0. The probability of 1 is $q_1 = 1 - q_0$ as indicated in the 'NRZ-Symbols' table.

Its specification tables are the same as those for the RZ signalling format with the exception of the first code word which is $X_1 = 01$. Obviously, just like RZ, the Manchester code requires twice the transmission bandwidth of the original symbol sequence. Unlike RZ, the Manchester code introduces at least one transition for every input symbol interval.

Also, it is easy to see that the longest interval without a transition is equal to the period of the input symbols. The twofold increase of the information content and the signal bandwidth allows for elimination of the low frequency components. This result is shown in Fig. 4.5 below.



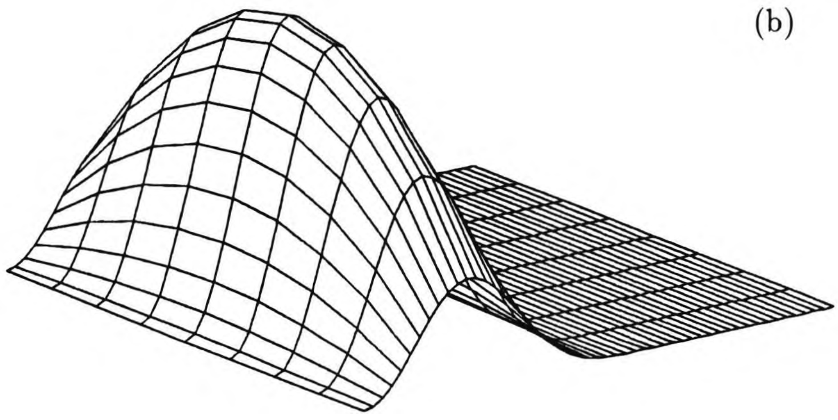


Fig. 4.5 Conventional (a) and parametrical (b) PSD plots of the Manchester code

One-state codes of higher complexity are sometimes used to achieve features, which are not directly related to shaping the signal spectrum. Codes from this group are usually a simple one-to-one transformation of a set of input words into a set of output words. The complexity of such codes is directly proportional to the number of input words, M. A typical example is the mB1C group of codes, [42] which insert an $(m+1)$th bit, complementary to one of the bits in a block of m input symbols. This technique is used to provide a minimum number of transitions in the coded signal and a very basic error monitoring capability. Its only advantage is the simplicity of the coding rules. No special provision is made to achieve a certain shape of the PSD function, which in some cases may be affected considerably, as is shown in Fig. 4.6 for $m = 3$ and $m = 5$.

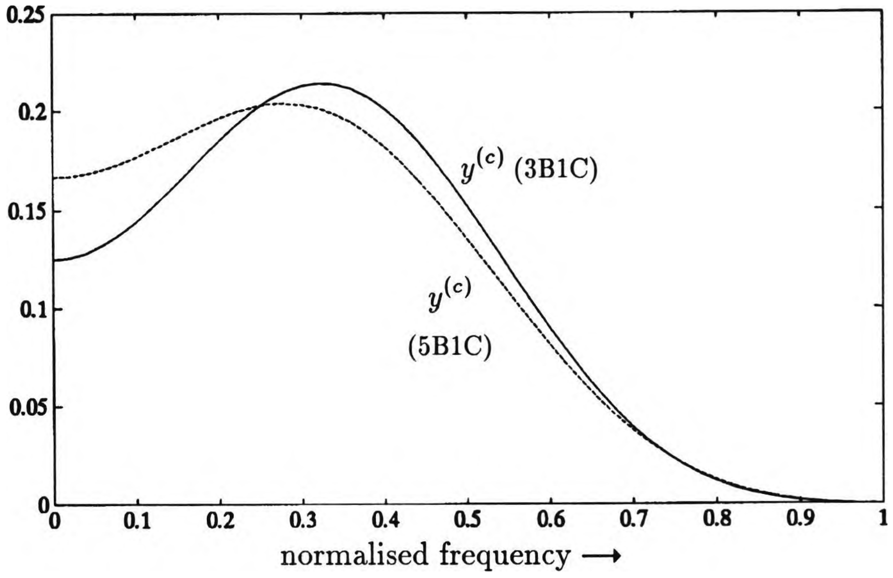


Fig. 4.6 The spectra of the 3B1C and 5B1C codes for equiprobable input symbols

Most of the coding schemes which can be modelled as a one state system are fairly easy to analyse. Very few of them can be implemented as an efficient solution to the problems of line coding. These types of codes are mainly used for the purposes of conversion between signals with a different number of levels. Certain spectrum shaping with a one-state coder can be achieved through the introduction of redundancy by using a subset of the possible output words, whose total number is larger than that of the input words. In most cases this leads to low efficiency. An alternative approach in producing a specific spectral density function is to implement systems with memory (i.e. the coder has more than one state).

4.1.2.2 Two-State Binary Coding Schemes

The presentation of the analysis results continues with examples from the next level of complexity. The codes from this group are modelled as a two-state FSSM and the number of possible output words is two. The Differential line formats provide basic examples of such coding techniques. Some of them have been introduced in Chapter 2 as NRZ-S and NRZ-M codes. The specification of the latter is given Tables 4.6(a,b) below followed by the plots of the respective PSD functions[12] in Fig. 4.7.

Differential - Symbols

input	output
$d = [0, 1]$	$x = [0, 1]$
$D_1 = 0$ $D_2 = 1$	$X_1 = 0$ $X_2 = 1$
$N = 1, M = 2$	$L = 1, J = 2$

Table 4.6a

Differential - Code

D_m	s_1		s_2		S_m	Z_m
0	0	s_1	1	s_2	$\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$	$\begin{bmatrix} 0 \\ 1 \end{bmatrix}$
1	1	s_2	0	s_1	$\begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$	$\begin{bmatrix} 1 \\ 0 \end{bmatrix}$

Table 4.6b

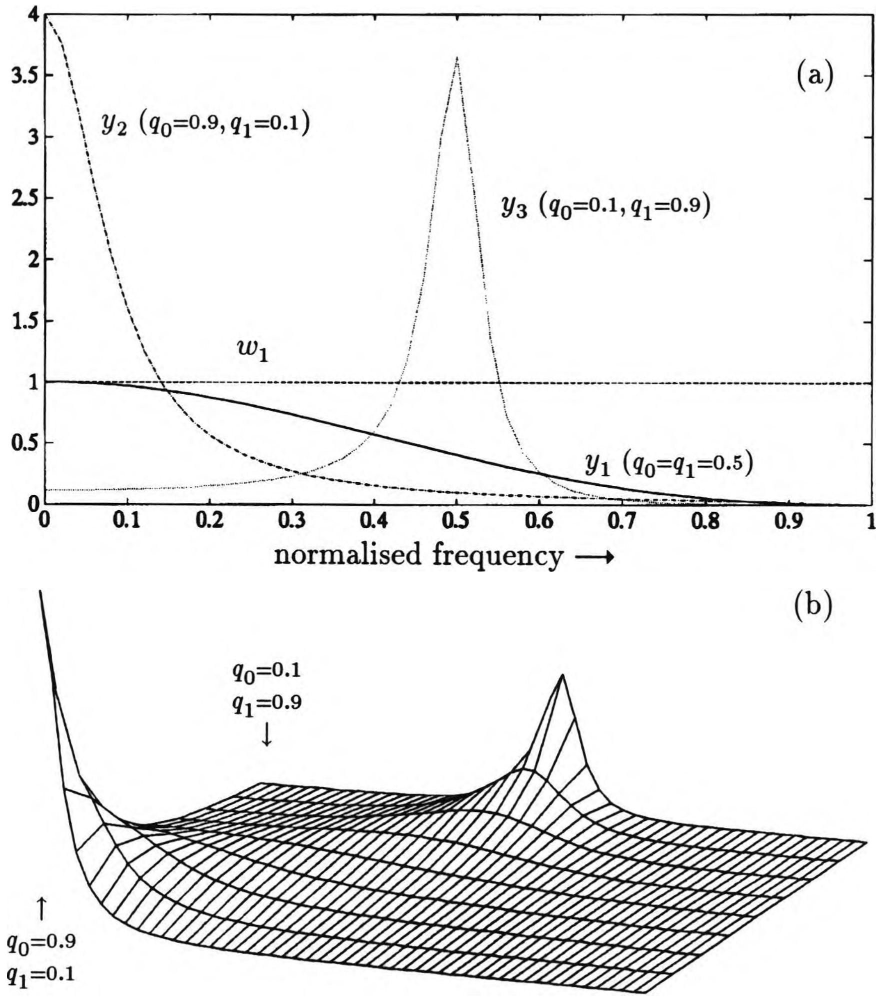


Fig. 4.7 Conventional (a) and parametrical (b) PSD plots of the Differential code

[12] The numerical data and the graphics correspond to NRZ-M and are identical to those of NRZ-S when a few trivial inversions are appropriately applied.

The two plots illustrate the frequency characteristics of the NRZ-M signalling format in a much more comprehensive way than is usually done in the literature. The spectral density graph y_1 in Fig. 4.7a confirms that for equiprobable input symbols the Differential code is equivalent to the basic NRZ format. The other two plots, y_2 and y_3 correspond to input probabilities of $q_0 = 0.9$ and $q_0 = 0.1$ respectively. The significant features of the code for unequal input probabilities are easy to observe from the parametrical 3D plot (Fig. 4.7b). The NRZ-M scheme introduces transitions for sequences of consecutive 1-s. This is why, when input 0-s are more likely to occur than input 1-s (indicated with the respective probabilities in Fig. 4.7b), the signal energy is distributed over the range of lower frequencies. For probabilities of input ones greater than the probabilities of input zeros (indicated with the respective probabilities in Fig. 4.7b), the magnitude of the lower frequencies is reduced and the spectrum is shifted towards the higher frequencies.

Obviously these results agree entirely with the description given in Chapter 2. However, it can be seen that the parametrical plot itself is sufficiently informative and provides for convenient and direct comparison with the spectral characteristics of other codes, especially when they are analysed through the same theoretical model.

The number of codes with two states and two possible output words is relatively small. Those which can be used for practical modification of the signal spectrum require state-dependent decoding. These problems are discussed in greater detail in the next chapter, where the general category of this type of coding is summarised. The presentation of the spectral analysis continues with codes which still have a two-state model but various numbers of possible output words.

4.1.2.3 Two-State Ternary Codes

Next to be shown is the spectral density of a classic example, the AMI line code. The specification of this code is given in Tables 4.7(a and b):

AMI - Symbols

input	output
$d = [0,1]$	$x = [-1, 0, +1]$
$D_1 = 0$ $D_2 = 1$	$X_1 = -1$ $X_2 = 0$ $X_3 = +1$
$N = 1, M = 2$	$L = 1, K = 3$

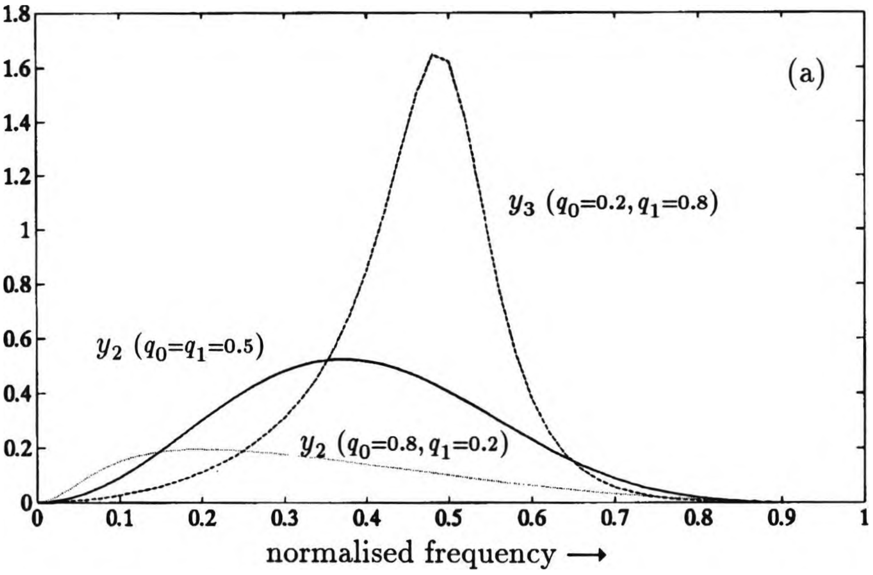
Table 4.7a

AMI - Code

D_m	s_1	s_2	S_m	Z_m
0	$0 \quad s_1$	$0 \quad s_2$	$\begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 1 \\ 1 & 0 \end{bmatrix}$	$\begin{bmatrix} 0 \\ 0 \\ -1 \\ +1 \end{bmatrix}$
1	$-1 \quad s_2$	$+1 \quad s_1$		

Table 4.7b

The spectral density plots of the output signal, for equiprobable input symbols are shown as y_1 in Fig. 4.8a. This is a well known spectrum and does not require any special comment. In addition the two PSD graphs, for input probabilities $[q_0 = 0.8, q_1 = 0.2]$ and $[q_0 = 0.2, q_1 = 0.8]$ are given in Fig. 4.8a as y_2 and y_3, respectively, to indicate the sensitivity of the AMI spectrum to changes in the balance of the input symbol distribution.



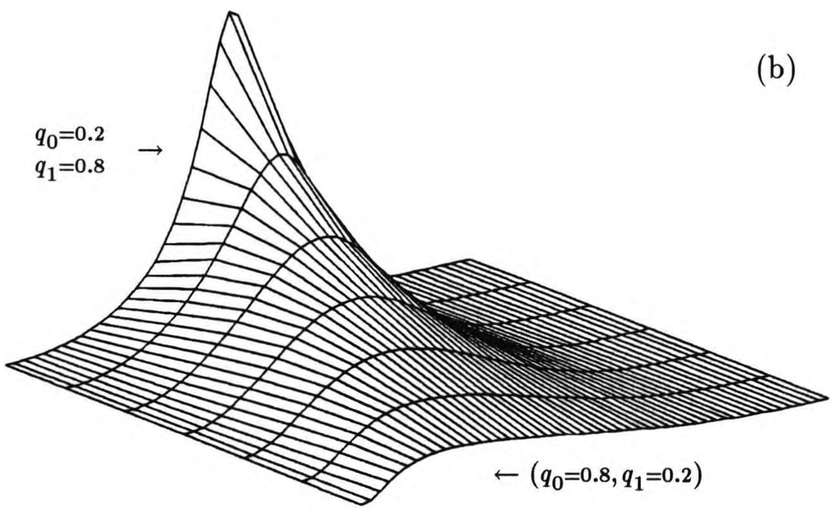


Fig. 4.8 Conventional (a) and parametrical (b) PSD plots of the AMI code

This is another case where the variations of the PSD function with the statistics of the input symbols are not difficult to predict. However, the 3D plot in Fig. 4.8b provides a good illustration of the balance features of the code. For probabilities of input 1, greater than the probabilities of input 0, the signal energy moves to the range of higher frequencies, while the spectrum becomes narrower (around the frequency $f = 0.5F_x$) as the number of transitions is higher due to the alternating output symbols. In the inverse probabilities case, the signal energy is reduced as there are more zeros in the output sequence. The lower-frequency components increase as the transitions are less probable, while the spectral density function still decreases towards $y^{(c)}(0) = 0$ as the frequency approaches zero. This shows that the code always retains its overall balance.

The Duobinary code is another typical example based on the same symbol set. There is a very important difference between this scheme and the AMI code. While the latter is uniquely decodable, the Duobinary is not, as the output zero may represent both input symbols, 0 and 1. This is evident from its specification in Tables 4.8(a and b), which give the complete definition of the Duobinary code.

Duobinary - Symbols

input	output
$d = [1,0]$	$x = [-1,0,+1]$
$D_1 = 0$ $D_2 = 1$	$X_1 = -1$ $X_2 = 0$ $X_3 = +1$
$N = 1, M = 2$	$L = 1, K = 3$

Table 4.8a

Duobinary - Code

D_m	s_1	s_2	S_m	Z_m
0	0 s_2	-1 s_2	$\begin{bmatrix} 0 & 1 \\ 0 & 1 \\ 1 & 0 \\ 1 & 0 \end{bmatrix}$	$\begin{bmatrix} 0 \\ -1 \\ +1 \\ 0 \end{bmatrix}$
1	+1 s_1	0 s_1		

Table 4.8b

At this point it is possible to further appreciate the advantages of the uniform code definition. Unlike the conventional descriptions of the coding rules, the 'common language' of the table presentation shows exactly what are the most essential similarities and differences between the last two techniques. The loss of a unique-decodability feature with the Duobinary scheme (which can be viewed as moving towards a higher level of complexity) has allowed a substantial achievement – twofold reduction of the transmission bandwidth, (Fig. 4.9). At the same time this reasoning can be further applied to suggest that the increase in complexity has not been sufficient to provide for the same achievement and still preserve the low frequency features of the AMI code. A tendency starts to become noticeable. It is related to several factors which are easy to point out due to the method of unified code specification. The first is the ratio between the number of output and input words, $\frac{M}{J}$ and the second is the number of coder states I. These two factors determine significantly the ability to modify the signal spectrum.

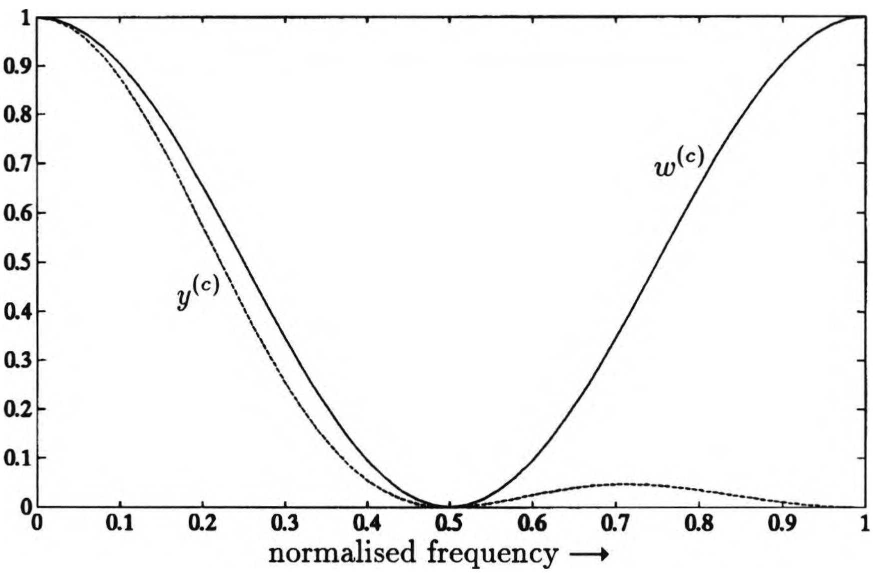


Fig.4.9 The spectrum of the Duobinary code for equiprobable input symbols

This ability is proportional to the number of valid and different spectral density functions for a given set of integers $\{M, I, K\}$, which correspond to the numbers of input words, states and output words respectively. The variety of possible modifications of the PSD will receive further attention in the following chapter with respect to the general classification structures.

The CMI code, which is specified next, is a frequently used technique. It has been described in Chapter 2 as a version of the biphase signalling format. The descriptive definition of this technique is another example of a standardisation effort without adopting sufficiently generalised basis (Fascicle III.3 – Recommendation G.703 of CCITT). This code is an interesting combination of features from both Manchester and AMI codes. The relation with the former comes through the common complete set of output words, $[X_1 = 00,\ X_2 = 01,\ X_3 = 10,\ X_4 = 11]$. Manchester code uses X_2 and X_3, while CMI uses X_1, X_2 and X_4. On the other hand CMI is related to AMI through the rule of 'code word alternation'. According to the preliminary classification arguments, mentioned with respect to the code complexity, the CMI is regarded in the category of the two-state codes. The brief assessment given in this paragraph can be completed by a straightforward comparison of the table specifications of the codes and the respective spectral analysis results. Those of CMI are shown in Tables 4.9(a,b) and Fig. 4.10.

CMI - Symbols

input	output
$d = [0,1]$	$x = [0,1]$
$D_1 = 0$ $D_2 = 1$	$X_1 = 0\ 0$ $X_2 = 0\ 1$ $X_3 = 1\ 1$
$N = 1, M = 2$	$L = 2, K = 4, J = 3$

Table 4.9a

CMI - Code

D_m	s_1	s_2	S_m	Z_m
0	0 1 $\quad s_1$	0 1 $\quad s_2$	$\begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 1 \\ 1 & 0 \end{bmatrix}$	$\begin{bmatrix} 0 & 1 \\ 0 & 1 \\ 0 & 0 \\ 1 & 1 \end{bmatrix}$
1	0 0 $\quad s_2$	1 1 $\quad s_1$		

Table 4.9b

The expected consequences from combining features of the Manchester and the AMI codes into the CMI are based on the following correspondence between structural and spectral features:

FEATURES		
<u>structural</u>		<u>spectral</u>
alternation rule	$\leftrightarrow$	small low frequency component
two transitions per input symbol interval	$\leftrightarrow$	double the original bandwidth
use of 00 and 11	$\leftrightarrow$	shift of spectral density towards the lower frequencies

The modifications of the frequency characteristics of a CMI coded signal, as summarised above, are clearly illustrated with the spectral analysis results shown in Fig. 4.10a,b.

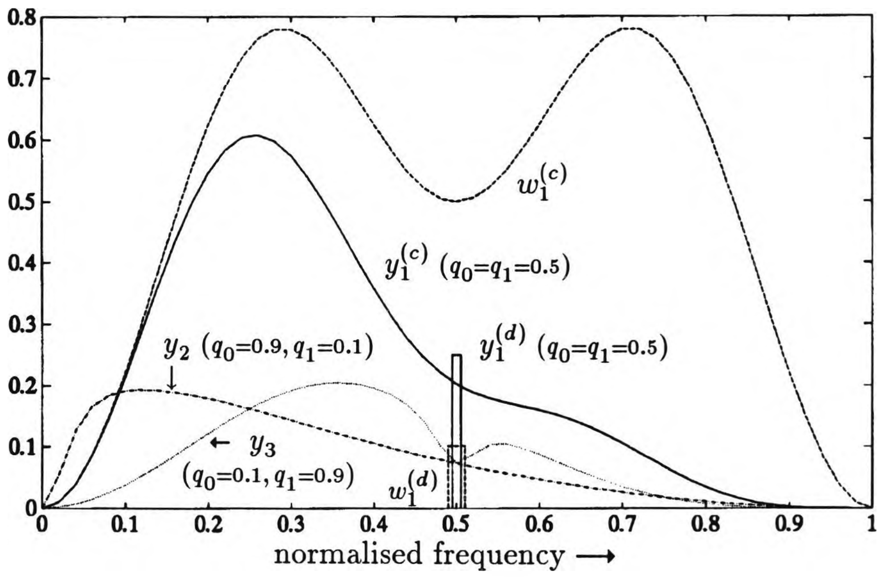


Fig. 4.10a Conventional PSD plot of the CMI code

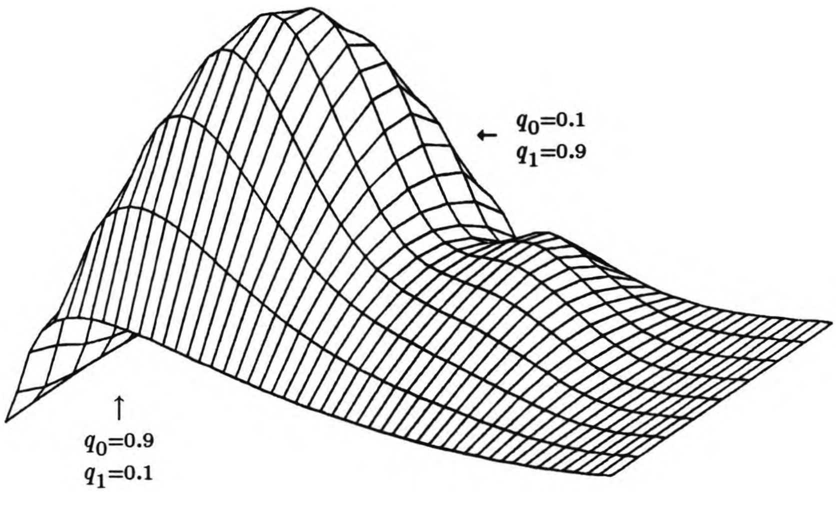


Fig. 4.10b **Parametrical PSD plot of the CMI code**

Obviously a lot more information about the shaping of the spectrum can be obtained from the plots of the CMI spectral density functions. For example, the existence of a discrete component, $y_1^{(d)}$ at the input symbol rate $F_d = 0.5\ F_x$ and the shift of the signal energy (plots y_1 and y_3) with the probability of the input symbols, can be very important if the efficiency of transmission is to be determined. The validity of the arguments for the unified analysis approach becomes even more apparent through the next example

A conventional 3B4B code is taken to represent the binary alphabetic structures with a constraint on the digital sum. It is given as a typical case of the extension of the two-state group of codes, where $M = \alpha^N$ input words are mapped into a subset of $K = \beta^L$ output words[13]. This code has been chosen to demonstrate the possibility for direct comparative assessment provided by the spectral analysis method.

There is a big variety of possible ways to assign a subset of all 4-digit binary words to the set of all binary words of length 3. The problem is to find out when such a mapping results in a practical code with a suitably shaped spectrum. A good example of a 3B4B code is specified in Tables 4.10(a,b). The set of output words consists of all binary blocks of four symbols except 0000, 0011, 1100 and 1111. These combinations have been left out for the obvious reason to avoid long sequences of identical digits. A careful examination of Table 4.10b shows that the code words are arranged to achieve well balanced output sequence. This results in suppression of the low frequency components. The PSD of the 3B4B code is shown in Fig. 4.11a.

[13] α and β are the numbers of the input and the output symbols. N and L are the lengths of the input and the output words, respectively, so that $\alpha^N < \beta^L$.

3B4B - Symbols

INPUT		OUTPUT	
$d=[0,1]$ $q=[0.5,0.5]$	Q_m	$x=[0,1]$	
$D_1 = 0\ 0\ 0$	.125	$X_1 = 0001$	$X_9 = 1010$
$D_2 = 0\ 0\ 1$	.125	$X_2 = 0010$	$X_{10} = 1011$
$D_3 = 0\ 1\ 0$	.125	$X_3 = 0100$	$X_{11} = 1101$
$D_4 = 0\ 1\ 1$	.125	$X_4 = 0101$	$X_{12} = 1110$
$D_5 = 1\ 0\ 0$	.125	$X_5 = 0110$	$X_{13} = 0000$(not used)
$D_6 = 1\ 0\ 1$	.125	$X_6 = 0111$	$X_{14} = 0011$(not used)
$D_7 = 1\ 1\ 0$	.125	$X_7 = 1000$	$X_{15} = 1100$(not used)
$D_8 = 1\ 1\ 1$	.125	$X_8 = 1001$	$X_{16} = 1111$(not used)
$N=3,\ M=8$		$L=4,\ J=12,\ K=16$	

Table 4.10a

3B4B - Code

D_m	s_1	s_2	S_m	Z_m
0 0 0	0 1 0 1 s_1	0 1 0 1 s_2	SM = $\begin{matrix}1&0\\0&1\end{matrix}$	ZM = $\begin{matrix}0&1&0&1\\0&1&0&1\end{matrix}$
0 0 1	1 0 0 1 s_1	1 0 0 1 s_2	$\begin{matrix}1&0\\0&1\end{matrix}$	$\begin{matrix}1&0&0&1\\1&0&0&1\end{matrix}$
0 1 0	1 1 1 0 s_2	0 1 0 0 s_1	$\begin{matrix}0&1\\1&0\end{matrix}$	$\begin{matrix}1&1&1&0\\0&1&0&0\end{matrix}$
0 1 1	1 1 0 1 s_2	1 0 0 0 s_1	$\begin{matrix}0&1\\1&0\end{matrix}$	$\begin{matrix}1&1&0&1\\1&0&0&0\end{matrix}$
1 0 0	0 1 1 1 s_2	0 0 1 0 s_1	$\begin{matrix}0&1\\1&0\end{matrix}$	$\begin{matrix}0&1&1&1\\0&0&1&0\end{matrix}$
1 0 1	1 0 1 1 s_2	0 0 0 1 s_1	$\begin{matrix}0&1\\1&0\end{matrix}$	$\begin{matrix}1&0&1&1\\0&0&0&1\end{matrix}$
1 1 0	0 1 1 0 s_1	0 1 1 0 s_2	$\begin{matrix}1&0\\0&1\end{matrix}$	$\begin{matrix}0&1&1&0\\0&1&1&0\end{matrix}$
1 1 1	1 0 1 0 s_1	1 0 1 0 s_2	$\begin{matrix}1&0\\0&1\end{matrix}$	$\begin{matrix}1&0&1&0\\1&0&1&0\end{matrix}$

Table 4.10b

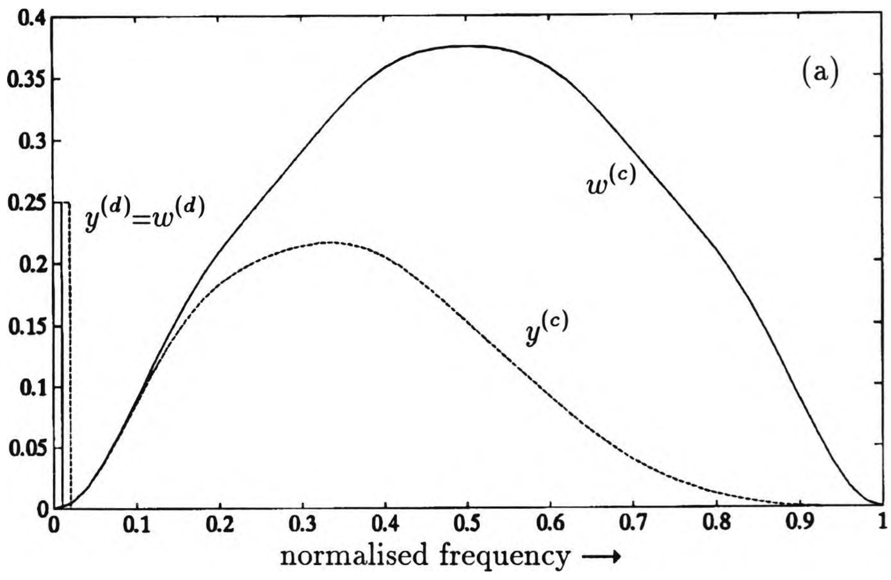


Fig. 4.11 The spectrum of the 3B4B code for equiprobable input symbols

Before continuing with the examples from the next subsection it is worth pointing out a general similarity in describing the coder categories. The basic coding schemes in the two groups outlined above have been specified as having one and two states respectively and a minimal number of input/output blocks of symbols (the NRZ and the Differential codes). The next level of complexity within each of the groups, from this and the previous subsections, has been suggested indirectly, but its discussion is postponed until Chapter 5. The codes from this level are defined as having equal number of possible input and output words, $M = J$. In the one-state category these types of codes offer very little in terms of spectrum shaping. For two-state coders with $M = J$ the possibility to modify the signal spectrum is still very restricted, except under certain conditions, e.g. unequal input symbol probabilities.

The next complexity level, for a given number of states, has been assumed to be codes with a number of output words greater than the number of input words, $M < J$. In this case the freedom of designing a coding scheme, closely matching particular requirements, is substantial, which can be seen easily from the examples of 3B4B and 5B6B codes, given later in this chapter. In general the binary block techniques of the type nB$(n+1)$B are very popular for long distance, high capacity communication lines. Many of them have received considerable attention and their spectral characteristics have been studied thoroughly [7,21]. Amongst the fixed length block codes with $n > 5$ the 7B8B scheme has become widely used, because of its good efficiency and the possibilities for modifications providing for error monitoring and ancillary channels [Brooks, 1981].

Finally it should be noted that when the number of states, I and the sizes of the input/output word sets (M and J) are fixed, it is possible to devise a variety of codes corresponding to different subsets of J valid output blocks of symbols. This is illustrated by the examples of the RZ and the Manchester techniques. The set of output words, X for these codes is smaller than the set of all $K = \beta^L$ possible output words, where β is the number of output symbols and L is the length of the output words. Obviously the larger the numbers J and K the greater the variety of possible coding structures. In the case of two-state coders this also means a relative increase in complexity of the code rules.

A different level of complexity is reached through codes with more than two states. Very interesting effects in terms of spectrum shaping can be achieved with such techniques, even if the size of the output dictionary is relatively small, say $J \leq 4$. Some existing coding schemes from this category are presented in the next subsection.

4.1.2.3 Coding Structures With More Than Two States

The use of more than one coder state has been associated with memory, i.e. the coded output depends on previous events. This has been discussed in Chapter 3 and formally indicated in Fig. 3.2 as the delay component. Although a general solution to the problem of how to determine the set of states is described later in this chapter, it has been indicated on several occasions in previous sections that in most cases it is possible to identify the sign and/or the value of preceding output symbols as the coder states. Obviously the larger the number of time intervals on which the present output depends, the greater the number of combinations of past events which may lead to different states.

The descriptive definitions of the codes from Chapter 2 show that in most cases the conditions for the output to take some value are specified as combinations of previous output or input values. The dependence of the output word on one preceding input symbol for the Duobinary format and on two preceding symbols for the Modified Duobinary (see Section 2.3.2) is very important for the process of 'translating' the conditions from the description of the coder rules into the appropriate set of states. Consequently the Modified Duobinary scheme is modelled as a four-state code, unlike the Duobinary which requires two states as shown in the previous subsection. A new practical procedure for determining the states of a coder in general is suggested in a forthcoming section.

The unified specification and the spectral analysis results shown below are based on the assumption that coder states represent sequences of signal values as defined in the respective code descriptions (Section 2.3.2). Thus the set of states for the Modified Duobinary technique can be determined as the following combinations of the two preceding input symbols: $s_1 = 00$; $s_2 = 01$; $s_3 = 10$; $s_4 = 11$. This allows the complete specification of the code to be given in the table form shown below.

Modified Duobinary - Code

D_m	s_1	s_2	s_3	s_4	S_m	Z_m
0	$0 \quad s_1$	$0 \quad s_3$	$-1 \quad s_1$	$-1 \quad s_3$	$\begin{bmatrix} 1&0&0&0 \\ 0&0&1&0 \\ 1&0&0&0 \\ 0&0&1&0 \end{bmatrix}$	$\begin{bmatrix} 0 \\ 0 \\ -1 \\ -1 \end{bmatrix}$
1	$+1 \quad s_2$	$+1 \quad s_4$	$0 \quad s_2$	$0 \quad s_4$	$\begin{bmatrix} 0&1&0&0 \\ 0&0&0&1 \\ 0&1&0&0 \\ 0&0&0&1 \end{bmatrix}$	$\begin{bmatrix} +1 \\ +1 \\ 0 \\ 0 \end{bmatrix}$

Table 4.11

The specification of the symbol sets is identical to that of the Duobinary given in Table 4.8a. The Modified Duobinary code has been developed to preserve the bandwidth reduction achieved by the Duobinary format as well as to achieve the advantage of a spectrum with suppressed low frequency components similar to that of the AMI code, for example. The resulting spectral density distribution is shown in Fig. 4.12.

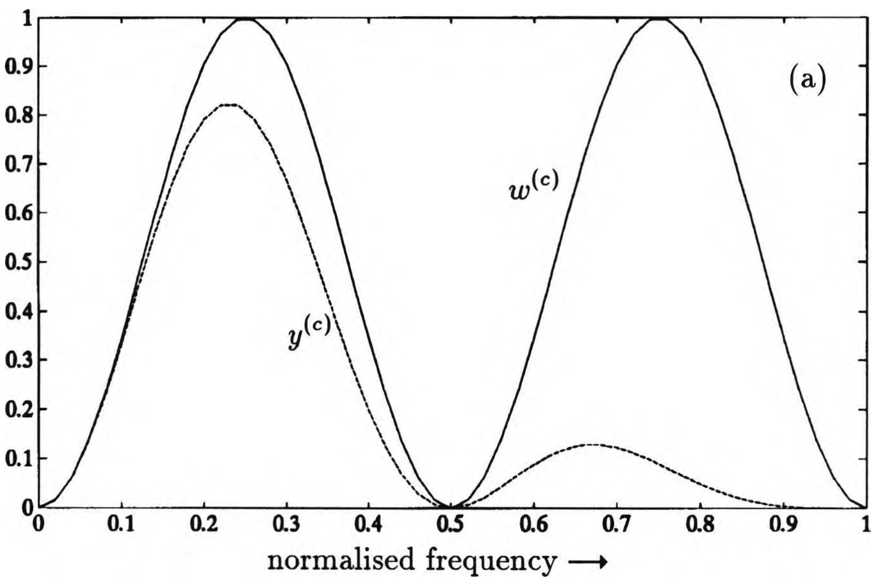


Fig. 4.12 The PSD of the Modified Duobinary code for equiprobable input symbols

The comments made about the PSD of the Duobinary code, shown in Fig. 4.9, can be extended naturally to its modified version. Apart from the obvious increase in complexity by having four states instead of two, it is also possible to identify clearly, from Table 4.11, the specific design considerations.

The general structure of a random input binary sequence can be informally divided in two probabilistic types: one with frequent occurrence of both input symbols and another with long strings of identical digits. For an input sequence of the first type the coder will 'switch' frequently between the two rows of the 'Modified Duobinary-Code' table, i.e. it will be attaining either 0 or -1 and $+1$ with equal probability[14]. For the second type of input sequence structure, long strings of either symbol will always make the output sequence converge to 0 (states s_1 and s_4). In both cases the average output is zero which leads to the elimination of the d.c. component. If the input sequence structure is described in a greater detail, the spectral features of the output sequence become more obvious by applying the above reasoning.

[14] The 2D plot in Fig. 4.12 is computed for equiprobable input symbols.

Assessment of a coder by 'operating' it through its state-transition table is obviously equivalent to the conventional descriptive definition of codes. However, the significant advantage in using the table presentation is its uniformity and its universally applicable structure. This allows most techniques to be compared directly, as well as important features of different codes to be easily identified. In addition, it is possible to derive the analytical form of the coder definition (the state-transition and the output matrices) directly from 'Code' tables. The most significant advantage provided by the unified code definition is the direct implementation of the general spectral analysis model through the computational procedure, which requires the sets of input and output words, the state-transition matrix SM and the output matrix ZM as the only initial numeric data to perform precise evaluation of the spectral density function.

Before presenting the analysis results of other codes with more than two states a possible approach in deducing essential features of codes from their unified specification is summarised in brief. The comparative overview involves the NRZ, Manchester, Differential, AMI, Duobinary and Modified Duobinary schemes. By following this order in the presentation of the essential characteristics of these codes, the process can be viewed as increasing the complexity of the particular techniques in order to accomplish specific features of the spectral density function.

The 'Manchester-Code' definition, Table 4.12, indicates that to reduce the amount of low frequency components with a one state coder, in a simple way, information redundancy is introduced.

Manchester - Code

D_m	s_1		S_m	Z_m
0	0 1	s_1	1	0 1
1	1 0	s_1	1	1 0

Table 4.12

This allows transitions to be provided for every input symbol, therefore long sequences of uniform digits are eliminated. The goal is achieved (see Fig. 4.5a) at the expense of low efficiency (0.5, as only two out of four possible output words are used) and doubling the transmission rate. Further increase in the first level of complexity, the ratio $\frac{L}{N}$, is not expected to bring any significant improvements in shaping the spectrum. At the same time the efficiency becomes very low (for the purposes of line coding) when $\frac{L}{N} > 2$. It is , however, possible to accomplish some interesting results with a one state coder by keeping $\frac{L}{N}$ relatively small and using more than two input words, i.e. $N > 1$ in the binary case. Such codes have not been used in this assessment as they are not widely implemented in practice. They are considered in greater detail in Chapter 5.

The next complexity level, as defined earlier is the introduction of two states. The simplest representative of this class is the Differential code, which provides for no redundancy and no increase in the transmission rate. It has been shown in Chapter 2 that by suitable redefinition of the output symbols it is possible to represent the code as a one-state technique. The result of applying Differential coding can be summarised as follows: with no change in the transmission rate, code efficiency of 1 and use of a second state, the effect in modifying the spectrum is only 'partial', i.e. the spectral density for equiprobable input symbols is identical to that of the NRZ format and changes in the frequency distribution occur only for unequal input probabilities (Fig. 4.7a). Although it is difficult to generalise at this stage, an assumption can be made that the increase in the level of complexity has been insufficient to provide for flexibility in modification of the spectrum.

The comparative analysis proceeds on the basis of two-state coding. To achieve the effect of reducing the low frequency components (as with the Manchester format) the AMI code can be used, Fig. 4.8a. Examination of the 'AMI-Code', Table 4.7b, shows that the efficiency (0.63) is less than 100%, although it is still higher than that of Manchester. There is no increase in the transmission rate and a special feature, the unique decodability[15] has been preserved. In a few words, it has taken an increase of the number of states and an introduction of redundancy to accomplish suppression of the low frequency components of the coded signal without changing the transmission rate.

Another significant achievement with respect to the modification of spectra is the possibility to reduce the transmission bandwidth. This effect is produced by the Duobinary code and illustrated with the PSD plot in Fig. 4.9. The direct comparison between the 'AMI-Code' and the 'Duobinary-Code', Table 4.7b and Table 4.8b, shows a single difference: the loss of unique decodability with the latter scheme. The comments regarding the transition between the AMI and the Duobinary, which have been made earlier, are summarised in Table 4.13.

By the introduction of state-dependent decoding, while preserving the other structural features and the code efficiency unchanged (compared to AMI), the Duobinary technique provides for a twofold reduction of the transmission bandwidth, but 'fails to keep' the low frequency components suppressed. Additional increase in complexity is required in order to produce a coded sequence which features both: suppressed low frequencies and reduced transmission bandwidth. This can be done through further structural changes. The Modified Duobinary code achieves it by the use of four states instead of two, while the structural feature *a*) (Table 4.13), therefore the efficiency, remains the same.

[15] No output word corresponds to more than one input word.

CODE	FEATURES	
	structural	*spectral*
A M I	a) $N = 1, M = 2$ $L = 1, J = 3$ b) two states	suppressed low-frequency components
Duobinary	a) $N = 1, M = 2$ $L = 1, J = 3$ b) two states c) *state-dependent decoding*	reduced transmission bandwidth
Modified Duobinary	a) $N = 1,\ M = 2$ $L = 1,\ J = 3$ b) *four states* c) state-dependent decoding	a) suppressed low-frequency components b) reduced transmission bandwidth

Table 4.13 Relations between structural and spectral features

The next example of a four-state code, described in Chapter 2 as 'Miller', illustrates another interesting approach in the development of a technique, which provides for the following frequency characteristics

a) small amount of low frequency components;

b) narrow transmission bandwidth.

Characteristic a) is achieved through the use of two output symbols for every input symbol, which leads to increase of the number of transitions in the coded sequence. The specification of appropriate symbol sets is given in Table 4.14a.

Miller - Symbols

input	output
$d = [0, 1]$	$x = [0, 1]$
$D_1 = 0$ $D_2 = 1$	$X_1 = 0\ 0$ $X_2 = 0\ 1$ $X_3 = 1\ 0$ $X_4 = 1\ 1$
$N = 1, M = 2$	$L = 2, J = K = 4$

Table 4.14a

The definition of the symbol sets suggests two important features of the code. The output sequence is binary and the output dictionary consists of all possible words. By its specification the Miller code is closest to the group including the RZ, CMI and Manchester formats, all of which require twofold increase of the transmission bandwidth.

In this case, however, the characteristic b) is achieved through assigning the output words to the sequence of input symbols in a way which ensures that a transition in the output sequence is always followed by at least two identical symbols. This condition provides that no more than one transition in the coded signal appears within an interval of duration T_d (the period of the input symbols). The table presentation of the Miller code is based on four states which can be identified as all possible combinations of the last two consecutive output symbols corresponding to the descriptive definition of this technique given in Chapter 2, i.e.: $s_1 = 10$; $s_2 = 00$; $s_3 = 01$; $s_4 = 11$. The complete specification of the code is then given in Table 4.14b.

Miller - Code

D_m	s_1	s_2	s_3	s_4	S_m	Z_m
0	0 0 s_2	1 1 s_4	1 1 s_4	0 0 s_2	$\begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \end{bmatrix}$	$\begin{bmatrix} 00 \\ 11 \\ 11 \\ 00 \end{bmatrix}$
1	0 1 s_3	0 1 s_3	1 0 s_1	1 0 s_1	$\begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix}$	$\begin{bmatrix} 01 \\ 01 \\ 10 \\ 10 \end{bmatrix}$

Table 4.14b

It is not difficult to see directly from the 'Miller-Code' table that every output symbol is adjacent to at least one identical digit. Indeed, an input 0 is always coded as 00 or 11, while for an input 1 the second symbol of the respective output pair is always the same as the first symbol of the output pair following next. This ensures the minimum interval between two transitions in the coded signal to be equal to the input symbol period, T_d. The conventional way to demonstrate this feature of the Miller code is by showing pulse waveforms of sample sequences. The resulting spectral density distribution is shown in Fig. 4.13.

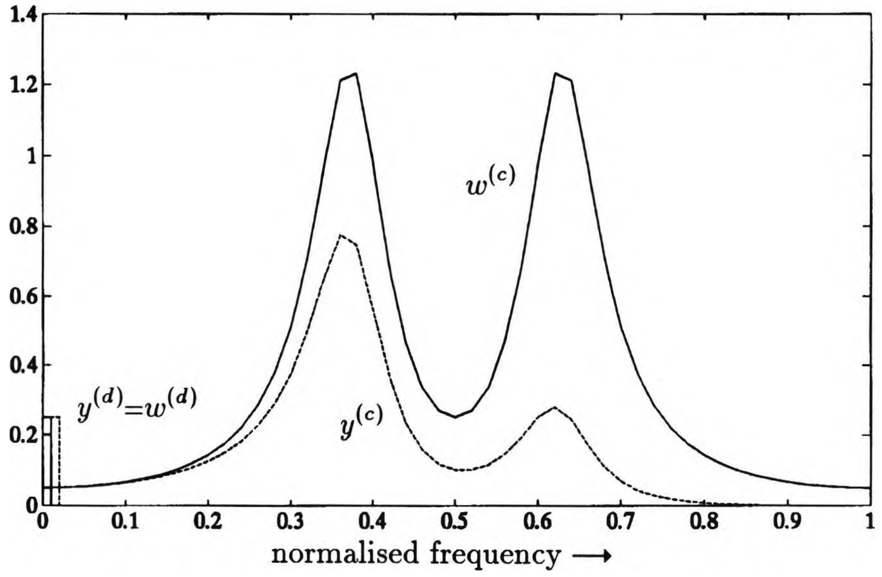


Fig. 4.13a Conventional PSD plot of the Miller code

Clearly the PSD of the Miller-coded signal possesses features a) and b) specified above. The reduction of the low frequencies in the output sequence, typical of a 1B2B structure, is well illustrated by the plot in Fig. 4.13a. The situation is similar for the higher frequency range. Another important result is that most of the signal power is concentrated below $0.5F_x$, i.e. within the bandwidth of the input sequence.

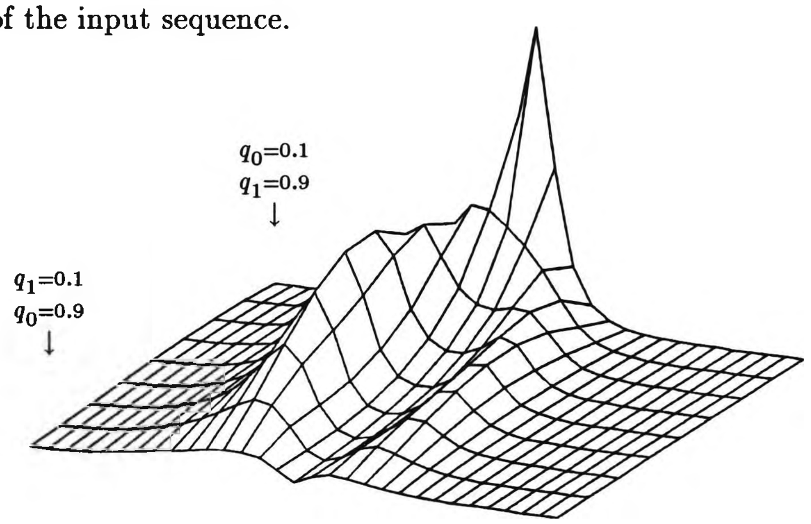


Fig. 4.13b Parametrical PSD plot of the Miller code

The code analysed next is a four-state scheme which shows both the spectrum shaping potential of codes from this complexity level as well as possible confusions which arise from applying approximate and incomplete methods of analysis. The 3B2T-RBS coding technique, used as an introductory example in Section 4.1.1, offers a good illustration of the drawbacks of descriptive PSD evaluation. The assessment of the frequency characteristics in reference [26] is performed by calculating the probabilities of certain symbol combinations. The balance features of the code are determined through a similar approximation. Based on deduced results, the paper suggests that the 3B2T-RBS features superior balance, transmission bandwidth and overall performance to similar codes and the 5B6B in particular.

The validity of the above arguments can be assessed by considering the basic relation between the code-symbol transmission rates of 5B6B and 3B2T-RBS (the latter being a version of the 3B4B block codes). The expression relating the two transmission rates, $(F_x)_{\text{5B6B}}$ and $(F_x)_{\text{3B2T-RBS}}$, is derived by assuming that both codes are applied to the same input sequence with a symbol period of $T_d = \frac{1}{F_d}$, where F_d is the input symbol repetition rate:

$$(F_x)_{\text{5B6B}} = 1.2F_d = 0.9(F_x)_{\text{3B2T-RBS}}$$

This result corresponds to the potentially higher coding efficiency of nBlB block codes with higher values of $\frac{n}{l}$. It should be noted, however, that the last statement applies for equivalent initial conditions. The first condition has been specified as the use of the same input symbol rate with all codes to be compared. Another important requirement is to match the complexity level of the respective coding schemes. (Complexity is discussed in grater detail in Chapter 5.) For the example involving the 5B6B and 3B2T-RBS schemes the second requirement can be interpreted as follows. Although it might be possible to find a four-state block code, n_1Bl_1B, which performs better than a two-state block code, n_2Bl_2B, where $\frac{n_1}{l_1} < \frac{n_2}{l_2}$, the larger number of coder states itself does not guarantee higher efficiency. At the same time, for an equal number of coder states, an n_iBl_iB can always be designed to be more bandwidth-efficient than an n_jBl_jB code, while $\frac{n_i}{l_i} > \frac{n_j}{l_j}$.

The PSD computational procedure allows to achieve more accurate analysis and comparison of the two schemes discussed above. The spectra of the 3B2T-RBS and 5B6B[16] are shown in Fig. 4.14a, together with the conventional 3B4B code[17].

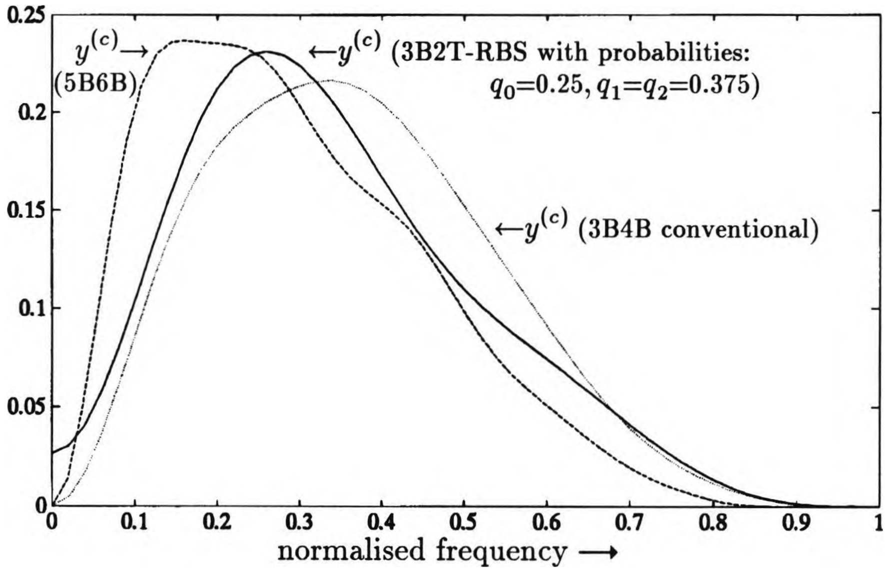


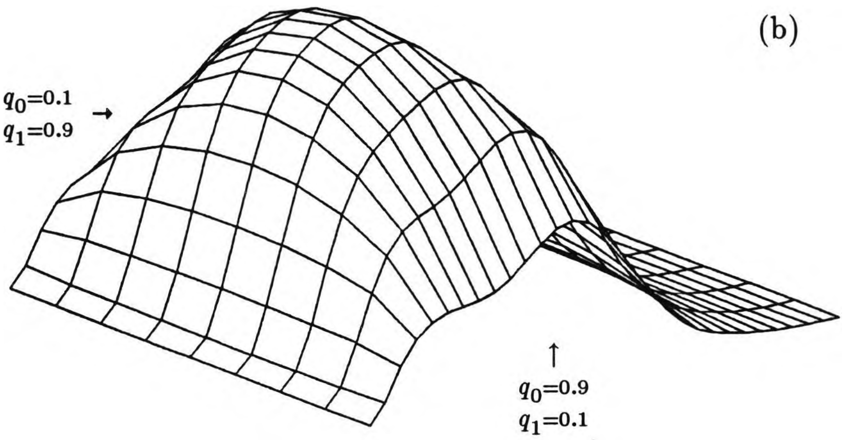
Fig. 4.14a Spectra of the 5B6B, 3B4B and the 3B2T-RBS codes

Although the actual signal bandwidth depends on the practical implementation, it can be seen that, for the same input sequence, 3B2T-RBS does not show any bandwidth superiority over 5B6B. No particular improvement is evident even in comparison with the conventional 3B4B code. At the same time an obvious

[16] The spectrum of the 5B6B code sequence has been evaluated on the basis of a specification given in [26].

[17] Referred to as "fibre optics" code in [18].

disadvantage of the 3B2T-RBS spectrum is the poorer suppression of the low-frequency components and its nonzero value for $f = 0$. In fact it is possible to show that the 3B4B code exhibits better balanced features by applying the parametrical spectral analysis. The results are given in Fig. 4.13(b,c) and reveal the higher stability of the 3B4B spectrum, (b) compared to that of 3B2T-RBS, (c), which shows the familiar shift towards low or higher frequencies for different input-symbol probabilities.



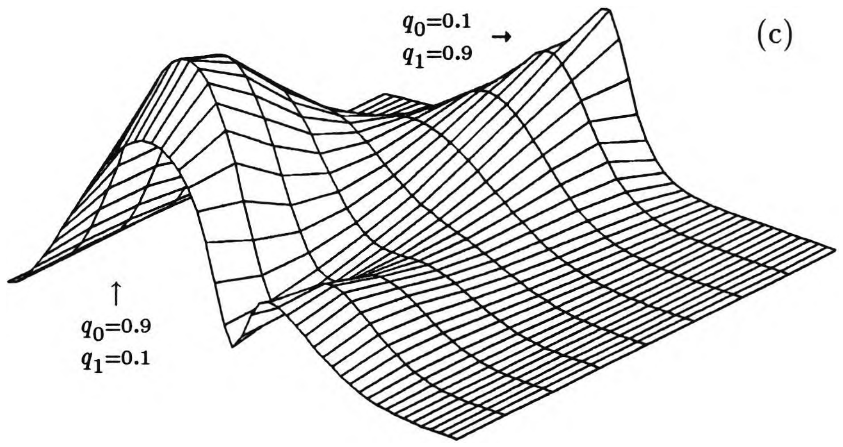


Fig. 4.14 PSD-s of the 3B4B and the 3B2T-RBS codes for variable input probabilities

An additional complication in producing a conclusive spectral analysis evaluation of the 3B2T-RBS technique arises from the lack of well defined initial conditions. According to its specification, [26] the code is a "3B-2T-4B" transformation, where the input statistics should be determined by the probabilities of the binary symbols comprising the input words. In this case the intermediate transitions, 3B→2T and 2T→4B, are only used for convenience in deriving the coding rules and the result is a version of the 3B4B block codes, which has been given as an introductory example (Tables 4.3 and Fig. 4.3).

At the same time the frequency analysis considerations, given in the publication quoted above, are based on the probabilities associated with the ternary symbols 0, 1 and 2. It has been shown in Chapter 3 that the evaluation of the statistics of the coded sequence depends on the degree of stationarity and ergodicity of the input. If the ternary sequence in the 2T4B transformation derives from a 3B2T conversion, it is not only the symbol probabilities which change, but its structure and overall statistics are altered, too.

The difference between the spectral features of the 3B-2T-4B scheme, specified in [26] as the 3B2T-RBS code and the corresponding ternary-to-binary conversion schemes, where only the variation of the input probabilities is taken into account, can be illustrated by the following example. A 1T2B code is specified according to the 3B2T-RBS rules in Tables 4.15(a,b).

1T2B-Symbols

input			output	
$d = [0,1,2]$	Q_m (i)	Q_m (ii)	$x = [0,1]$	
$D_1 = 0$	0.333	0.250	$X_1 = 0\ 0$	$X_3 = 1\ 0$
$D_2 = 1$	0.333	0.375		
$D_3 = 2$	0.333	0.375	$X_2 = 0\ 1$	$X_4 = 1\ 1$
$N = 1, \quad M = 3$			$L = 2, \quad J = K = 4$	

Table 4.15a

1T2B-Code

D_m	s_1	s_2	s_3	s_4	S_m	Z_m
0	10 s_1	10 s_2	01 s_3	01 s_4		
1	11 s_4	00 s_1	11 s_4	00 s_1	$\mathrm{SM} = \begin{bmatrix} 1&0&0&0\\0&1&0&0\\0&0&1&0\\0&0&0&1\\ \\0&0&0&1\\1&0&0&0\\0&0&0&1\\1&0&0&0\\ \\0&0&1&0\\0&0&0&1\\1&0&0&0\\0&1&0&0 \end{bmatrix}$	$\mathrm{ZM} = \begin{bmatrix} 1&0\\1&0\\0&1\\0&1\\ \\1&1\\0&0\\1&1\\0&0\\ \\0&1\\0&1\\1&0\\1&0 \end{bmatrix}$
2	01 s_3	01 s_4	10 s_1	10 s_2		

Table 4.15b

The results from the spectral analysis of the 1T2B coded sequence are shown in Fig. 4.15 for two cases:

i) equiprobable input symbols, $p(0) = p(1) = p(2) = \frac{1}{3}$;

ii) probabilities corresponding to to the 3B2T transformation rules, where the ternary block 00 is not used, i.e. $p(0) = 0.25$, $p(1) = p(2) = 0.375$.

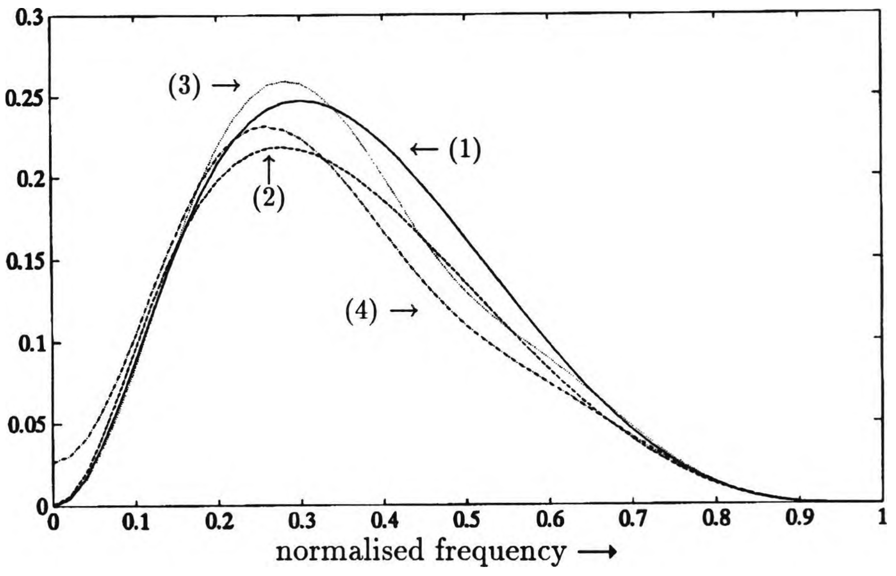


Fig. 4.15 Spectra of the 3B2T-RBS code versions

The four curves in Fig. 4.15 represent the spectral densities of binary sequences resulting from the same coding rules applied to differently specified input sequences. Curves (1) and (2) correspond to cases i) and ii) above. These two spectra can only be achieved if the ternary input sequence is independent and random. The other two curves, correspond to the 3B-2T-4B transformation. PSD (3) is produced by a 3B2T-RBS equivalent code, assuming equiprobable input symbols. In this case the intermediate ternary transformation is disregarded or (which is effectively the same thing) the probabilities of the input blocks of two ternary symbols are assumed equal. Such a definition utilises the balanced features of the code rules, which shows in the zero value of the PSD for $f = 0$. The original 3B2T-RBS, however, assumes unequal probabilities of the blocks of ternary symbols, due to not using the 00 combination. This leads to the specification of a 3B4B structure, equivalent to the 3B2T-RBS with unequal probabilities of the input blocks of three binary symbols. The resulting spectral density is shown as plot (4) in Fig. 4.15. This graph exhibits the least favourable spectral density and is the only one which does not decrease to zero for $f \rightarrow 0$. It can be seen that the first and the second pair of curves represent different codes, although they derive from the same coding rules, given in descriptive form in [26]. This difference is adequately reflected in their definitions (Tables 4.3 and 4.15), specified through the uniform code assessment method.

The unified analysis technique can be applied to a much larger number of existing codes than it is practical to include in this presentation. More coding techniques from various publications[18] have been assessed only to confirm the regularity and the validity of conclusions which have been reached empirically. A major consequence from the suggested scheme for general assessment of codes is the possibility to deduce common structural and spectral features, allowing particular spectrum shaping features to be produced. The enhancement of the spectral analysis algorithm into a block-code generating procedure[19] allows a large number of new schemes, with virtually any size of the symbol and coder states sets, to be designed. The presentation and the discussion of new structures, proposed in this section, has been limited to a few examples only. These are considered sufficient to demonstrate the practical implementation of the relations between the complexity of codes and the ability to modify the PSD. The examples do not aim to suggest perfect solutions to line coding problems. Most of the spectral density functions are different from those of conventional codes and may be suitable for particular applications, but their main purpose is to illustrate the use of some important design considerations, deriving from the application of the uniform analysis approach.

4.1.3 New Codes Designed Through the Unified Spectral Analysis Approach

The structures discussed next are based on the four state scheme of the Miller code and use the 1B2B symbol transformation. The choice follows the summary of the spectral and structural relations, given in Table 4.13, which indicate the flexibility of the four-state schemes, like Modified Duobinary and Miller, added to the 50% redundancy provided by the 1B2B structures, like Manchester and CMI. The higher complexity levels of the codes, selected to illustrate the construction of new schemes, allows the latter to be presented not only as an interesting achievement, but also to show the most essential considerations in the design process, deriving from the systematic assessment and analysis routine.

The codes specified in Table 4.16 are 1B2B structures, which have been upgraded to the four-state complexity level.

[18] The following is a small sample list of the references, describing coding schemes which have been analysed, [31,32,41,42,43].

[19] Discussed separately in Chapter 5.

ARZ(a) and MRZ(b) - Code

D_m	s_1	s_2	s_3	s_4	S_m	Z_m(a) Z_m(b)
0	0 0 s_2	0 0 s_4	0 0 s_4	a) 00 s_2 b) 01	$\begin{bmatrix} 0&1&0&0 \\ 0&0&0&1 \\ 0&0&0&1 \\ 0&1&0&0 \\ 0&0&1&0 \\ 0&0&1&0 \\ 1&0&0&0 \\ 1&0&0&0 \end{bmatrix}$	$\begin{bmatrix} 00 \\ 00 \\ 00 \\ 00 \\ 10 \\ 11 \\ 10 \\ 11 \end{bmatrix}$ $\begin{bmatrix} 00 \\ 00 \\ 00 \\ 01 \\ 10 \\ 11 \\ 10 \\ 11 \end{bmatrix}$
1	1 0 s_3	1 1 s_3	1 0 s_1	1 1 s_1		

Table 4.16

These schemes are similar to the Miller code but their construction can be discussed from a different point of view. The structure to be analysed first uses option a) from the s_4 column of Table 4.16 for coding input zeros and is called Arbitrary Return-to-Zero (ARZ). If all code blocks for input 1 were either 11 or 10 only, the scheme will produce either NRZ or RZ sequences, respectively. The use of four states allows the output symbols to be coded as NRZ or RZ in an arbitrary succession. It can be seen that, having adopted the general table presentation, the design of the ARZ code has been based entirely on the relations between state transitions and output words. i.e., the two blocks of symbols, 11 and 10 have been chosen to combine the NRZ and the RZ patterns, while the switching between the different output words has been provided through a four-state coder. The state transition pattern has been taken from the Miller code, anticipating the symmetry of its state-transition scheme to preserve the balance of the input symbols. The result from the construction of the ARZ scheme is expected to combine the spectral features of the two conventional techniques, i.e. the bandwidth should be narrower than that of the RZ and the low frequencies should be reduced due to the introduction of transitions in a predetermined pattern. Indeed, the spectral density of the ARZ coded signal, given as plot (1) in Fig. 4.16 shows the effectiveness of the design approach.

The bandwidth is reduced to half that of the code symbol rate, i.e. half the RZ spectrum and the low frequency components have been suppressed almost by half. It should be noticed that, if $y^{(c)}(f) = 0.25$ is taken as the normalised NRZ value for $f = 0$, signal power has been shifted from the low towards the middle frequencies, around $f \approx 0.25F_x = 0.5F_d$ of the reduced bandwidth, which is an additional spectrum modification effect.

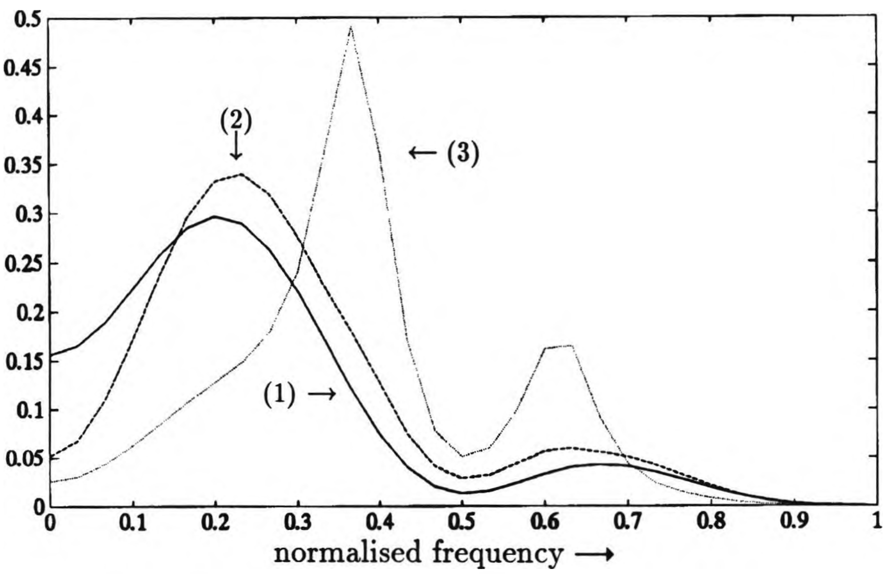


Fig. 4.16 Spectra of the ARZ and the MRZ coding schemes

The ARZ is not a code of an exceptional practical value, but it is a good illustration of the transition from the systematic analysis towards generalised implementation of the essential relations between structural and spectral features of codes, for the purpose of spectrum shaping. It has been demonstrated that the uniform interpretation of the code table definitions and the analysis results has allowed specific code characteristics to be combined in order to produce predictable spectrum modification results. The same simulation mechanism can be used to reiterate the applicability of the design considerations allowing the spectral density function to be modified predictably, through the selection of specific structural features.

By using the same state transition scheme, as with the ARZ code, it is possible to experiment with a 'fine-tuning' alteration, which is expected to produce a distinctive change in the PSD. Another new code can be determined from Table 4.16 by using version b) in column s_4, which replaces the output word $X_{14} = 00$ with 01. The scheme is referred to as Modified Return-to-Zero (MRZ). The larger number of transitions in the output sequence, resulting from the change, leads to an increased level of high frequency components at the expense of reduced low frequency components. The latter is due to the 01 output block, used for input 0 and state s_4, which limits the maximum number of consecutive zeros in the coded sequence. The spectrum modification effect of the MRZ, compared to ARZ, is exactly what has been anticipated as shown by graph (2) in Fig. 4.16.

It is not necessary to go into too much detail, in order to present the results from one further step in the investigation of new coding schemes. Without showing the respective 'Code' table explicitly, it is possible to specify a structure which uses only the 01 and 10 blocks to code input 1, while preserving the transformation of input 0 into 01 at least for one coder state. Applying similar reasoning as for the ARZ and MRZ structures, the prediction for further suppression of the low frequency components and some increase of the high frequencies can be made. This is due mainly to the elimination of the 11 output block. It turns out that the predictions are correct and the result seems to be a very attractive[20] spectrum, shown as plot (3) in Fig. 4.16. Unfortunately, the price paid for such a considerable spectrum shaping effect is the introduction of complicated state-dependent decoding, which significantly reduces the practical value of such a scheme.

The investigation into the possibilities to construct new codes can be extended by considering a different type of alterations in Table 4.16. Instead of increasing the number of transitions, as has been done for the MRZ code, the all-zeros transformation of input 0 provided by the ARZ code is preserved and in addition, one of its output blocks, used to code input 1, is replaced with a 00 output word. The resulting structure is given in Table 4.17.

D_m	s_1	s_2	s_3	s_4	S_m	Z_m(a)	Z_m(b)
0	0 0 s_2	0 0 s_4	0 0 s_4	0 0 s_2	$\begin{bmatrix} 0&1&0&0\\ 0&0&0&1\\ 0&0&0&1\\ 0&1&0&0\\ 0&0&1&0\\ 0&0&1&0\\ 1&0&0&0\\ 1&0&0&0 \end{bmatrix}$	$\begin{bmatrix} 00\\ 00\\ 00\\ 00\\ 00\\ 01\\ 11\\ 11 \end{bmatrix}$	$\begin{bmatrix} 00\\ 00\\ 00\\ 00\\ 00\\ 01\\ 11\\ 10 \end{bmatrix}$
a)		0 1 s_3	1 1 s_1	1 1 s_1			
1	0 0 s_3						
b)		0 1 s_3	1 1 s_1	1 0 s_1			

Table 4.17 Combined specification of two modified RZ codes

An immediate effect of this change is expected to be the reduction in the high-frequency contents of the spectrum due to the fewer transitions introduced by the scheme. The spectral density function, corresponding to the code, which uses the output words of version a) from Table 4.17, is given as plot (1) in Fig. 4.17. It exhibits the predicted decrease of signal power in the high frequency range, compared to the PSD-s of ARZ and MRZ, shown in Fig. 4.16. A similar change is achieved with a coding structure which differs from version (a) of Table 4.17 only in the value of the output word $X_{24} = 11$, which is replaced with 10. This change is given as version (b) and its effect can be explained by considering the timing relations between transitions in the coded signal. The transitions,

[20] Narrow bandwidth and small high frequency components.

which are possible to appear when block 11 is used, are at least one input-symbol interval, T_d apart. The code blocks 01 and 10 introduce transitions which can be as close as one output-symbol interval, $T_x = 0.5T_d$ to each other. The most likely result is having larger high frequency components and it shows up very distinctively in the PSD, which is given as plot (2) in Fig. 4.17.

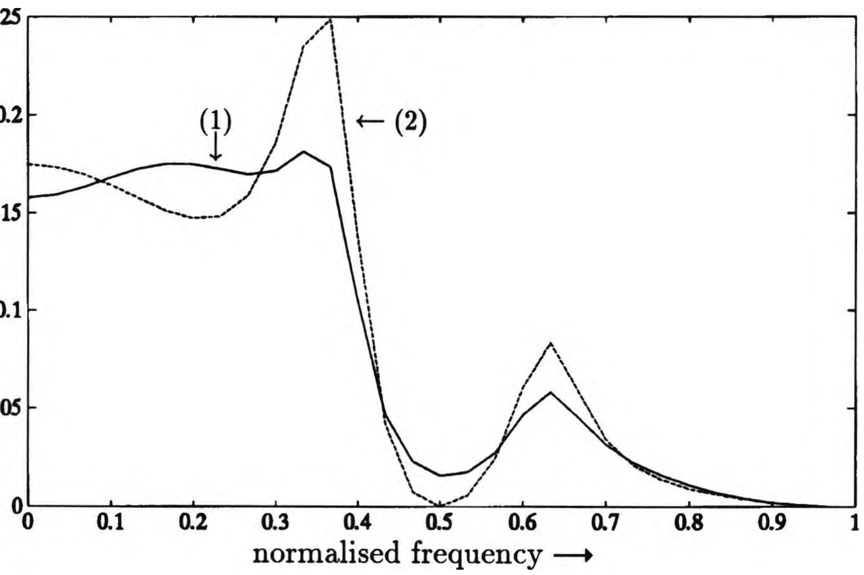


Fig. 4.17 PSD functions with preselected features

Although the proposed new codes do not represent optimal combinations of four states and the four output words 00, 01, 10 and 11, the resulting spectral densities exhibit certain features of practical interest. The bandwidth of the coded signals, produced by all four schemes, is within the range 0-F_d, where F_d is the input-symbol rate, although the code-symbol rate is $F_x = 2F_d$. The coding techniques show the direct applicability of the relations between structural and spectral features, which have been revealed through applying the proposed systematic analysis approach to existing code structures. The uniformity of the code definitions, as well as the analysis results, allow useful features of popular schemes to be combined, achieving predictable spectrum shaping effects.

The ARZ, MRZ and the two codes from Table 4.17 have been used mainly to illustrate the potential for creating new structures by selecting an appropriate state-transition scheme and experimenting with various subsets of output-symbol blocks. The initial objectives of the research work presented in the thesis have been to use the theoretical coder model from Chapter 3 in the development of a general analysis procedure, which would allow uniform and accurate assessment of virtually any block-coding scheme to be performed. This has been expected to facilitate the comparison of different coding techniques and the ability to determine the most appropriate choice of a code for a particular application. Additionally the suggested uniform analysis approach has been intended to provide a suitable basis for the design of new structures.

The achievement of these objectives has been demonstrated through the results from the systematic analysis applied to most of the popular codes, used for spectrum shaping in digital signal transmission, presented in this chapter. It has also been shown how coding schemes can be created by synthesising the required characteristics through combinations of essential structural and spectral features, identified through the uniform analysis of existing codes. A large variety of new structures can be derived similarly to the four codes proposed above. The simplicity and the flexibility of the software routine, created to implement the coder model, allow the analysis of various combinations of coder states and symbol sets to be performed almost instantaneously.

In the process of research the computational procedure has been developed into a simulation algorithm, which is capable of generating an unlimited number of codes. Consequently, the random search for interesting code structures has been transformed into a systematic and exhaustive investigation of block-coding schemes, produced and arranged in classes and categories according to predefined levels of complexity. This is why the discussion of new codes is further presented in Chapter 5, where the advantages and the viability of the proposed method for general classification of fixed-length block codes are fully revealed.

The comparative assessment of coding techniques, through the suggested analysis algorithm, requires transformation of the descriptive definitions of coder rules into 'Code' specification tables. A new method, which has been created to achieve this transformation is presented in the next section.

4.2 A General Method for Constructing the 'Code' Specification Table

In essence this method is a further generalisation of the ideas, summarised at the beginning of Section 4.1.2.3 above, about the definition of states for the Modified Duobinary code. To simplify its introduction it is best to start with an example which shows the link between the values assigned to the states of that code and its specification in Table 4.11. These values have been assumed to represent the last two symbols, $D((k-2)T)$ and $D((k-1)T)$ from the input sequence, $(k = 0, \pm 1, \pm 2, \ldots)$. As the Modified Duobinary is a symbol-by-symbol coding scheme, the problem to be solved is to determine the state, $s((k+1)T)$ into which the coder goes after being in state $s(kT)$, for every possible value of the latter. Knowing that these values are two-digit binary numbers (the input dictionary is $D = [0,1]$), four states can be identified in this particular case by applying the following relation:

$$s(kT) \equiv \{D((k-2)T), D((k-1)T)\} \quad \Rightarrow \quad s((k+1)T) \equiv \{D((k-1)T), D(kT)\} \qquad (4.1)$$

where the 'imply' sign, $\Rightarrow$ represents transition.

Having assumed a random binary sequence as an input, all possible states involved in transitions (4.1) can be represented by the combinations of three consecutive input symbols, $\{D((k-2)T), D((k-1)T), D(kT)\}$. For the Modified Duobinary this representation is specified in Table 4.18 below:

s_i	$s(kT)$		$D(kT)$	$X(kT)$	s_i
s_1	0	0	0	0	s_1
s_1	0	0	1	+1	s_2
s_2	0	1	0	0	s_3
s_2	0	1	1	+1	s_4
s_3	1	0	0	–1	s_1
s_3	1	0	1	0	s_2
s_4	1	1	0	–1	s_3
s_4	1	1	1	0	s_4
present	$D((k-2)T)$	$s((k+1)T)$		output	next

Table 4.18 State definition scheme for the Modified Duobinary code

The fifth column in Table 4.18 shows the output symbols produced by a Modified Duobinary coder. It derives directly from the alternative description given in Section 2.3.2. The values of the output sequence can also be viewed as calculated from the algebraic relation between the input and the output symbols given by (2.4b) applied to the rows from the second, third and fourth columns of Table 4.18. The main advantage in constructing this table, however, is the straightforward way of deriving the transitions between the states. As each coder output corresponds to three consecutive input symbols, all possible combinations of three binary digits are formed. The next step is to assign a coder state s_i to every two-digit binary number from the $s(kT)$ columns (for example, $s_1 \equiv \{0,0\}$, $s_2 \equiv \{0,1\}$, etc.). The different binary numbers represent all possible states the coder could have been in and an appropriate labelling is shown in the first column, s_i(present). In a similar way the combinations of the present input, $D(kT)$ with the previous, $D((k-1)T)$ show all possible states the coder can move into. These two-digit binary numbers are given in columns $s((k+1)T)$. By substituting each one with the respective state, as they were defined for the first three columns of Table 4.18, the next state is determined as shown in the last column, s_i(next). The final result is contained in the relations specified by the first and the last three columns. They allow for the output, $X(kT)$ and the following state, s_i(next) to be determined for every input, $D(kT)$ and state, s_i(present). This in effect gives the table presentation of the coder. In the above example it is easy to see that the result corresponds exactly to Table 4.11, which was used for the definition and the spectral analysis of the Modified Duobinary technique.

4.2.1 The State-Definition Procedure

In general the suggested method allows the 'Code' specification table to be derived from either an explicit description or an algebraic definition of the coder rules. This is achieved through presenting the correspondence between all different combinations of input blocks of symbols which are used to produce the respective output symbols and by applying the technique demonstrated above. Indeed, there are no limitations to the general applicability of the reasoning used in the above example. This follows from the fact that no restrictive reference has been made at all to values of the parameters specifying the particular coding scheme. The number, r of past values of the input sequence, which is 2 in this case, can be any integer. The same is valid for the values of the input symbols and the length of the input word, N ($d = [0,1]$ and $N = 1$ for the above example). And finally, the state transitions are determined irrespective of the values and the parameters specifying the output sets, ($x \equiv X = [-1, 0, +1]$ and $L = 1$ in this case) as long as the code generates a valid output for every possible input.

The combinations from previous input values in the last example, which determine the output of the coder, can be viewed as the range of a parameter or a variable representing the coder states. In general it is possible to specify a number of parameters affecting the output of the coder, hence representing different coder states. Some examples are given below:

i) previous values of input and/or output symbols;
ii) the sign of the last polar input or output symbol;
iii) the accumulative (limited) sum of a finite or infinite number of previous input or output symbols;
iv) results from algebraic transformations applied to input and/or output symbols.

The number of variables, which form the combinations representing coder states, is not restricted and their ranges are determined by the specification of particular coding rules.

The suggested method for deriving the set of coder states and their relations in 'Code' tables is presented in general form as a procedure consisting of several steps. To simplify the analytical notation only three of the most common conditions, used in practical coding schemes, are specified as variables in determining the coder states. These are:

Δ	the combination of preceding blocks of symbols, which are used to determine the coder output[21],
Γ	the value of appropriately specified digital sum,
Λ	the sign of a suitably chosen input or output symbol.

The steps of the procedure can now be defined as follows:

Step 1: Specification of the range of values for the different parameters. The possible values and the number, r of preceding blocks $D((k-r)T)$, $D((k-r+1)T)$, ..., $D((k-1)T)$, $k = 0, \pm 1, \pm 2, \ldots$, on which the present output of the coder depends, are used to determine the range of Δ. The range of Γ is some set of integers while that of Λ contains two elements, '$+$' and '$-$'.

The variables specified in Step 1 depend on the input and/or the output of the coder, so they can be represented as functions, i.e. $\Delta(D_k)$, $\Gamma(D_k, X_k)$ and $\Lambda(D_k, X_k)$. The notation adopted for the arguments of these functions does not specify a particular range of values. It is used to indicate the sets, which comprise the range of the variables and also that the latter are determined for every kT, while their actual value may depend on a number of previous input and/or output symbols.

Step 2: All valid combinations of the parameters, specified in Step 1 and the present input, $D(kT)$ of the coder are formed, using the following structure[22]

[21] For some codes combinations of output symbols are used.

[22] The structures are simply collections of variables, placed in certain order and the adopted notation does not involve multiplication.

$$\Gamma(D_{k-1},X_{k-1})\ \Lambda(D_{k-1},X_{k-1})\ \Delta(D((k-r)T)\ldots D((k-1)T))\ D(kT) \equiv$$

$$\underbrace{\Gamma_{k-1}\ \Lambda_{k-1}\ \Delta_{k-1}}_{\text{(a)}}\ D(kT) \tag{4.2}$$

The combinations given by part (a) of (4.2) correspond to all possible past events in the coding process, which affect the output, $X(kT)$. In this respect (4.2) represents the output function of the coder, which has been defined by (3.4) in Chapter 3, i.e.

$$X(kT) = \mathfrak{B}[\Gamma_{k-1}\ \Lambda_{k-1}\ \Delta_{k-1}, D(kT)] \tag{4.3}$$

Expression (4.3) suggests that part (a) in (4.2) corresponds to all possible states of the coder, which leads to the next step.

Step 3: All distinct combinations, $\Gamma_{k-1}\ \Lambda_{k-1}\ \Delta_{k-1}$ are labelled as different coder states, s_i $(i = 1,\ldots, I)$ as follows

$$(s_i)_{\text{present}} \equiv \left\{\Gamma_{k-1}\ \Lambda_{k-1}\ \Delta_{k-1}\right\} \tag{4.4}$$

The number of possible states, I is given by $I = \gamma\lambda M^r$, where γ and λ are the numbers of possible values Γ and Λ, while M is the number of possible blocks of input symbols and r is specified in Step 1. (If any of γ or λ is zero, I is assumed 1.)

Up to this stage a code is specified with respect to all possible outcomes regarding combinations of input or output values and certain parameters associated with them. It is necessary to use the coder rules (usually given as some description of symbol transformations, algebraic expression, etc.) in order to identify the appropriate transitions between states, which are required for the completion of the 'Code' tables.

Step 4: The coder rules are applied appropriately for each combination specified by (4.2) in Step 1, to determine all possible values of the output, $X(kT)$ as well as the new values for the parameters Δ, Γ and Λ. The resulting parameter structure is presented below as a development of (4.2).

$$\Gamma_{k-1}\ \Lambda_{k-1}\ D((k-r)T)\ldots D((k-1)T)\ D(kT)\ \Gamma_k\ \Lambda_k \equiv$$

$$\Gamma_{k-1}\ \Lambda_{k-1}\ D((k-r)T)\ \underbrace{\Delta_k\ \Gamma_k\ \Lambda_k}_{\text{(b)}} \tag{4.5}$$

The result from Step 4 suggests the possibility to interpret part (b) in (4.5) as the states the coder goes into for every input, $D(kT)$ after being in the respective states determined from part (a) in (4.2). This leads to the last step, which is specified below.

Step 5: All distinct combinations, $\Gamma_k\ \Lambda_k\ \Delta_k$ are labelled as different coder states

$$(s_i)_{\text{next}} \equiv \left\{\Gamma_k\ \Lambda_k\ \Delta_k\right\} \tag{4.6}$$

according to the correspondence $s_i \leftrightarrow (\Delta\Gamma\Lambda)$ established in Step 3.

Finally, it is important to note the special use of the term $D(kT)$ in structures (4.3) and (4.5). In the former it is used to form all valid combinations of parameters determining the state of the coder, $(s_i)_{\text{present}}$ and the value of the present input, $D(kT)$ so that it becomes possible to compute the output $X(kT)$ and the new values of the parameters according to Step 4. At the same time $D(kT)$ is used in (4.5) to determine the parameter Δ_k, which is used to specify the states $(s_i)_{\text{next}}$ according to Step 5. The double role of $D(kT)$ means that it will always appear in structures (4.2) and (4.5) regardless of the existence of the parameter Δ. The latter, however, will determine whether $D(kT)$ should be included in part (b) of (4.5), hence in (4.6), or not.

The interpretation of parts (a) and (b) of structures (4.2) and (4.6) also requires special attention. While the values of parameter Δ are most likely to be numbers, the values of Γ and Λ can be arbitrary symbols, e. g. '$+$', '$-$', 'L' (for low level), 'H' (for high level), etc. This is why the combinations determined as states according to (4.6) should be carefully identified with the respective combinations specified by (4.4), regardless of the order of the variables Δ, Γ and Λ. This is of particular importance if the state definition method is presented in the form of a table, which may not be very easy to rearrange when proceeding from Step 3 to Step 4, but otherwise very convenient and useful.

In certain occasions it may be suitable to represent the combinations of variables as complete numbers. Such an arrangement allows to implement the five-step procedure in a software algorithm, capable of processing more complex coding schemes. For these cases structures (4.2) and (4.5) have to be developed separately in order to maintain the correct order and values of the digits, comprising the combinations of variables represented as numbers.

4.2.2 Practical Implementation of the State-Definition Procedure

The method of specification of coder states can be implemented in a generalised table form, similar to Table 4.18, used for the Modified Duobinary example, at the beginning of this section, by adopting the notation from the previous section. Such a table, however, requires the introduction of a large number of additional variables to represent the possible elements of parameter sets like Δ, Γ and Λ. Its explicit construction is too complex for the purposes of this presentation. It seems more appropriate and useful to complete the section with practical examples.

First, it should be noted that the definition of the coder states for the Modified Duobinary code, presented in Table 4.18, complies entirely with the specifications of the procedure developed in the previous subsection. The arrangement of structure (4.5) in table form has been achieved relatively easily as the only parameter which determines the coder states, is Δ. The range of values of Δ consists of all binary numbers represented by the last two input symbols, $D((k-2)T)\ D((k-1)T)$. Another specific feature of Table 4.18 is the double use of the present input value, $D(kT)$: firstly, it is combined with the present states $(s_i)_{\text{present}} \equiv \left\{D((k-2)T)\ D((k-1)T)\right\} \equiv \left\{\Delta_{k-1}\right\}$, to determine the coder output $X(kT) = \mathfrak{B}[\![\Delta_{k-1}, D(kT)]\!]$ and secondly, it is used to determine the next state through $(s_i)_{\text{next}} \equiv \{D((k-1)T)\ D(kT)\} \equiv \left\{\Delta_k\right\}$.

A different implementation of the state-definition procedure is illustrated with the example of the AMI code given below:

Step 1 in this case results in $\Delta = \Phi$, $\Gamma = \Phi$ (where Φ is the empty set) and $\Lambda = [+, -]$ is a set with elements the sign of the last non-zero output symbol.

Step 2 leads to the formation of structure (4.2) as $\Lambda_{k-1}\ D(kT)$. This allows the output $X(kT)$ and Λ_k to be determined for every value of the combination $\Lambda_{k-1}\ D(kT)$.

Step 3 produces the states of the coder, $(s_i)_{\text{present}} \equiv \left\{\Lambda_{k-1}\right\}$, which correspond to the conventional AMI definition. According this definition the coder states depend only on the sign of the last coded 1.

Step 4 develops the structure (4.5), which in this case is $\Lambda_{k-1}\ D(kT)\Lambda_k$, leading to the final stage.

Step 5 determines the states into which the AMI coder goes for every combination of a present state, as defined in Step 3, and an input symbol, $D(kT)$ as $(s_i)_{\text{next}} \equiv \{\Lambda_k\}$.

By substitution of the variables specified above with the respective values for the AMI code, it is possible to construct the table which is used to determine the state transitions for the FSSM model of this scheme.

s_i	Λ_{k-1}	$D(kT)$	$X(kT)$	Λ_k	s_i
s_1	$-$	0	0	$-$	s_1
s_2	$+$	0	0	$+$	s_2
s_1	$-$	1	+1	$+$	s_2
s_2	$+$	1	−1	$-$	s_1
present	$s(kT)$	input	output	$s((k+1)T)$	next

Table 4.19 State definition scheme for the AMI code

No doubt the AMI-'Code' specification derives directly from Table 4.19.

Another example of the implementation of the suggested coder-specification method is the 1T2B code derived from 3B2T-RBS coder rules, [26]. These rules are given in the usual descriptive form as follows:

Input 0	is coded as 01, if the last coded symbol is 1 and as 10, if the last coded symbol is 0.
Input 1	is coded as 00 and 11 alternatively.
Input 2	is coded as 01, if the last coded symbol is 0 and as 10, if the last coded symbol is 1.

In this case there are two state-definition parameters. Their specifications, however, are different from the ones used so far as shown in the procedure below.

Step 1 gives the following two variables:

$\Gamma = [0,1]$	the values of the last coded output symbol,
$\Lambda = [00,11]$	represents the last output block of symbols used to code an input 1.

It should also be noted that the output does not depend on previous input symbols, therefore $r = 0$ and Δ is an empty set.

Step 2 produces the structure $\Gamma_{k-1}\ \Lambda_{k-1}\ D(kT)$, corresponding to (4.2), where the value of the input symbol, $D(kT)$ is taken from the set $D = [0,1,2]$. The output, $X(kT)$ of the coder, where $X(kT) \in [00, 01, 10, 11]$, is determined according to the description of the coder rules, for every combination of values of the above structure. This allows the values Γ_k and Λ_k to be determined. The results of this step are collected in columns 6 and 7 of Table 4.20.

Step 3 leads to the specification of the coder states according to the expression $(s_i)_{\text{present}} \equiv \left\{\Gamma_{k-1}\ \Lambda_{k-1}\right\}$, given in the first column of Table 4.20.

Step 4 results in the formation of the extended structure $\Gamma_{k-1}\ \Lambda_{k-1} D(kT)\ \Gamma_k\ \Lambda_k$, which corresponds to the values from columns 2–7 in Table 4.20.

Step 5 gives the final result $(s_i)_{\text{next}} \equiv \left\{\Gamma_k\ \Lambda_k\right\}$, which is shown in column 8 of Table 4.20.

s_i	Γ_{k-1}	Λ_{k-1}	$D(kT)$	$X(kT)$	Γ_k	Λ_k	s_i
s_1	0	00	0	10	0	00	s_1
s_1	0	00	1	11	1	11	s_4
s_1	0	00	0	01	1	00	s_3
s_2	0	11	0	10	0	11	s_2
s_2	0	11	1	00	0	00	s_1
s_2	0	11	0	01	1	11	s_4
s_3	1	00	0	01	1	00	s_3
s_3	1	00	1	11	1	11	s_4
s_3	1	00	0	10	0	00	s_1
s_4	1	11	0	01	1	11	s_4
s_4	1	11	1	01	0	00	s_1
s_4	1	11	0	10	0	11	s_2
present	$s(kT)$		input	output	$s((k+1)T)$		next

Table 4.20 State definition for the 3B2T-RSB code (version 1T2B)

The suggested method for definition of the coder states and the respective state transition scheme can be used in the specification of the FSSM model for virtually all fixed-length, finite-memory block-codes of practical interest. It has been successfully applied to most existing coding techniques of moderate and high levels of complexity, included in the spectral analysis results. The 5-step implementation procedure described in this section is a useful tool for constructing the table presentation of the coder functions required in the spectral analysis model from Chapter 3. As the main analytical expressions (3.25) are based on the matrix presentation of the FSSM model of a coder and the matrix form of the coder functions derives directly from the 'Code' definition table, the significance of the method presented here can be appreciated by noting that it provides the link between the descriptive code specifications and the systematic approach in their comparative assessment and analysis.

5. THE FIXED-LENGTH BLOCK CODE GENERATOR

A major result from the unified approach in specification and analysis of coding techniques is the development of a new method for design of digital line codes. It is based on a further enhancement of the computational algorithm, used to analyse the frequency characteristics of coded sequences. The method allows for systematic generation and assessment of a large variety of code structures. This is accomplished through a generalised specification of their initial parameters (the sets of input/output symbols and coder states) and spectral analysis of all coding schemes which are determined as valid according to predefined conditions.

The first section of this chapter gives the underlying arguments for arranging different types of code structures into categories and classes. A particular type is assumed to be represented by a set of specification parameters. Most types discussed below include one or more of the practical examples analysed in the previous chapter. The modified computational algorithm is outlined in brief and the results from the spectral analysis of several groups of codes are presented. These results are summarised as the basis for the method of creating new coding schemes.

5.1 Categories of Digital Code Structures

The concept of combining different codes into categories according to their specification is based on the existence of a number of different 'Code' tables for every given set of symbols and states. This can be illustrated by assuming a matrix $\mathfrak{X}$ collecting all output words, from the table in Fig.4.2, as follows

$$\mathfrak{X} = \begin{bmatrix} X_{11} & X_{12} & \cdots & X_{1I} \\ X_{21} & X_{22} & \cdots & X_{2I} \\ \vdots & \vdots & \vdots & \vdots \\ X_{M1} & X_{M2} & \cdots & X_{MI} \end{bmatrix} \tag{5.1}$$

All entries of this matrix are taken from the code dictionary, X as defined in the table of Fig.4.1, i.e. $X_{mi} \in X = [X_1, X_2, \ldots, X_K]$, where $m = 1, \ldots, M$, $i = 1, \ldots, I$ and K is the total number of output words. It is possible to construct another matrix $\widehat{\mathfrak{X}}$ so that at least one of its elements is different from the element with the same indices in $\mathfrak{X}$, i.e. $\widehat{X}_{mi} \neq X_{mi}$ where $\widehat{X}_{mi} \in X$, too. If the elements of $\widehat{\mathfrak{X}}$ are used in the same table instead those of $\mathfrak{X}$, the resulting code definition will be different from the one corresponding to the original table.

It is not difficult to see that the variety of coding schemes, corresponding to a particular set of symbols and states, increases rapidly with the sizes of D, s and X. In order to perform a systematic assessment of the codes resulting from possible combinations of initial parameters, the latter are suitably redefined to accommodate the specification of coding structures in a more general form. Such structures are presented and analysed in groups which are further referred to as categories.

5.1.1 A General Description of Code Structures

The fundamental quantities adopted to describe different categories of codes are the integers M, I and K which correspond to the size of the input dictionary D, the set of coder states s and the size of the output dictionary X as defined in the general coder model (Chapter 3). Thus all coding schemes employing I states, which transform M blocks of input symbols into a subset of the K blocks of output symbols are denoted as $DMSIXK$. In some cases the generality may lead to specification of more or less abstract code structures of very little practical value in terms of line coding. This has been taken into account and where necessary in the following discussion, such cases are discounted. For the purposes of this presentation the range of values $M = \{2,4\}$, $I = \{1,2,3,4\}$ and $K = \{2,3,4,8\}$ has been chosen so that the resulting categories cover most coding techniques of practical importance. As an example, the AMI code, which is specified in the previous section, has a structure of two input words, three output words and two states $(M = 2, I = 2, K = 3)$. Therefore it is considered in the category of $D2S2X3$ codes.

Obviously the variety of combinations is very large, even with the relatively small range of values for the sizes of the three sets D, s and X. However, as will be shown later, a considerable proportion of them do not correspond to code specifications of practical interest. Those which are considered relevant to the presented analysis are discussed separately below. It should be noted that, although the basic point of view in this discussion is spectral analysis of coded signals, the general specification technique may be applied to describe various systems in the area of digital signal processing, as long as they can be modelled as a FSSM transforming one set of symbols into another.

The way of denoting the code categories as $DMSIXK$ has also been chosen out of convenience. Although it describes adequately all examples of existing schemes, some ambiguity may arise about the type of codes included in a given category, when the numbers M and K are exact powers of integers, i.e. in the cases of $M = \alpha^N$ and/or $K = \beta^L$. The numbers α, N, β and L are associated with the symbol alphabets and the lengths of the input/output words as they were defined for the coder model in Chapter 3. In the case of a one-state category, for example, denoted as $D2S1X4$ it is not possible to specify the group of coding schemes unless the exact number of symbols in the output alphabet is known. Two types of codes can be referred to by this notation. The first one is the case of binary codes where $x = [x_1, x_2]$ and the set of output words, X consists of all possible combinations of two-symbols. The second type involves all codes with output blocks of length $L = 1$, which corresponds to an output alphabet $x = [x_1, x_2, x_3, x_4]$.

The possible ambiguity described above can be resolved by using an extended notation for the categories of code structures. One solution to this problem is $D\alpha NSIX\beta L$ which conveys information about the symbol sets (α, β), as well as the sets of words (N, L). However, it has been considered unnecessary to adopt the extended form of notation here, because most structures included in the analysis can be specified uniquely in the initially suggested way. In the very few cases where this is not achieved additional specification is provided explicitly. At this stage it is sufficient to note that all codes of practical interest[1] considered in the thesis are based on the binary set of input symbols, $d = [0, 1]$ which allows the possible input dictionaries to be uniquely specified by the length of the input word or equivalently by the number of input blocks of symbols. As for the set output words only two cases are included in the generalised assessment, namely binary and ternary signalling. In the first case the number of output words is taken over the range of $\{2, 4, 8, \ldots\}$, while in the second case only $K = 3$ has been considered. The conditions specified above provide for the use of $DMSIXK$ in a general description of the coding structures without ambiguity.

Following the conventions introduced in this section it is necessary to give a more detailed description of several terms which have been used previously without formal definition. First it should be noted that the expression '*complexity* of a code' has been applied with respect to the numbers M, I and K in the following sense: the bigger these numbers the higher the complexity. As the sizes of the three sets specifying a code may vary independently[2], it is also necessary to introduce some priorities in the way complexity is related to M, I and K. The number of states, I has already been suggested to determine the highest level of complexity, (Section 4.1.2.2). This can be interpreted by considering $DMSI_1XK$ codes to be of higher complexity than $DMSI_2XK$, if $I_1 > I_2$. In a similar sense for

[1] The 1T2B scheme is the only exception.

[2] Except for certain practical restrictions to the minimum number of output words for a given number of input blocks of symbols.

$I_1 = I_2$ the second and the first complexity levels are determined through M and K respectively. Three of the codes discussed earlier, can be given as an example of complexity relations. The Duobinary, which is a representative of the $D2S2X3$ group, is considered of higher complexity than the Manchester scheme ($D2S1X4$) and of lower complexity compared to the CMI code ($D2S2X4$). Later it is shown that, in fact, CMI is not the best representative of its category, although it has been widely implemented in practice.

The other two terms which have been used in the comparative assessment of codes in the previous section are *structural* and *spectral* features of coding schemes and the respective PSD functions (Section 4.1.2.3). Structural features refer to the overall specification of the input/output symbol and word sets and the number of states in particular. These features also relate to a special characteristic which derives from the code definition and reflects the existence of output words which correspond to more than one input word for a particular 'Code' table. With respect to the technical implications from the implementation of codes with such a feature, it is denoted as state-dependent decoding. As for the spectral features, no precise analytical definition has been considered necessary for the purposes of the presented comparative assessment of digital coding techniques. Some of the characteristics of the spectral density function, which are commonly used in discussions of frequency analysis results, are adopted in their most general form as follows[3]:

- bandwidth of the coded signal;
- shift of a spectrum towards lower or higher frequencies;
- suppression of low- or high-frequency components in the spectra;
- presence of a d.c. and/or other discrete components (spectral lines).

5.1.2 Compiling Categories of Coding Schemes

The basic idea of combining code structures in groups has been implemented in a modified version of the main computational algorithm, described in Chapter 3. The main enhancement of the software (see Appendix-A) is the part which generates the complete set of initial parameters required for the specification of all codes within a given category and their subsequent combination in particular structures according to certain predefined conditions. The easiest way to describe how the different code specifications are compiled is by following the stages of the computational procedure as presented below.

[3] All spectral features are relative, i.e. assessed in terms of comparison of two or more PSD functions.

1) The two sets of input and output symbols are declared in the same form as for the original program. The same conditions also apply with respect to whether the values of the symbols are given explicitly or the sets of binary words are compiled automatically through the numbers N and L, which determine the sizes of the input and output words respectively. The most significant result from this stage is the availability of the complete sets D and X of M input and K output blocks of symbols for a particular category $DMSIXK$.

2) This stage has the important function of generating the matrices S_m and Z_m deriving from all possible combinations of M input words, I states and any subset of the K output words which can result in a 'Code' table. There are two possible approaches to constructing the state-transition and the output matrices. The first one requires all possible matrices S_m and Z_m for given I and K to be specified explicitly at the beginning of the program. It can be used for relatively small numbers of combinations by applying the following reasoning:

a) A state-transition matrix, S_m has I rows

$$S_m(i) = [S_m(i,1), S_m(i,2), \ldots, S_m(i,I)] \tag{5.2a}$$

where $i = 1, \ldots, I$. Every row has exactly one nonzero element of 1. The number of different rows is I and the number of all possible matrices S_m constructed from any combination of rows $S_m(i)$ is I^I.

Example: For $I = 3$ there are three rows (5.2a) given by

$$S_{m1} = [0\ 0\ 1],\ S_{m2} = [0\ 1\ 0] \text{ and } S_{m3} = [1\ 0\ 0]$$

The number of matrices of size 3×3, which can be constructed from these rows is $3^3 = 27$.

b) An output matrix Z_m also has I rows and every row is an output block of L symbols, i.e.

$$X_m(i) = [X_m(i,1), X_m(i,2), \ldots, X_m(i,L)] \tag{5.2b}$$

where $i = 1, \ldots, I$ and $X_m(i) \in X$. As there are K different output words in X, the total number of possible matrices Z_m collecting any combination of I output words is K^I.

Example: For a category of binary codes denoted as $D2S2X4$ all possible rows for the matrices Z_m are

$$X_1 = [0\ 0], X_2 = [0\ 1], X_3 = [1\ 0] \text{ and } X_4 = [1\ 1]$$

The number of possible output matrices in this case is $4^2 = 16$.

Obviously it is very tedious to specify all possible matrices for I and K bigger than 3. This is why the second approach involves generation of the state-transition and the output matrices automatically and requires only one additional set R, collecting all possible rows (5.2a). In the software routine R is specified as a $I \times I$ matrix, just like the sets D and X and can also be generated automatically for any given number of states.

3) The third stage of the computational procedure involves the evaluation of the PSD function through the original software routine, described in Chapter 3. This part of the program is executed within several nested 'for' loops. Their number is $2M$ and is determined by the size of the set of input blocks of symbols, D. The outer loops select a combination of M state-transition matrices, S_m. At this point the TPM matrix S is computed and its validity is verified as described in Section 3.3.1. The selection of a combination of M output matrices, Z_m through the inner loops proceeds only for a valid TPM. The final result of this stage is the construction of the two matrices SM and ZM, which specify a unique code as defined in the general 'Code' table from Section 4.1.1. The plot of the respective spectral density function is displayed and the computation continues with the construction of another coding structure.

4) The last stage of the algorithm organises the collection of the analysis data and its appropriate storage. Two main problems have to be solved at this stage. The first is to avoid accumulation of repetitive identical results without losing information about their origin. The second problem is to ensure efficient data storage which is important for two reasons:

- the amount of computational results increases very quickly with $\{M, I, K\} > 3$;
- the assessment of the final data requires manageable access to the spectral density data and easy identification of the various coding structures.

This problem is purely technical and it can be tackled according to the available hardware and software environment.

The first three stages of the computational routine indicate the most essential elements of the process of compiling the different coding structures which correspond to a particular category *DMSIXK*. The total number of these structures was shown to depend on the number of all possible state-transition and output matrices, S_m and Z_m. The estimation of the size of the complete set of codes comprising a given category also involves the number of input words, M. As described in 3) above, a combination of M state-transition matrices and M output matrices is required for every generated coding scheme. Therefore the number different sets of M matrices S_m is $\left(I^I\right)^M$. Analogously, for Z_m this number is $\left(K^I\right)^M$. In the most general case (regardless of whether the combinations represent valid codes) all possible sets of Z_m matrices can be combined with every possible set of S_m matrices. Then the total number of different structures is given by the following expression:

$$\textit{number of DMSIXK code combinations} = (I^I)^M \times (K^I)^M \tag{5.3}$$

The first impression given by the above formula is that assessment of the analysis results for categories of higher complexity codes soon becomes impractical due to the large number of combinations. Fortunately this is not the case at least up to values of M, I and K which specify categories including most of the fixed-length block codes which are presently implemented in digital communication systems. More details are revealed in the next subsection. At this point it is sufficient to mention that a considerable proportion of structures generated for the combinations of state-transition and output matrices, as described above, do not represent valid or meaningful codes. One of the reasons for this is the existence of sets of matrices S_m which do not correspond to a valid TPM (according to conditions which have already been discussed). Another reason for compiling impractical coding schemes is the possibility to select a set of matrices Z_m which do not represent a realistic coded sequence in terms of information transmission. A simple example is the case when every row of all output matrices (for any code) is the same output block of symbols. In general an invalid line coding scheme results from all combinations where the number, J of selected output words is smaller than the number of input words, M. Finally, a third reason to expect a significant reduction in the amount of analysis data is the high proportion of repeating spectral density functions which in most cases correspond to equivalent code structures.

Obviously, by the use of simple verifications the software routine can be adjusted to select specific types of code structures. This is achieved by checking for various conditions imposed on the validity of combinations of S_m and Z_m matrices. In general, it can be expected that even with categories of relatively high order (that is, the sizes of D, s and X are greater than 3) a reasonable number of criteria can be devised so that the generated family of codes is not prohibitively large for analysis. An additional possibility to keep the assessment of various groups of codes within manageable limits is to apply certain restrictions derived from practical considerations. It can be noticed that in many cases of existing coding techniques the S_m and the Z_m matrices feature various kinds of symmetry. A typical example are the two state-transition matrices of the CMI code given below.

$$S_1 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \qquad S_2 = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$$

Most schemes with two states and some alternating feature, like AMI, Differential, 3B4B, etc. exhibit this or a similar type of symmetry. Other codes, like the Modified Duobinary, can be taken as examples of symmetry in the output matrices:

$$Z_1 = \begin{bmatrix} 0 & 0 & -1 & -1 \end{bmatrix}' \qquad Z_2 = \begin{bmatrix} +1 & +1 & 0 & 0 \end{bmatrix}'$$

The analytical relation between certain features of the spectral density and different patterns of matrix elements requires a substantial amount of research and can be determined by some repetitive analysis methods. However, for practical design purposes it is considered appropriate to restrict the number of generated structures by specifying a particular combination of S_m or Z_m matrices which is expected to produce a particular type of results. For any given category $DMSIXK$ the final number of structures selected for assessment can be easily evaluated through expression (5.3). Thus, if only r_0 of all state-transition matrices are required to appear in any of the codes to be analysed, this number is given by

$$\frac{M!}{r_0!(M - r_0)!} \times \left(I^I - r_0\right)^{M - r_0} \times (K^I)^M, \text{ for } r_0 \leq M \tag{5.4}$$

The variety of possible ways to determine the group of structures related to certain design requirements, as well as the particular category of interest is substantial. In this respect it should be noted that the levels of complexity, which have been adopted as an overall description of the upgrading features of different codes, do not impose any particular order to the formation or the analysis of the code categories. The decision to consider the number of states to be the highest level of complexity has been taken on empirical basis. It reflects the following two tendencies in the practical designs of coding techniques:

- implementation of relatively large input and output sets of words, ($M, K > 4$) and a small number of coder states ($I < 4$), typical examples being 5B6B, 4B3T, etc.;
- use of coders with more than three states and input/output sets of words whose sizes rarely exceed four[4], where Modified Duobinary and Miller codes are typical representatives.

In summary, the following order of complexity levels , associated with the various categories of coding structures, is adopted to facilitate the discussion in subsequent sections:

The *first (highest) level* of complexity is assumed to be the number of coder states, I. A coding structure of the type $DM_1SI_1XK_1$ is considered of higher complexity than $DM_2SI_2XK_2$, if $I_1 > I_2$, regardless of the values of M and K.

The *second level* of complexity is represented by the number of input words, M. All codes of the type $DM_1SI_1XK_1$ are considered of higher complexity than those of type $DM_2SI_2XK_2$, only if $M_1 > M_2$ and $I_1 \geq I_2$ (regardless of the value of K).

The *third level* is represented by the number of output blocks of symbols, K. Analogously a coding scheme from the category $DM_1SI_1XK_1$ is of higher complexity order than the ones from $DM_2SI_2XK_2$ provided that $K_1 > K_2$, only if $M_1 \geq M_2$ and $I_1 \geq I_2$.

The presentation of the practical results from the analysis of several complete categories of coding structures is given in the next section by following an increasing complexity order, according to the assumptions made above.

5.2 Generation and Analysis of Codes From Basic Categories

This section reveals the full scale of generalisation achieved through the adopted theoretical model of a coder and the suggested unified approach in specification and assessment of digital coding techniques. The presentation of the spectral analysis results and the discussion are based on the general definition of

[4] The numbers used here are only typical and do not represent any strict limitations.

code categories developed in the previous section. The major consequence from the implementation of a common structural description of codes is the availability of a mechanism for generating the complete set of schemes corresponding to a predefined group. The core of this mechanism is the specification of all possible state-transition and output matrices for a given set of initial parameters which correspond to a particular category of fixed-length block codes.

The basic set-up of the generating conditions can be illustrated by taking the simple example of the $D2S2X2$ category. The total number of possible structures is evaluated straightforwardly by using (5.3) to $(2^2)^2 \times (2^2)^2 = 256$. After applying the definitions given in stage 2) of the enhanced computational procedure, described in the previous section, the complete sets of S_m and Z_m matrices are determined as follows:

$$(S_m)_1 = \begin{bmatrix} 0 & 1 \\ 0 & 1 \end{bmatrix} \quad (S_m)_2 = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \quad (S_m)_3 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \quad (S_m)_4 = \begin{bmatrix} 1 & 0 \\ 1 & 0 \end{bmatrix}$$

$$(Z_m)_1 = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \quad (Z_m)_2 = \begin{bmatrix} 0 \\ 1 \end{bmatrix} \quad (Z_m)_3 = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \quad (Z_m)_4 = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

The above matrices have been constructed over the range of initial parameters specified to the program as the input and output sets of symbols or words, (or at least their sizes). Subsequently the computing algorithm forms all 256 combinations of matrices

$$\mathrm{SM} = \begin{bmatrix} (S_1)_{\sigma_1} \\ (S_2)_{\sigma_2} \end{bmatrix} \qquad \mathrm{ZM} = \begin{bmatrix} (Z_1)_{\zeta_1} \\ (Z_2)_{\zeta_2} \end{bmatrix}$$

where $(\sigma_m, \zeta_m) \in \{1,2,3,4\}$ for $m = 1,2$. Each of these combinations formally corresponds to a 'Code' table and specifies a particular coding structure. Clearly not all combinations correspond to meaningful schemes, e.g. a code with output matrices $Z_1[0\ 0]'$ and $Z_2 = [0\ 0]'$ produces a sequence of zeros regardless of the input signal. As it has been explained in the previous section various conditions can be applied to select only the structures of interest. One of the obvious restrictions, which can be imposed to avoid computation of coding schemes of little practical value, is the requirement to perform the analysis only for sets of matrices ZM collecting output words whose number is at least equal to the number of input words, i.e. $J \geq M$. The main reason for presenting the computational routine in its most general form is to suggest possibilities for implementation of the assessment technique to systems transforming digital sequences used for purposes other than line coding.

In summary, the enhanced computational procedure, based on the spectral analysis model given in Chapter 3, allows the design of fixed-length block codes of any type by specifying the sets of input and output blocks of symbols and the number of coder states. These initial parameters identify a category of coding structures, all of which can be generated and analysed. In the general case the design process involves inspection of the analysis results and selection of the combinations matching particular requirements. Two techniques for control of the type of anticipated results and their overall quantity can be applied:

i) Imposing restrictions on
– the validity of structures, produced from combining initial parameters,
– the number and the size of different subsets of initial parameters.

ii) Specifying the design limitations of interest over the range of analysis data, e.g. a certain range of frequencies.

The completing stage of the basic design procedure consists of identification of the results which are most suitable to the particular application and constructing the respective coding schemes through their 'Code' table specification. (The latter corresponds uniquely to the combination of SM and ZM matrices which has produced the spectral analysis results of interest.) Further details and examples of practical implementation of the method for systematic design and assessment of coding structures are given in the following subsection through the presentation of results for a number of code categories.

5.2.1 *DMSIXK* Codes – Analysis Results

The number of block-coding schemes of the type *DMSIXK*, as defined in section 5.1.1, is infinitely large, therefore an exhaustive survey is practically impossible. The complete investigation becomes a formidable task even for relatively small values of the numbers M, I and K. However, the analysis results produced for a range of categories with manageable levels of complexity are shown to provide an amount of data sufficient for most design objectives in practical communication systems. A further proof of this claim is the fact that virtually all existing line coding techniques are covered by the categories included in this presentation. These categories are given below in an order of increasing complexity, described in the previous section, and arranged in subsections with respect of the number of states, I.

The formal specification of coding types requires a few words about categories such as $D1S1XK$. In terms of information transmission it obviously has no practical meaning. Formally, however, it is possible to use such a description in case of a systems which represents the simple function of transforming a sequence of identical symbols into some other sequence of identical symbols. There are other possible specifications which do not correspond to realistic coding structures (at least in terms of line coding). To avoid going into unrelated topics further comments on such structures are omitted from the discussion.

For similar reasons another type of coding schemes, with one state, $I = 1$ and word sets identical to the symbol sets ($D \equiv d, X \equiv x$), are not included in the presentation. In their basic form such schemes can be viewed as 'identity codes', if $D \equiv X$ and every input symbol is mapped into itself, or simple transformation systems which convert the input symbols into different output symbols. These ideas are illustrated in brief with the comments about the case of $D2S1X2$ which is mentioned in the following section for the sake of completeness with respect to the comments from Section 2.3.1.

5.2.1.1 Coding Structures With a Single State – $DMS1XK$

Having excluded the lowest level of complexity and the range of schemes other than binary as not closely related to the main topic of the presentation, three sample categories have been selected to outline the most typical features of this group of codes. All structures from these categories have the same symbol sets, $d = [0,1]$ and $x = [0,1]$. The essential difference between them are the respective sets of words which are produced as combinations of the lengths of the input and output blocks given by $N = 1,2$ and $L = 2,3$.

$\boxed{D2S1X4}$

This category includes the basic coding schemes, usually denoted as 1B2B formats, i.e. $N = 1$ and $L = 2$. Their common specification is given in Table 1.1:

D2S1X4 - Symbols

input	output	
$d = [0,1]$	$x = [0,1]$	
$D_1 = 0$	$X_1 = 0\ 0$	$X_2 = 0\ 1$
$D_2 = 1$	$X_3 = 1\ 0$	$X_4 = 1\ 1$
$N = 1,\ M = 2$	$L = 2,\ K = 4$	
$s = [s_1]$		

Table 5.1

The respective 'Code' table can only be defined in a general form as it formally corresponds to 16 different schemes. (Not all of them, though, represent practical code structures as indicated in the previous subsection.)

This is why an explicit specification of the coding rules in table form cannot be more informative than the general definition from Fig. 4.2. The only difference for this and the other categories of codes presented below is the substitution of I and M with the corresponding numeric values.

The result from the spectral analysis of all structures included in this group are shown in Fig. 5.1. As expected they look familiar and indeed the three spectral density plots (2, 3 and 4) correspond to the basic coding schemes defined earlier as RZ, NRZ and Manchester, respectively.

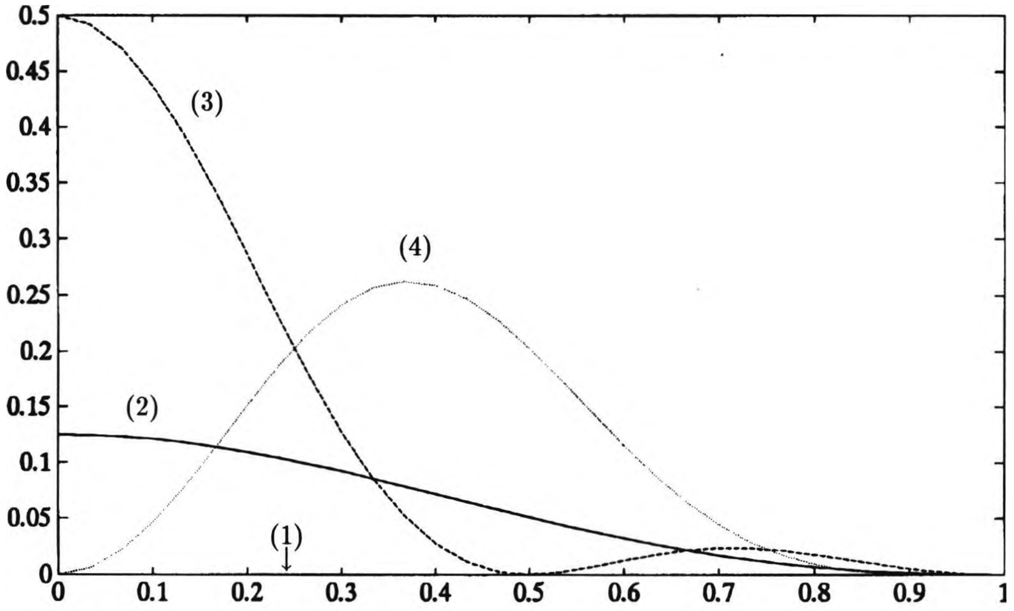


Fig. 5.1 PSD plots representing all codes from the $D2S1X4$ category

The numbers used to indicate the separate plots are taken from the table of equivalence, which is generated during the simulation of the various coding structures from a particular category. This table shows which codes within the same specification group have identical PSD functions. The results for the $D2S1X4$ type are given in Table 5.2.

Type of PSD plot	Simulation number of code structure
(1)	1, 6, 11, 16,
(2)	2, 3, 5, 8, 9, 12, 14, 15,
(3)	4, 13,
(4)	7, 10,

Table 5.2 The PSD simulation order

The numbers from the first column are assigned sequentially to every PSD in the process of generating and analysing the codes. Every one of these numbers corresponds to a distinct spectral density function. The simulation number of a code structure (the second column) indicates the sequential order of simulation for the various combinations of state-transition and output matrices and is used to identify the structures which have produced a particular PSD plot[5]. For example, the 2nd, 3rd, 5th, etc. schemes produce spectrum (2), which corresponds to the RZ code. This is not difficult to explain as all of them use the same type of output dictionary, $X^{(2)} = [X_1^{(2)}, X_2^{(2)}]$, where $X_1^{(2)}$ consists of two identical symbols and $X_2^{(2)}$ consists of two different symbols. The 'equivalence' feature which is attributed to codes with identical spectra can be illustrated by taking the second and the third combinations of output matrices, $(Z_1 = [0,0], Z_2 = [0,1])$ and $(Z_1 = [0,0], Z_2 = [1,0])$ respectively. The latter is clearly the RZ structure used in the spectral analysis from Section 4.1.2, while the former is just another version of this code which uses a different code word, $X_2 = [0,1]$.

In practice very little effort is needed to identify any coding scheme through the number of the selected PSD and the information provided in the form of Table 5.2, because the necessary data is automatically generated for every simulated category. The group of codes corresponding to the spectral density function (4) represents the Manchester scheme, whose output dictionary consists of the two possible blocks $X_1^{(4)} = [0,1]$ and $X_2^{(4)} = [1,0]$. (The second combination is from the same blocks in inverse order.) Similarly, the 'PSD' denoted as (1) corresponds to all structures for which $X_1^{(1)} = X_2^{(1)}$. Obviously these are invalid codes as they result in mapping both input words into the same output block of symbols. This is why their spectrum is a straight line coinciding with the horizontal axis.

Finally and not surprisingly, the PSD plot given as (3) represents the 'no coding' formats specified as NRZ in the preceding chapters. It is not difficult to see that for an output dictionary where both words consist of two identical symbols (but different for each word)[6], the respective scheme can be viewed as transformation of each input symbol into one output symbol. In such a case, however, the output dictionary becomes one of two possible words only and indicates that this type of code in fact does not belong to the category under consideration. Indeed, the lowest complexity group which can accommodate the

[5] The correspondence in this case derives from the sequential order of generating the combinations of output matrices, Z_1 and Z_2. Thus, the following sets have been used for the first three structures simulated for this particular example: $Z_1 = [0,0]$, $Z_2 = [0,0]$; $Z_1 = [0,0]$, $Z_2 = [0,1]$; $Z_1 = [0,0]$, $Z_2 = [1,0]$.

[6] 00 and 11 in this case.

NRZ scheme is $D2S1X2$ and it contains only two practical transformations, identity and inversion. The former can be viewed as mapping of the input sequence into itself, while the latter inverts every input symbol. The other two possible combinations of output matrices for $D2S1X2$ result in $Z_1 = Z_2$, which is not a real code. This situation does not affect the results from the analysis. It only highlights a computational peculiarity which can be easily controlled through specifying appropriate conditions in the software algorithm. The sets of code words [0,1], [00,11], [000,111], ... are typical examples, which indicate that identical coding structures will appear in the respective categories, $D2S1X2$, $D2S1X4$, $D2S1X8$, ... To preserve the generality in the presentation of the code design procedure and the possibility of independent assessment of various categories, some simple structures have been repeatedly generated in the analysis of higher level categories.

Apart from variations in the structure of individual blocks of output symbols, which leads to equivalent codes with identical spectral density functions, there is another feature common to all schemes from the same category. This feature indicates a relation between M different structures which can be interpreted as 'inversion'. It appears in its most basic form for $M = 2$. The simplest version has been mentioned above with respect to the NRZ scheme. Other examples of inversion are the two Manchester types, simulation numbers 7, ($Z_1 = [0,1]$, $Z_2 = [1,0]$) and 10, ($Z_1 = [1,0]$, $Z_2 = [0,1]$), the two RZ types, 3 and 9, etc.

In general the two features, 'equivalence' and 'inversion', specified for coding schemes from one and the same category may take various and more sophisticated forms of relation between the structures of a feature group. In any case, the significant result is the identity of the spectral densities of all codes within such a group. The essential difference between sets of structures exhibiting the two features is that inversion results from various ways of mapping the input words onto the same set of output words, while equivalence refers to structures with the same PSD using output sets which differ at least in one word. Although all codes from the $D2S1X4$ category have been discussed in previous sections, they have been used again to simplify the introduction of the code generating algorithm and demonstrate the advantages of the systematic assessment and analysis based on the definition of code categories. The benefits of this approach are further revealed through the results obtained for structures of higher complexity levels, which are presented in the following sections.

$\boxed{D2S1X8}$

This category is one level of complexity higher than the previous (considering binary input and output symbol sets only). All codes from this group map one input symbol into a block of three output symbols, i.e. $N = 1$ and $L = 3$.

$D2S1X8$ - Symbols

input	output	
$d = [0,1]$	$x = [0,1]$	
$D_1 = 0$	$X_1 = 0\ 0\ 0$	$X_5 = 1\ 0\ 0$
	$X_2 = 0\ 0\ 1$	$X_6 = 1\ 0\ 1$
$D_2 = 1$	$X_3 = 0\ 1\ 0$	$X_7 = 1\ 1\ 0$
	$X_4 = 0\ 1\ 1$	$X_8 = 1\ 1\ 1$
$N = 1,\ M = 2$	$L = 2,\ K = 8$	
$s = [s_1]$		

Table 5.3

The general 'Symbols' specification is given in Table 5.3. The 'Code' definition table is not shown for the same reasons as those given for the previous category. This time it corresponds to 64 structures. Again a certain number of them are of no practical value, but in this case such codes are omitted from the discussion altogether. The comments about the common table presentation and the invalid structures, as they have been made for the previous category, will remain unchanged for all categories and therefore will not be repeated further in the presentation.

The code generating algorithm applied for the $D2S1X8$ category produced nine different spectral density plots. None of them corresponds to a popular line coding structure, basically because the efficiency of a 1B3B scheme is low and in most cases a threefold increase in the signalling rate would be required. The analysis results, given in Fig. 5.2, are interesting mainly from a research point of view, but some of them can have practical value, too.

The table of equivalence can be constructed in the same way as for the previous category. It is not shown, because the relations between the codes within the group are very similar to those already discussed for the $D2S1X4$ schemes. Other types of relations can be revealed in this case by comparing coding structures from the two categories. The PSD plot number (5) needs no comment as it turns out to be the basic NRZ format using X_1 and X_8. An obvious candidate for investigation is PSD number (7) as it provides the best frequency distribution for the most common requirements of line coding (small low-frequency components and a shift away from the high frequencies. One of the combinations producing this spectrum employs output words 001 and 100. These two blocks of symbols can be viewed as corresponding to some type of a Manchester format as they introduce a transition between the start and the end of an input symbol time interval, T_d. The difference, however is that the transitions do not appear in the middle of the time interval. In some respect the code producing PSD (7) can be considered a better alternative to the Manchester code as it requires 1.5 times the bandwidth of the input sequence, while Manchester doubles that bandwidth. This result illustrates how the combination of the analysis algorithm with the suggested classification allows useful coding schemes to be found where otherwise they do not seem likely to appear.

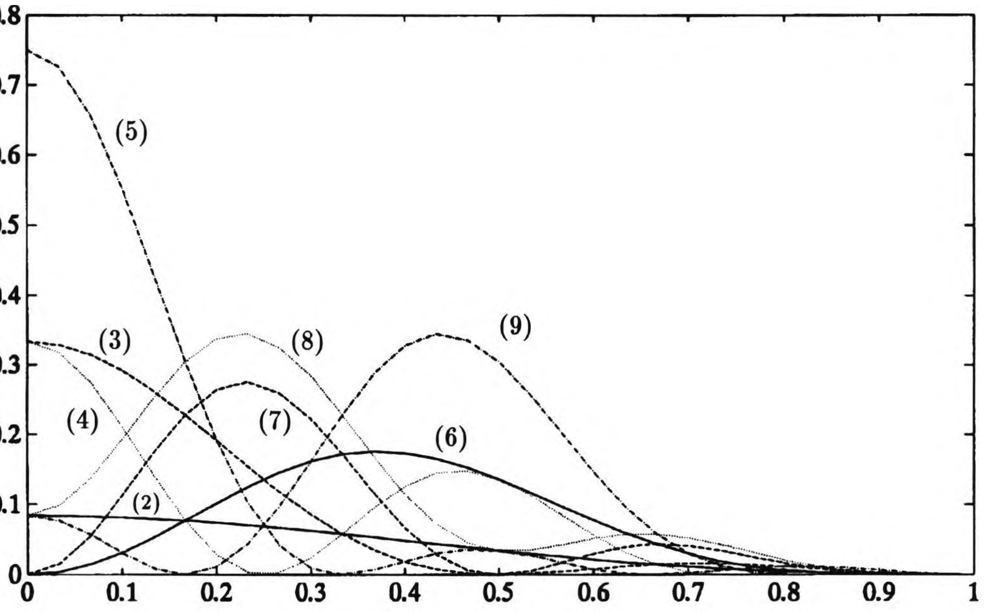


Fig. 5.2 The PSD plots for the $D2S1X8$ category

Another interesting plot is number (6) and one of the codes producing this PSD uses output words 001 and 010. The significant feature common to all structures with this spectrum is the introduction of two transitions within the input symbol interval by one of the output words, which explains the increase in the amount of high frequencies (compared to plot (7) for example). Similar reasoning suggests that combinations of output words, which introduce two transitions within the time interval T_d for both input symbols, should correspond to the spectrum with the highest level of high-frequency components. Indeed, this is exactly the case and an example of such a combination is the code with output words 010 and 101, which produces the spectral distribution of plot (9).

The assessment of codes generated for the $D2S1X8$ category is only suggestive and cannot serve as an exhaustive investigation of this group of structures. There are many more possibilities to select and group different coding schemes with respect to a much greater variety of features. More detailed interpretation of the analysis results will be entirely dependent on the design purposes or a particular application. The same considerations apply to the discussion of the other categories presented in this chapter. Only their most essential features are outlined with just a few sample structures selected to support a particular argument or to illustrate certain characteristics.

$\boxed{D4S1X4}$

The brief presentation of this category serves mainly to illustrate the change in the complexity level due to the third parameter – the number of input blocks of symbols, M. First it should be noted that in spite of the grater number of possible combinations, 256 compared for the previous two categories, the actual set of valid codes is relatively small. The reason for this is easily recognised to be the equal sizes of the input and output words, $M = K$ as shown in the 'Symbols' specification table below.

$D4S1X4$ - Symbols

input		output	
$d = [0, 1]$		$x = [0, 1]$	
$D_1 = 0\ 0$	$D_3 = 1\ 0$	$X_1 = 0\ 0$	$X_3 = 1\ 0$
$D_2 = 0\ 1$	$D_4 = 1\ 1$	$X_2 = 0\ 1$	$X_4 = 1\ 1$
$N = 2,\ \ M = 4$		$L = 2,\ \ K = 4$	
$s = [s_1]$			

Table 5.4

Clearly, the only variation in the process of transformation of input symbols into output blocks of symbols, without reducing the original information content, comes from the use of different permutations of the four output words. Hence, the number of practical codes this category yields is $K! = 24$. It is not difficult to see that with respect to the frequency distribution these combinations at most produce only a change in the places of the input symbols within a block of four without affecting the statistics of the input sequence. Therefore all of them exhibit the same spectral density, marked number (10) in Fig. 5.3.

This plot corresponds exactly to the PSD of an NRZ signal which is produced when $X_i = D_i$, for $i = 1,\ldots,4$. If the latter is assumed to represent the identity transformation, it is possible to think of the rest from the group of 24 valid schemes as some kind of inversions. At least some of them, like the one which produces the transformations $\{D_1 = 00\} \rightarrow \{X_1 = 11\}$; $01 \rightarrow 10$; $10 \rightarrow 01$; $11 \rightarrow 00$, can be recognised as fitting such a description straightaway.

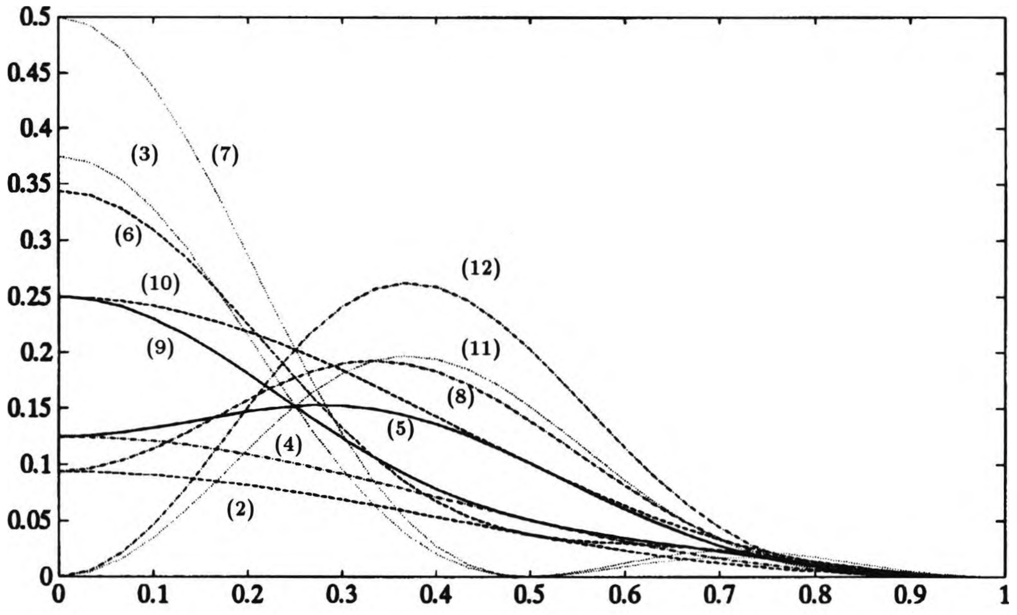


Fig. 5.3 The PSD plots for the $D4S1X4$ category

The rest of the combinations generated for the $D4S1X4$ category may not produce valid line codes, but still indicate possibilities which can become of interest in other applications. For example, a requirement may be imposed to eliminate certain combinations of two consecutive binary digits in a signal which has sufficient redundancy or can tolerate some ambiguity. Simulating and analysing all structures from this category also shows that there might be a great number of plots (10 in Fig. 5.3) which appear to be useful, but only a fraction of them correspond to valid schemes. This situation emphasises the need for careful specification of the simulation conditions which would ensure a selection of practical codes matching the design objectives.

The three basic categories presented in this subsection illustrate the most essential features of the enhanced analysis procedure based on the general coder model:

- Groups of codes can be generated and spectrally analysed according to a common set of specifications and selection parameters.
- A systematic comparative assessment can be performed over a range of coding structures arranged in categories with respect to appropriately specified hierarchy of complexity levels.

The simulation of all possible schemes deriving from the 'Symbol' specification tables of the three categories allows a number of intermediate conclusions to be reached:

- The conventional signalling formats, NRZ, RZ and Manchester, whose definition as codes is based on the suggested unified approach, have been encountered in the expected categories – $D2S1X2$, for NRZ and $D2S1X4$, for the others.

- The appearance of some structures in different categories is not a problem when an independent and exhaustive analysis is carried out within a single group. However, repetitive structures should receive special attention in order to eliminate ambiguities when different categories are investigated.

- Suitable conditions have to be specified to avoid (time consuming) analysis of combinations which do not represent meaningful spectral densities.

- It is possible to encounter PSD plots with shapes which seem close to realistic requirements, but do not correspond to valid structures (at least in the sense of line coding).

- A considerable number of coding schemes produce identical spectral density functions regardless of the difference in their state and symbol sets[7].

- Computation of identical PSD functions can be reduced significantly by recognising that all permutations of the output matrices, Z_m, within a given set ZM, with respect to their indices, represent effectively the same coding scheme when the probabilities of the input words are equal.

Consequently, the total number of combinations (with repetitions) of output matrices can be reduced from $(K^I)^M$ to

$$\binom{K^I + M - 1}{M} = \frac{(K^I + M - 1)!}{M!(K^I - 1)!} \tag{5.5}$$

Finally, it should be noted that the interrelations between the categories described above indicate the adequacy of the classification based on the suggested complexity levels. The lowest level is illustrated by the three categories with minimal number of input words and states ($M = 2, I = 1$), mentioned in this

[7] This conclusion is based on comparison of the continuous part of the PSD to avoid going into too much detail. Reviewing the results with the discrete PSD included poses no difficulties in principle.

subsection. Within this level the complexity increases, following the number of output words, from $D2S1X2$, with only four combinations, to $D2S1X8$ with 64 schemes and larger variety of spectral density plots. The complexity level discussed next, having assumed that $K > 8$ is of little practical interest, corresponds to an increased number of input words, $M = 4$. As codes with $K < 4$ are not likely to be very useful, the basic structure representing this level has been assumed $D4S1X4$. The larger number of coding schemes (256) indicates that the classification hierarchy is still applicable.

5.2.1.2 Coding Structures with Two States – *DMS2XK*

This section includes the categories which cover many of the remaining popular codes, as most of those which are in wide-spread use at present are 'two-state' schemes. AMI, CMI, Duobinary, 3B4B, etc. are only a few of the block codes which are commonly implemented in practical communication systems. The results from the analysis of these codes, given in Chapter 4, will be referred to in the following discussion, using the adopted abbreviations and terminology. New structures will be revealed by relating the known spectral densities to the most interesting of the plots, many of which correspond to unknown codes. Three categories, $D2S2X2$, $D2S2X3$ and $D2S2X4$ have been selected to represent the two-states group. The results, given in the following subsections, use only a small part of the large amount of data generated by the simulation procedure. Every effort has been made to select PSD plots which are typical for groups of codes and exhibit features of practical value for the purposes of line coding. The main restriction to the number of simulations has been imposed by implementing additional verification on the combinations of the output matrices. This leads to a reduction of the number of codes, given by (5.5) for every combination of state matrices by the number of combinations of M output matrices which contain less than M different output words. For the chosen three $DMS2XK$ categories the number of input words is $M = 2$ and expression (5.5), developed to accommodate the additional restriction, becomes

$$\binom{K^2+1}{2} - K = \frac{(K^2+1)!}{2!(K^2-1)!} - K = 0.5K(K^3+K-2)$$

In practice the analysis results for each category are stored in separate files for every valid pair of state matrices due to limitations in the available computer memory. Therefore the PSD selection strategy is discussed separately for the groups of plots presented below.

$\boxed{D2S2X2}$

This category, specified in Table 5.5, can be considered the smallest collection of meaningful two-state structures, where two input symbols are transformed into two output symbols.

$D2S2X2$ - Symbols

input	output
$d = [0, 1]$	$x = [0, 1]$
$D_1 = 0$ $D_2 = 1$	$X_1 = 0$ $X_2 = 1$
$N = 1,\ M = 2$	$L = 1,\ K = 2$
$s = [s_1, s_2]$	

Table 5.5

There are 256 combinations of state and output matrices, but the number of those which correspond to valid codes is much smaller. A further reduction in the number of simulation results comes from the fact that most structures produce identical spectra. There are only 6 different spectral density plots, which are shown in Fig. 5.4.

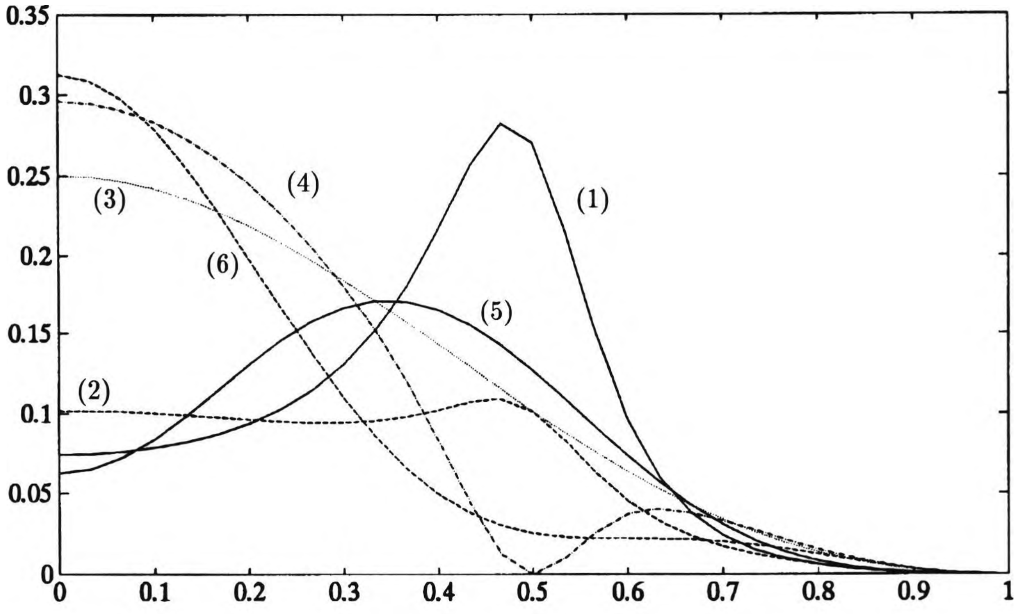


Fig. 5.4 A selection of PSD plots for the $D2S2X2$ category

The only PSD graph, recognisable as corresponding to an existing code, is number (3), which represents the familiar Differential scheme. This code is associated with plot (3), because it is a two-state technique, although it has been shown in Chapter 2 that the Differential is a version of NRZ, which provides for specific frequency characteristics only when the input symbol probabilities are not equal. As all simulations in this chapter are produced for equiprobable input symbols, PSD plot (3) is the same as that of unipolar NRZ.

Most of the interesting results are revealed by studying the table of equivalence. It is not practical to show this table in full, but sample groups of codes, corresponding to the graphs in Fig. 5.4, are discussed bellow to illustrate some useful interrelations. A selection of coding schemes from the $D2S2X2$ category, producing plot (3), are specified in Tables 5.6(a-c).

D_m	s_1	s_2	S_m	Z_m
0	0, s_2	0, s_2	$\begin{bmatrix}0&1\\0&1\\0&1\\1&0\end{bmatrix}$	$\begin{bmatrix}0\\0\\1\\1\end{bmatrix}$
1	1, s_2	1, s_1		

Table 5.6a A sample $D2S2X2$ code

D_m	s_1	s_2	S_m	Z_m
0	0, s_2	0, s_2	$\begin{bmatrix}0&1\\0&1\\1&0\\1&0\end{bmatrix}$	$\begin{bmatrix}0\\0\\1\\1\end{bmatrix}$
1	1, s_1	1, s_1		

Table 5.6b A sample $D2S2X2$ code

D_m	s_1	s_2	S_m	Z_m
0	0, s_2	1, s_2	$\begin{bmatrix}0&1\\0&1\\0&1\\1&0\end{bmatrix}$	$\begin{bmatrix}0\\1\\1\\0\end{bmatrix}$
1	1, s_2	0, s_1		

Table 5.6c A sample $D2S2X2$ code

D_m	s_1	s_2	S_m	Z_m
0	0, s_2	1, s_2	$\begin{bmatrix}0&1\\0&1\\1&0\\1&0\end{bmatrix}$	$\begin{bmatrix}0\\1\\1\\0\end{bmatrix}$
1	1, s_1	0, s_1		

Table 5.6d A sample $D2S2X2$ code

D_m	s_1	s_2	S_m	Z_m
0	0, s_2	1, s_2	$\begin{bmatrix}0&1\\0&1\\1&0\\1&0\end{bmatrix}$	$\begin{bmatrix}0\\1\\0\\1\end{bmatrix}$
1	0, s_1	1, s_1		

Table 5.6e A sample $D2S2X2$ code

D_m	s_1	s_2	S_m	Z_m
0	1, s_2	0, s_2	$\begin{bmatrix}0&1\\0&1\\1&0\\1&0\end{bmatrix}$	$\begin{bmatrix}1\\0\\1\\0\end{bmatrix}$
1	1, s_1	0, s_1		

Table 5.6f A sample $D2S2X2$ code

The common features, relating these codes to the Differential (Table 4.6b), can be identified by considering typical transformations of certain input symbol patterns. These transformations are described below for all of the Tables 5.6(a-f), but it should be emphasised that they are not required for the unique definition of the codes. The latter are completely specified by $D2S2X2$-Symbols (Table 5.5) and the respective 'Code' (Table 5.6a-f). The transformations for the first two structures, (a) and (b) are straightforward:

input 0-s are coded as output 0-s,

input 1-s are coded as output 1-s.

Obviously the effect from these two schemes is identical to that of the conventional NRZ. The only difference is the existence of an absorbing[8] coder

[8] The terms absorbing and cyclic states are borrowed from the theory describing probability matrices and Markov chains. No special definitions is considered necessary here.

state, s_2 for input 0-s and cyclic states, s_1 and s_2 for input 1-s in the case of (a), while in (b) there are two absorbing states s_1 and s_2 for input 0-s and input 1-s, respectively. The relations between the structures from Tables 5.6(c-f) can be illustrated by specifying the transformation of several typical patterns of input symbols, as shown in Table 5.7.

input-symbol	coded-symbol pattern											
pattern	consecutive 0-s				consecutive 1-s				alternating 0 and 1			
consecutive 0-s			x	x	x	x						
consecutive 1-s						x	x	x	x			
alternating 0 and 1	x	x									x	x*
code table (5.6) →	c	d	e	f	c	d	e	f	c	d	e	f

Table 5.7 Symbol transformation patterns
(* inverted)

The comparative assessment of code structures performed in Table 5.7 is not complete as it relates the transformations of only three symbol patterns. The example, however demonstrates the possibility to deduce important code features in a uniform manner. In this case Table 5.7 indicates that the four codes are versions of the Differential scheme as the respective transformations appear to be closely related to the general description given in Section 2.3.1.

The remaining spectral density graphs also lead to various interesting conclusions. Plot (4), for example, illustrates what can be achieved by using state-dependent decoding. One of the codes, corresponding to this spectrum, is specified in Table 5.8.

D_m	s_1	s_2	S_m	Z_m
0	1, s_2	0, s_2	$\begin{matrix}0 & 1\\ 0 & 1\end{matrix}$	$\begin{matrix}1\\ 0\end{matrix}$
1	1, s_2	1, s_1	$\begin{matrix}0 & 1\\ 1 & 0\end{matrix}$	$\begin{matrix}1\\ 1\end{matrix}$

Table 5.8 A sample $D2S2X2$ code

It shows that 0 in some combinations of input symbols is coded as 1, when the preceding input symbol has been a 1 and the present coder state – s_1. Clearly knowledge of the state sequence is needed for correct decoding. This imposes stringent synchronisation requirements and increases the susceptibility to errors. Such a price, however may be worth paying in a case where bandwidth efficiency is of highest priority.

Other basic code features can be determined from the schemes of this category in order to demonstrate their direct relation to the shape of the respective spectra. The two structures, given in Tables 5.9a,b, correspond to plots (1) and (2) in Fig. 5.4, respectively.

D_m	s_1	s_2	S_m	Z_m
0	0, s_2	0, s_2	$\begin{bmatrix}0 & 1\\0 & 1\\0 & 1\\1 & 0\end{bmatrix}$	$\begin{bmatrix}0\\0\\0\\1\end{bmatrix}$
1	0, s_2	1, s_1		

Table 5.9a A sample $D2S2X2$ code

D_m	s_1	s_2	S_m	Z_m
0	0, s_2	0, s_2	$\begin{bmatrix}0 & 1\\0 & 1\\0 & 1\\1 & 0\end{bmatrix}$	$\begin{bmatrix}0\\0\\1\\0\end{bmatrix}$
1	1, s_2	0, s_1		

Table 5.9b A sample $D2S2X2$ code

The codes can be compared using the technique applied through Table 5.7. Both schemes perform equivalently for consecutive input symbols, introducing transitions in the output signal for consecutive 1-s. This shows as a shift of the signal power from the low frequency range towards the higher frequencies. The difference between the two schemes becomes apparent for transformations of alternating 0-s and 1-s: the code from Table 5.9a preserves the transitions in the input signal, while that from Table 5.9b does not. The effect of this difference is reflected in the levels of high frequency components of the respective spectra.

Finally, by similar investigations of the code tables, corresponding to plots (5) and (6) it is possible to identify their specific advantages and disadvantages. The sample structures, producing PSD (5), have been found to introduce considerable ambiguity and the comments, made in the preceding section about invalid line codes from the $D4S1X4$ category, apply in this case, too. PSD (6), on the other hand, is produced by schemes which transform alternating input symbols into consecutive 0-s or 1-s, while preserving consecutive identical input symbols unchanged. Effectively, this results in a reduced number of transitions, which leads to an expected shift of the spectrum towards the lower frequencies, compared to plots (1) and (2). The latter are produced by codes, which generally introduce transitions. The comparison with PSD (4) is interesting, too. It shows that the narrowed bandwidth of PSD (6) is achieved at the expense of a slightly increased level of the higher frequencies.

The assessment of the coding schemes from the $D2S2X2$ category, all having the complexity of the conventional Differential scheme, allows for some generalisations to be made.

- The simulation and the analysis of all codes from one complexity group is possible, just as has been done for the $D2S2X2$ category.
- The spectral analysis results, together with the respective specification tables, show the adequacy and the straightforward applicability of the suggested classification method.

- The capability of the developed computational procedure to generate all possible structures for a particular category allows new codes to be created, some of which feature spectral density functions of practical value.

- The degree of modification of the of the signal spectrum is proportional to the amount of information redundancy which can be introduced within the limits of a particular complexity level.

The interpretation of the last conclusion with respect to the $D2S2X2$ category means that only simple modifications, like narrowing the bandwidth and partial suppression of the low frequency components, can be achieved. The conclusion is an important link with the preliminary considerations about the relation between structural and and spectral features, given in Section 4.1.2.3 of the previous chapter. This is yet another proof of the viability of the suggested general classification, based on the complexity level of codes. This idea is further developed in the next subsection of this chapter.

$\boxed{D2S2X3}$

The three sets D, s and X of this category are given in Table 5.10. The number of code structures, deriving from these sets is too large to allow a complete assessment of all 1296 combinations. All possible codes, however, have been generated and analysed. The results are given in Fig. 5.5a and indicate, as with previous categories that there is a large number of equivalent structures.

$D2S2X3$-Symbols

input	output
$d = [0, 1]$	$x = [-1, 0, +1]$
$D_1 = 0$ $D_2 = 1$	$X_1 = -1$ $X_2 = \ \ 0$ $X_3 = +1$
$N = 1,\ M = 2$	$L = 1,\ K = 3$
$s = [s_1,\ s_2]$	

Table5.10

This significantly reduces the number of different plots, but it is still a formidable task to investigate every spectrum from Fig. 5.5a and the respective sets of codes. The purpose of this presentation is not to identify or select particular types of codes or categories. It is intended to provide a broad basis for research and design of coding techniques by demonstrating the uniform assessment approach and the advantages of combining the powerful analysis procedure with the general method of classification of code structures.

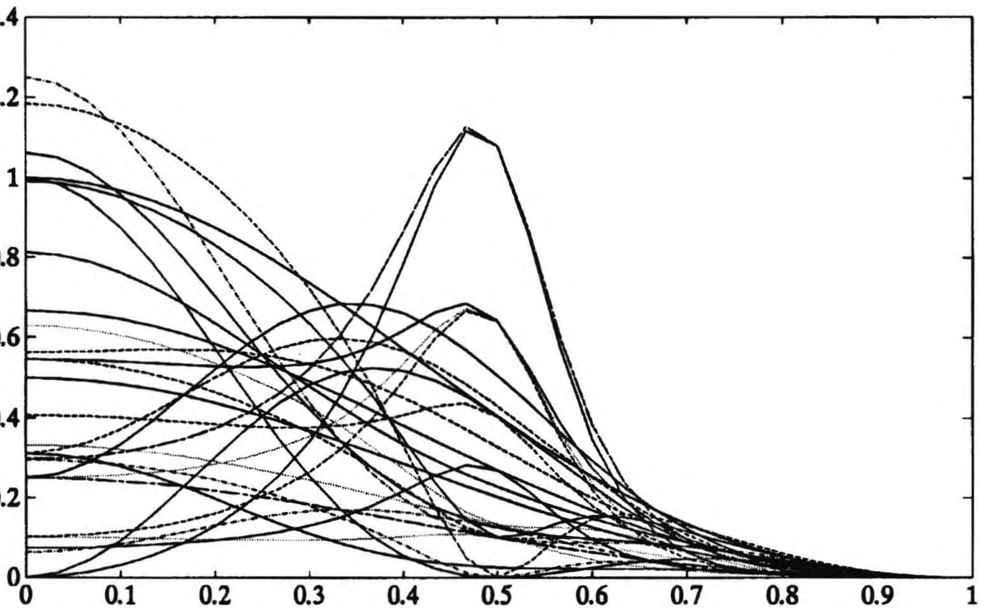


Fig. 5.5a The PSD plots for the $D2S2X3$ category

Some typical features of the codes from this category are discussed below with reference to a selection of spectral density plots shown in Fig. 5.5b.

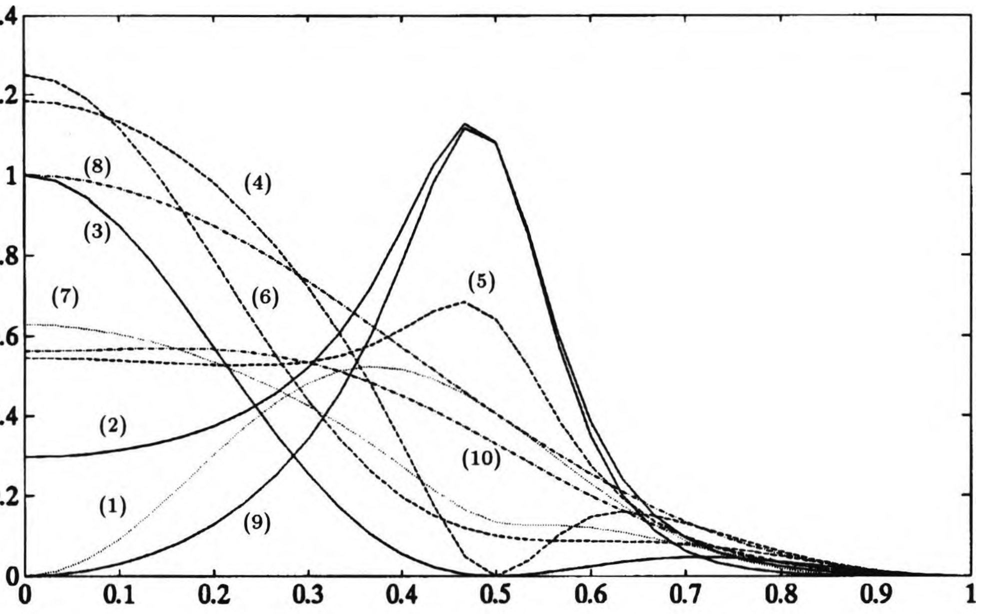


Fig. 5.5b A selection of PSD plots for the $D2S2X3$ category

The coding schemes corresponding to these plots are of the same complexity as the conventional AMI and Duobinary codes (specified in Tables 4.7 and 4.8, respectively), which also belong to this category. The spectral density functions of the two popular techniques can be recognised as graphs (1) and (3), which represent the most essential spectral features achievable in this category:

ı) suppression of the low frequency components,

ıı) reduction of the bandwidth by half.

These results can be viewed as directly related to the possibility to introduce higher amount of redundancy than in the previous category, as the *D2S2X3* codes use three output symbols to transform the two input symbols.

A brief overview of the PSD plots in Fig. 5.5b shows a variety of shapes representing different degree of spectrum modification. The basic type of PSD, number (8) predictably represents the NRZ equivalent transformations. A sample of the latter, which converts input zero into −1 and input one into +1, is specified in Table 5.11.

This code is another example of overlap between categories. As only two of the three output symbols are used, the structure from Table 5.11 can be considered belonging to the *D2S2X2* category.

D_m	s_1	s_2	S_m	Z_m
0	$-1,\ s_2$	$-1,\ s_2$	$\begin{matrix}0 & 1\\ 0 & 1\end{matrix}$	$\begin{matrix}-1\\ -1\end{matrix}$
1	$+1,\ s_2$	$+1,\ s_1$	$\begin{matrix}0 & 1\\ 1 & 0\end{matrix}$	$\begin{matrix}+1\\ +1\end{matrix}$

Table 5.11 A sample *D2S2X2/3* code

It has been mentioned in previous sections that special restrictions can be imposed through the process of construction of the computational matrices SM and ZM in order to avoid codes from lower categories recurring at higher complexity levels. In this case, however, the presence of the NRZ plot is used to illustrate the stages of modification leading from the basic spectrum, (8) to those of AMI and Duobinary. The modifications of the spectral density graphs for this category are considered only with respect to the essential features ı) and ıı) specified above. Thus plots (2) and (6) can be viewed as intermediate stages towards achieving reduced low frequency components and bandwidth, respectively. Examination of two code tables, corresponding to these plots, reveals that the PSD modifications have been achieved through structures from a lower category (*D2S2X2* again) at the expense of introducing certain ambiguity in the coded sequence.

The considerations given above show the importance of careful selection of the codes producing the required modification of the spectral density function. Good examples of valid codes with some degree of PSD modification capability are given in Tables 5.12a,b.

D_m	s_1	s_2	S_m	Z_m
0	$-1,\ s_2$	$-1,\ s_2$	$\begin{matrix}0&1\\0&1\end{matrix}$	$\begin{matrix}-1\\-1\end{matrix}$
1	$0,\ s_2$	$+1,\ s_1$	$\begin{matrix}0&1\\1&0\end{matrix}$	$\begin{matrix}0\\+1\end{matrix}$

Table 5.12a A sample $D2S2X3$ code

D_m	s_1	s_2	S_m	Z_m
0	$-1,\ s_2$	$-1,\ s_2$	$\begin{matrix}0&1\\0&1\end{matrix}$	$\begin{matrix}-1\\-1\end{matrix}$
1	$+1,\ s_2$	$0,\ s_1$	$\begin{matrix}0&1\\1&0\end{matrix}$	$\begin{matrix}+1\\0\end{matrix}$

Table 5.12b A sample $D2S2X3$ code

Plots (5) and (7) in Fig. 5.5b correspond to these two codes, showing a certain degree of low frequency suppression and bandwidth reduction respectively. At this point it can be suggested that valid codes from this category can probably achieve the expected reshaping of the basic spectrum (8), but only at the expense of some relative reduction of the overall power within the bandwidth of practical interest.

The PSD graphs (1) and (3) in Fig. 5.5b indicate the validity of the above consideration. They represent the maximum degree of spectrum modification for this category with an overall power reduction of approximately a half, compared to the basic NRZ type of spectrum. As already mentioned, the two plots are the spectral densities of the AMI and the Duobinary codes, but it should be noted that some invalid code schemes also produce the same PSD. This underlines the importance of careful specification of the restrictions in selecting combinations of state and output matrices in the process of simulation.

The investigation of the possibilities of spectrum shaping, provided by the codes from the $D2S2X3$ category, can be completed by considering graphs (4) and (9) in Fig. 5.5b. They seem to fulfil the requirements for narrowing the bandwidth and suppressing the low frequencies, without significant power reduction. It turns out, however, that the code structures corresponding to these plots are not valid combinations of state and output matrices.

The overall assessment of the simulation and the analysis results for this category confirm the limitations of the spectrum shaping capabilities, discussed in Section 4.1.2.3. In summary:

- Codes from category $D2S2X3$ introduce sufficient redundancy to achieve spectral features i) and ii), specified at the beginning of this subsection.

- The amount of redundancy, without introducing state-dependent decoding, is minimal, which results in only a few valid codes producing significant low frequency suppression or bandwidth reduction.

- As expected, the redundancy introduced by the $D2S2X3$ structures is insufficient to achieve spectrum modification which combines features i) and ii) in one PSD.

The additional redundancy, needed to produce a coded signal spectrum with both, small low frequency components and halved bandwidth, is provided through coding schemes of higher complexity.

$\boxed{D2S2X4}$

This is the highest complexity category, using two states and two input words, which contains a popular code. In spite of the apparent variety of combinations, illustrated by the family of PSD plots in Fig. 5.6a, CMI is the only scheme from this category which is widely used in practical communication systems.

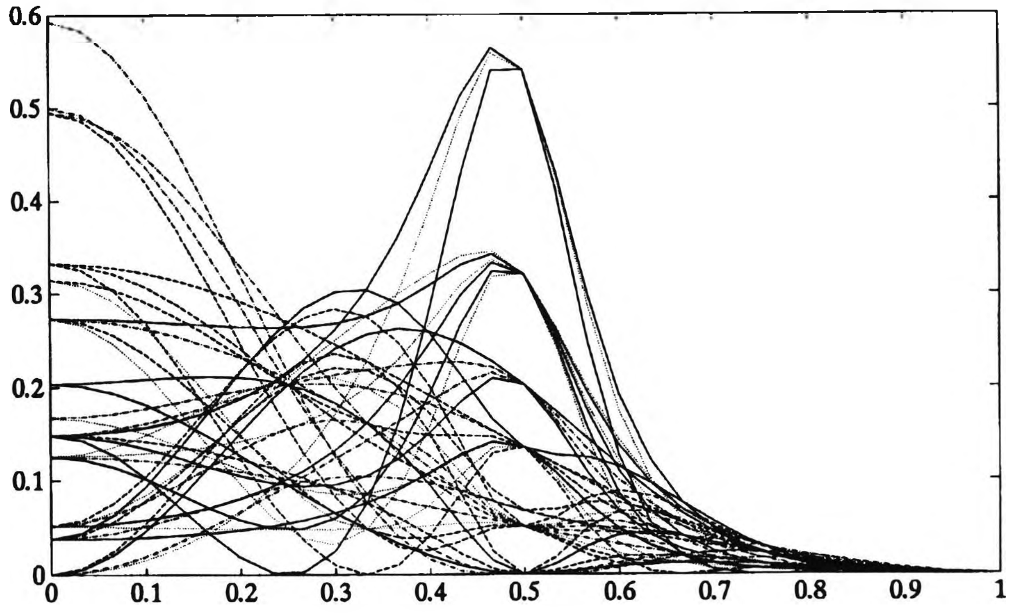


Fig. 5.6a A pattern-illustrating selection of PSD-s for the $D2S2X4$ category

The group of plots in Fig. 5.6a represents approximately a tenth of the complete selection of structures which have been simulated for the $D2S2X4$ category. Detailed assessment of individual structures from this group of codes is considered unnecessary and the plots shown in Fig. 5.6b have been selected only to illustrate some typical spectral features, achievable by using a two-state coder and two output symbols for every input symbol.

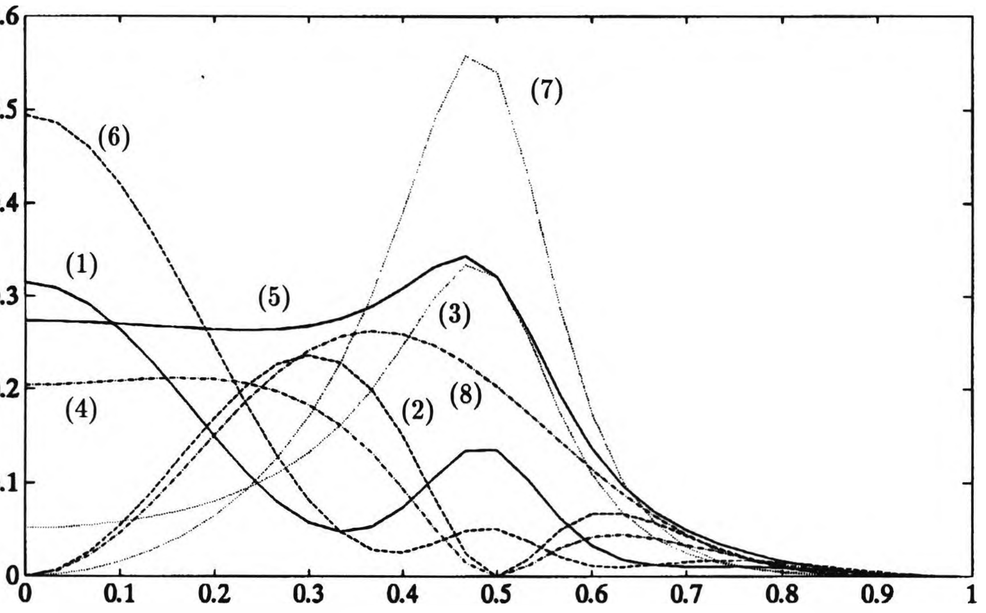


Fig. 5.6b A selection of typical PSD plots for the $D2S2X4$ category

First, it should be noted that only PSD graphs (1), (3-6) and (8) in Fig. 5.6b represent codes, which allow for unique decodability. Two of these spectra, (4) and (6) feature significant reduction of the bandwidth relative to the normalised NRZ spectrum. In this case a reduction of the output signal bandwidth by half only makes it the same as that of the input sequence, as the code symbol repetition rate, F_x is twice that of the input symbols, i.e. $F_x = 2F_d$. At the same time, it appears that the additional redundancy of the $D2S2X4$ structures is still insufficient to achieve both suppressed low frequencies and halved bandwidth without losing the feature of unique decodability. PSD plots (2) and (7) in Fig. 5.6b show the possibilities to produce the required spectrum modification, if state-dependent decoding is allowed for.

The new spectral feature, which is suggested by some codes from this category, is shown in graph (1). Although this spectrum contains large low frequency components, its main advantage is the possibility to reduce the bandwidth to a third of the output symbol rate. This is indicated by the normalised frequency of the first minimum of the spectral density function, which is approximately 0.33. Considering that, with respect to the input symbol rate the width of the resulting spectrum is still about $0.66F_d$, the saving is not too big, but certainly worth noting, as it is made with a uniquely decodable scheme. The result corresponds to the expected effect from having introduced more complexity, therefore redundancy, with the $D2S2X4$ category.

Finally, the two plots given in Fig. 5.6c indicate the limitations of the spectrum shaping capabilities of the codes from this complexity level. The PSD graph (1) is the familiar CMI spectrum, which is a classic example of a compromise in spectrum shaping. The two-level coded signal is considered one with well suppressed low frequency contents and reduced bandwidth, although the amount of high frequency components, above half the code symbol repetition rate, is not negligible.

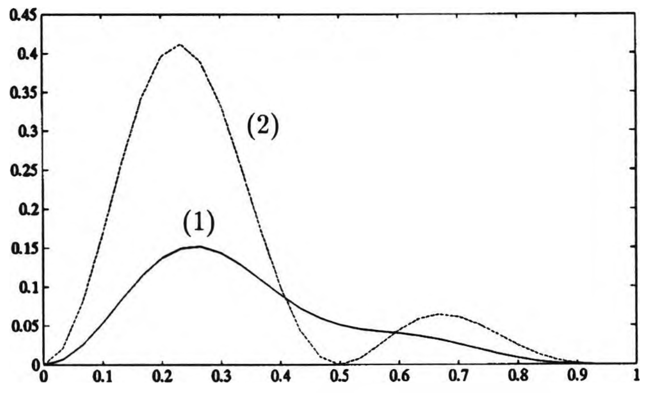


Fig. 5.6c The limiting spectrum modification ($D2S2X4$)

It is not unreasonable to expect the possibility of a code from this category to offer a modified spectrum which completely satisfies both requirements ı) and ıı), and has the generally preferred 'bell' shape within the $0 - 0.5F_x$ range. Indeed, the code specified in Table 5.13 produces exactly such a PSD, shown as (2) in Fig. 5.6c. This code seems to be a more successful version of CMI, but it has one disadvantage.

Without introducing state-dependent decoding, the improved shape of the spectrum (compared to that of CMI) is achieved by increasing the susceptibility to errors due to the use of all four output words.

D_m	s_1	s_2	S_m	Z_m
0	01, s_1	10, s_2	$\begin{bmatrix}1 & 0\\0 & 1\end{bmatrix}$	$\begin{bmatrix}01\\10\end{bmatrix}$
1	11, s_2	00, s_1	$\begin{bmatrix}0 & 1\\1 & 0\end{bmatrix}$	$\begin{bmatrix}11\\00\end{bmatrix}$

Table 5.13 The CMI related sample code

This is unlike CMI, where combination 10 is not used as a code word and allows for some level of error control. Despite of its high sensitivity to loss of block synchronisation, the code from Table 5.13 may be a suitable choice for certain bandwidth efficient applications.

5.2.1.3 Categories of Higher ($I > 3$) Complexity Levels

The two categories $D2S4X3$ and $D2S4X4$ have also been investigated, as they contain the Modified Duobinary and the Miller codes, which have been discussed in Chapter 4. The simulation data has shown the availability of a great variety of structures with considerable spectrum shaping potential. The presentation of coding schemes and the respective analysis results for these two categories requires more involved discussion of sophisticated 'Code' tables without contributing significantly towards the general description of the code generating algorithm and the uniform assessment procedure. It is sufficient to mention the availability of many coding schemes, which provide for significant reduction of the coded signal bandwidth, in some cases making it narrower than the bandwidth of the input sequence. Although the number of generated code structures, which preserve the unique decodability feature, is only a small fraction of all valid combinations, there is still a large variety of spectra with shapes of practical interest.

The investigation of coding structures, generated with the simulation routine can be extended almost indefinitely just by increasing the number of coder states. For categories with $I > 3$ and $M, K \geq 2$, the total number of possible structures increases very rapidly. Well devised restrictions and conditions of selection have to be implemented to avoid time consuming analysis of impractical and unwanted structures. Although a total of several thousand codes have been simulated and all essential data has been stored for further research, it is considered unnecessary to include more examples in this presentation. The results shown in the previous sections are sufficient to illustrate the fundamental concepts of the suggested uniform assessment and analysis approach. Most of the important stages in the generation and classification of new code structures have been demonstrated through the systematic investigation of some low-complexity level categories. Many of the popular coding techniques have been shown to appear precisely in the predicted categories, exhibiting the expected structural and spectral features typical of the respective group of schemes. These results prove the general applicability of the suggested classification method.

The investigation of complete categories is another significant achievement. All codes from $D2S1X4$, $D2S1X8$, $D4S1X4$ and $D2S2X2$ have been generated and analysed. This in effect precludes the possibility to 'invent' a new coding scheme with parameters specified within these categories. Some higher complexity level categories have also been exhaustively simulated and assessed, but their complete investigation will be presented in a future extension of this work.

6. THE INFORMATION CAPACITY OF CODES

The capability of different codes to shape the spectrum of a sequence of symbols can be assessed with respect to the amount of redundancy they introduce to achieve particular spectral features. A new method for evaluation of the efficiency of coding structures is proposed in this chapter. The method derives directly from the uniform approach in the assessment and analysis of codes and suggests the possibility to introduce an overall measure of the spectrum-shaping potential of a code for a particular symbol transformation efficiency. This parameter will be further referred to as the Code Information Capacity (CIC). The method is based on the system of general classification of codes, developed in Chapter 5 and makes use of the relations between coding schemes from different categories.

Conversely, if complexity represents the structural features of codes, it is possible to introduce another overall parameter, which reflects their efficiency with respect to the optimal use of a particular combination of the three sets[1] D, s and X for certain modification of the coded signal spectrum. Such a parameter can be viewed as specifying the information 'carrying' (or information 'transfer') capacity of a code. By evaluating the CIC within the same category it is possible to determine the optimum relation between the code structure, the spectral characteristics it can produce and its efficiency. It has been demonstrated that a fixed length block code can be uniquely represented by the specification of its three sets (D, s, X) and their interrelation through a particular 'Code' table. Any coding scheme, complying with the conditions discussed in Chapter 3, can be modelled as an FSSM and the spectral characteristics of the output symbol sequence evaluated accordingly.

6.1 The Code Assessment Parameters

The general considerations given above identify the three main factors, which can be used to determine the practical value of a coding structure in comparison to other codes. These factors can be specified as follows:

- the operational space of a coder,
- the spectral characteristics of the coded sequence,
- the information capacity of a code.

A detailed discussion of these factors is presented in the following subsections.

[1] As defined in Chapter 5.

6.1.1 The Operational Space of Coders

This term is introduced to indicate the limitations imposed to the variety of possible code schemes by the finite sizes of the sets of symbols and states. The Coder Operational Space (COS) is defined as a 3-dimensional[2] volume, whose vertices are determined by the integers M, I and K, previously defined as the sizes of the three sets D, s and X, respectively. To simplify the illustration, it is convenient to view the indices $m = 1,..,M$, $i = 1,..,I$ and $k = 1,..,K$ of $D_m \in D$, $s_i \in s$ and $X_k \in X$ as the three coordinates (m, i, k) of points from the 3D space, determined by the system of three orthogonal axes, as shown in Fig. 6.1.

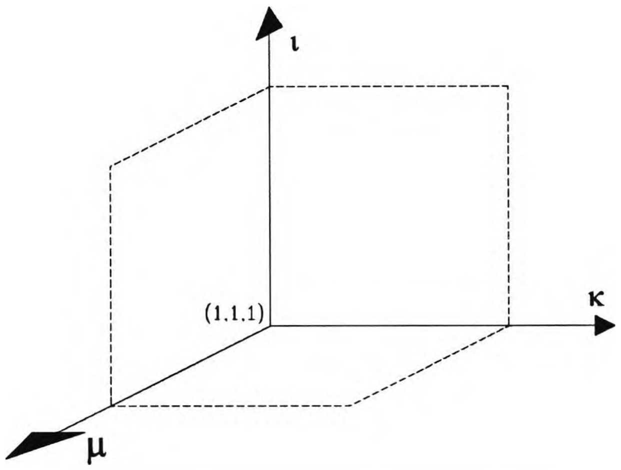


Fig. 6.1 3D representation of the coder sets

The intersection point of the axes μ, ι and κ has coordinates (1,1,1) and is assumed to be the origin of the system. The values of m, i and k are integers taken along the axes μ, ι and κ, respectively. Following the above assumptions it is possible to specify the COS as a 3D rectangular figure, determined by the coordinates of its vertices as shown in Fig. 6.2. The specification of the coordinates of the COS as combinations of the sizes of the three coder states determines the operational space for a group of codes, which belong to the same category, $DMSIXK$. This suggests the possibility to represent geometrically any coding scheme with respect to the COS of its category. Indeed, the correspondence between a geometrical shape and a code can be established as every 'Code'-table specification is completely determined by the unique combination of elements D_m, s_i and X_k with respect to their indices.

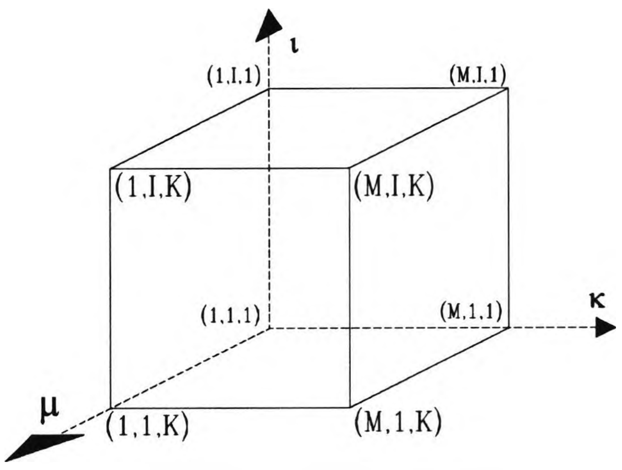


Fig. 6.2 The Coder Operational Space

[2] The abbreviation 3D will be used elsewhere in the presentation.

6.1.2 Spectral Characteristics of the Coded Signal

The second factor affecting the evaluation of a particular coding scheme is the spectral density distribution associated with the output symbol sequence. It has already been mentioned in Chapter 5 that no precise analytical definition of the suitability of a PSD function for particular applications is currently available. This is why quantitative estimation of spectral characteristic is not used in the subsequent discussion. This deficiency of the existing frequency analysis techniques, however, does not affect the applicability of the suggested evaluation method. The quantitative measure of the CIC can be used for comparative assessment of codes by assuming a set of descriptive criteria for an overall estimation of the spectral features of the coded sequences.

At this point it should be mentioned that in principle there are no obstacles in the development of a scheme for quantitative evaluation of the frequency characteristics of symbol sequences except, perhaps, the lack of a widely adopted uniform approach. This is why the results presented in this thesis can be regarded as an attempt to create a universal method for the design and analysis of most types of block codes. The possibility to describe analytically useful features of a PSD function has been discussed in the preceding chapters. Without going into any further detail the concept adopted for evaluation of frequency characteristics can be summarised as follows:

i) The bandwidth is determined as $f_{max} - f_{min}$, where f_{max} and f_{min} specify the frequency range containing a certain percentage of the signal power.

ii) The shift of the spectrum can be evaluated by the change of the position of the frequency range, $[f_{max}, f_{min}]$, used to determine the bandwidth of the output sequence as in i), relative to that of the input sequence.

Assuming, for the simplicity of the illustration, that there is no change in the width of the frequency range, the shift of the spectrum can be defined as the difference between the centre frequencies, $\frac{f_{max} - f_{min}}{2}$ of the frequency ranges of the input and the output sequences.

iii) Similarly the suppression of high- and low-frequency components can be determined by specifying the required transmission bandwidth and evaluating the difference between the percentages of the signal power contained within the respective frequency range before and after coding.

Alternatively, it is possible to evaluate the difference between the bandwidths of the original and the coded sequences, containing a given percentage of the signal power combined with the difference between the mid-points, $\frac{f_{max} - f_{min}}{2}$ of the respective frequency ranges.

iv) Finally, the quantitative evaluation of the spectral characteristics of the coded sequences can be completed with analogous measures involving the discrete frequency components of the spectrum.

The main purpose of this chapter is to give an overall description of the suggested evaluation method. The provision for precise analytical criteria for estimation of the spectral features of coded sequences is a vast research topic in itself. Therefore, to avoid unnecessary complexity, it is considered sufficiently accurate to present the method for evaluation of the CIC on a comparative basis, using typical examples of coding schemes and the respective description of the suitability of their frequency characteristics for practical applications.

6.1.3 Information Capacity of Codes

The third factor affecting the overall evaluation of a coding structure is associated with the ability of the codes to modify the spectrum or any other characteristic of the original sequence by introducing minimum redundancy. At present a quantitative measure of such an ability is given by the efficiency of codes. This measure, however takes into account only the relative change in the number of symbols per unit information. The insufficiency of the information conveyed by a number which shows solely the code's efficiency, can be illustrated by comparing the 3B4B and the 3B2T-RBS codes, analysed in Chapter 4. Both schemes have the same efficiency, 0.75, but the PSD of the latter has been shown to have certain disadvantages with respect to its use for the purposes of line coding.

The suggested method for evaluation of the CIC allows an integral parameter to be derived representing the combined effect of several essential characteristics of a coding structure:

- the classification category;
- the efficiency with respect to the sizes of the input and the output sets;
- the specific relation between the input/output sets and the set of coder states determined by the particular 'Code'-table.

Being a measure of capacity, the parameter described above is most adequately represented by a volume. Indeed, it is possible to associate the volume of a unique geometrical figure with any code within its operational space. In the general case this figure is determined by its vertices, which are all points with coordinates (m, i, k), corresponding to all combinations

$$(D_m, s_i, X_k) = (D_m, s_{mi}, X_{mi}) \tag{6.1}$$

from the 'Code' specification table as defined in Fig. 4.2, Chapter 4. Here s_i is the next coder state given by s_{mi}, which the coder goes into and X_k is the output word given by X_{mk} for the respective combination of the input word, D_m and the present state, s_i $(m = 1,\ldots, M;\ i = 1,\ldots, I;\ k = 1,\ldots, K)$. The position of a point with coordinates corresponding to the combination (D_2, s_2, X_2), in the COS of category $D3S2X3$ is shown as an illustration in Fig. 6.3.

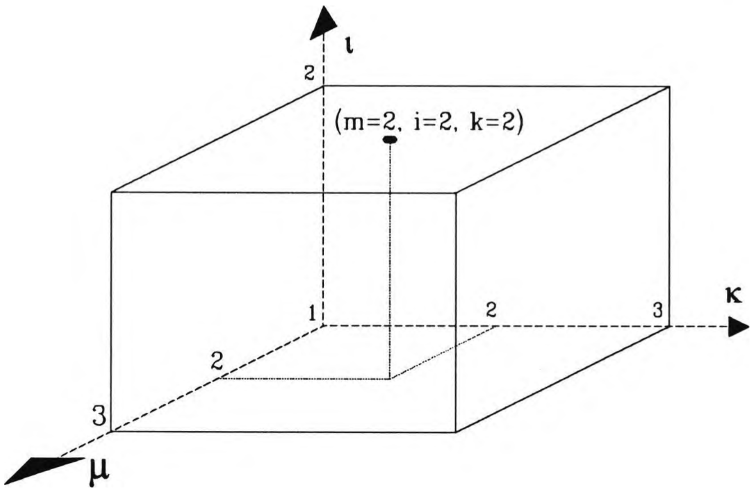


Fig. 6.3 Coordinates of a point determined from the coder set

All geometrical points and figures in the subsequent discussion correspond to particular coder specifications, therefore coordinates, given as (m, i, k), will always represent a (D_m, s_i, X_k) combination, without explicitly stating this relation. The correspondence between the geometrical interpretation of the CIC and the respective 'Code' table, for every example given below, is determined from the set of coordinates, (m, i, k) of the figure and their values.

6.2 Code Structures, Compared Through Their CIC

The presentation of the suggested method for evaluation of the CIC is based mainly on empirical results to avoid excessive involvement of complex analytical expressions. This is why some preliminary comment is required regarding certain extreme cases. The COS of categories from the lowest complexity level (e.g. $DMS1XK$) is two-dimensional[3]. All points with coordinates $(m,1,k)$ belong to the (μ,κ) plane, Fig. 6.1. A straightforward interpretation of the planar figures, corresponding to practical codes has not been suggested, due to the lack of an adequate theoretical basis. Additionally, some difficulties may be encountered in deciding which connections of points, representing higher complexity codes, have to be used to form the appropriate 3D volume. Although a definite rule is not yet available, some preliminary research results indicate that the geometrical shape, corresponding to the CIC measure, is the one with the largest possible volume for a given set of points. It may seem a formidable task to determine this volume when codes from very high order categories are considered, but the problem is purely geometrical and certainly manageable with the aid of modern computing resources.

The main purpose of this chapter is to demonstrate the possibilities to achieve consistent and informative results by applying the CIC evaluation method to a number of codes whose performance and characteristics are of practical interest. It is important to note that volume V of any figure, which represents a code from a particular category $DMSIXK$, has a finite value within the range $0 \leq V \leq V_{\max}$, where

$$V_{\max} = (M-1)\times(I-1)\times(K-1) \tag{6.2}$$

is the volume of the respective COS. This limitation is one of the arguments suggesting the possibility to find an optimum volume. Further considerations in this respect are given in the following subsections through a number of examples, used to illustrate the geometrical interpretation of the CIC.

6.2.1 The CIC of $D2S2X2$ Structures

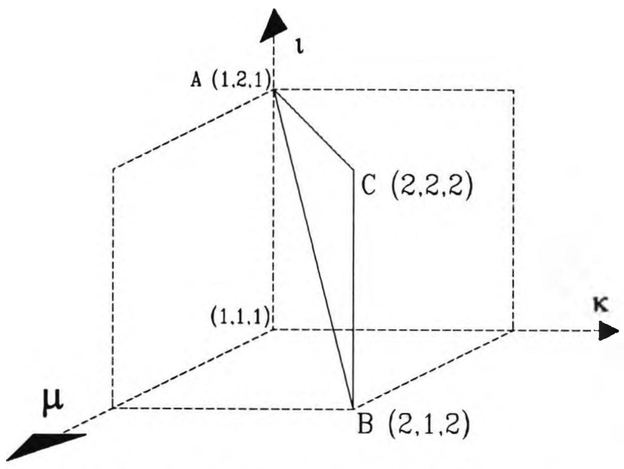


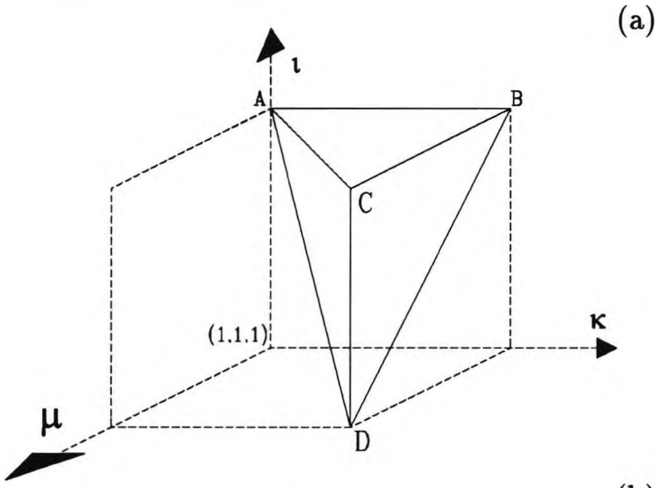
Fig. 6.4 The figure of a sample $D2S2X2$ code

The $D2S2X2$ category contains the lowest complexity level codes whose operational space has the minimum non-zero volume, $V_{\max} = 1$, according to (6.2). Without introducing any particular order, the structures specified by Tables 5.6a,b are the first to be represented geometrically in Fig. 6.4.

[3] The notation 2D will be used for short further in the presentation.

The coordinates derived from the two tables are [(1,2,1), (1,2,1), (2,2,2), (2,1,2)] and [(1,2,1), (1,2,1), (2,1,2), (2,1,2)], respectively. The connection of points A, B and C determines the figure for Table 5.6a, while points A and B represent Table 5.6b. Both tables specify codes which are equivalent to the conventional NRZ format. This can be related to the geometrical results by assuming that the zero volumes correspond to the 'no-coding' performance of the two structures, i.e. the codes do not produce symbol transformations and do not cause any modification of the spectrum. The comparison of the structures from Tables 5.6(c-f) is based on their geometrical representation, shown in Fig. 6.5a,b.

The coordinates of points A, B, C and D in Fig. 6.5a correspond to Table 5.6c and produce a tetrahedron with volume $\frac{1}{6}$ (having assumed the sides of the COS to be of unity length). The other three structures, Tables 5.6(d-f) are represented by the same rectangle, ABCD in Fig. 6.5b. The only difference is the order of the points regarding the place of the combinations determined from (6.1) in the respective tables. The shape of the conventional Differential scheme, defined in Chapter 4 is also given in Fig. 6.5b. It is determined by points B, F, D and E, forming a rectangle identical to ABCD.



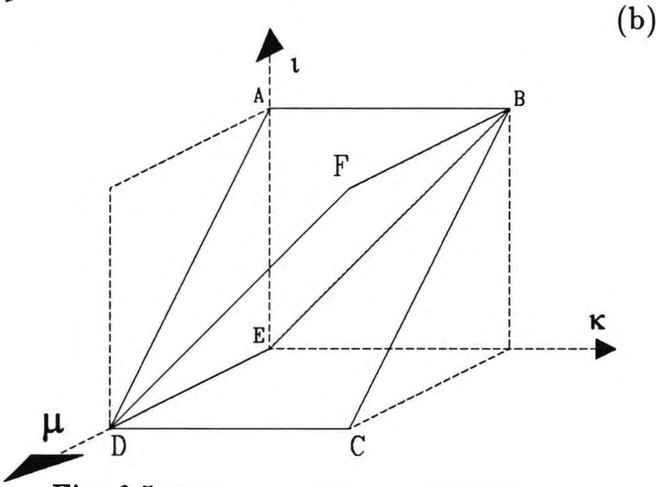


Fig. 6.5 Geometrical shapes of $D2S2X2$ codes

The last five structures have a common PSD, which is the familiar NRZ spectrum. The geometrical results from Fig. 6.5 can be interpreted as follows:

- The codes from Tables 5.6(d-f) and the Differential introduce redundancy through state-dependent decoding. This is assumed to correspond to the larger area of the figures compared to those from Fig. 6.4. The redundancy, however, is insufficient to achieve spectrum modification other than that of the Differential, shown in Fig. 4.7b, for unequal input symbol probabilities.

- The structure, which gives the 3D volume in Fig. 6.5a, differs from the other four in the amount of redundancy introduced, hence the non-zero volume. By careful investigation of the code's performance, however, it can be shown that the increased redundancy has contributed only towards achieving some more complex transformations, at the expense of certain ambiguity being possible for particular input sequences.

Obviously the coding ability, measured as CIC volume of $\frac{1}{6}$, which is associated with the higher redundancy, is still insufficient to produce any spectrum modification for equiprobable input symbols.

Similar reasoning can be applied to those codes from the *D2S2X2* category which can affect the spectrum, as discussed in Section 5.2.1.2 and shown in Fig. 5.4. It turns out that all of the examined structures, whose spectrum is different from the conventional NRZ, produce geometrical figures with zero volume. The results can be explained by assuming that such structures do not introduce additional redundancy and the spectrum modification effect is achieved entirely at the expense of increasing the ambiguity in the process of symbol transformation. The various areas of the 2D figures corresponding to these codes can be interpreted as being proportional to the degree of ambiguity introduced for particular changes in the respective PSD.

It is difficult to show the shapes representing all investigated structures, but the coding schemes discussed next, provide sufficient basis to answer the question: 'What is the optimum CIC volume and which are the codes with such an information capacity?'. The number of points, which determine the shape of a coder, from the *D2S2X2* category is four. (In the general case, this number is equal to $M \times I$.) A figure with a non-zero volume exists only when the four points do not belong to the same plane.

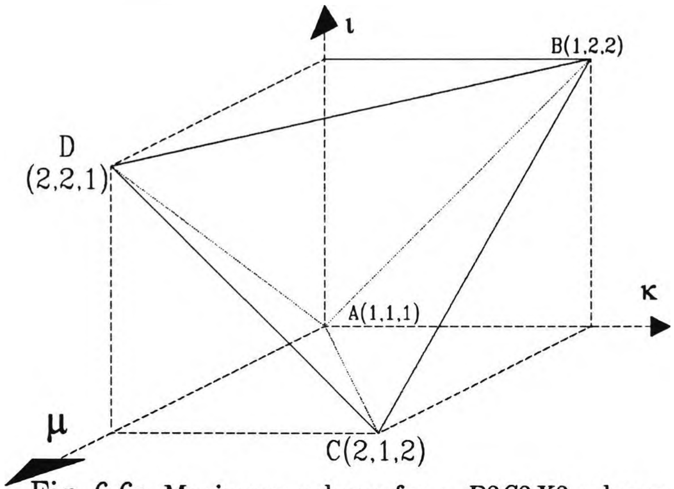


Fig. 6.6a Maximum volume for a *D2S2X2* scheme

For this category the 3D shape with non-zero volume, representing a code, is a pyramid, whose vertices are four of the of the 8 points, which determine the COS. It can be proven that the maximum volume pyramid is a tetrahedron with all sides being

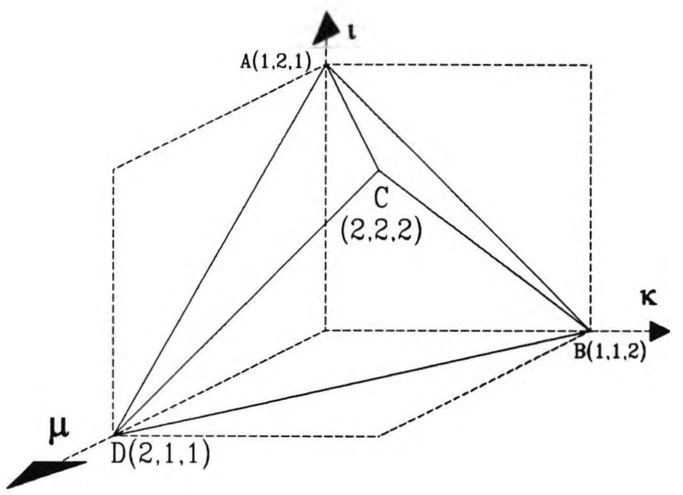


Fig. 6.6b Maximum volume for a $D2S2X2$ scheme

diagonals of the squares limiting the COS. In this case there are two possible configurations, which are shown in Fig. 6.6a,b. The two sets of coordinates, which correspond to the vertices of the two tetrahedrons, can be identified as codes, whose definitions are given in Tables 6.1a,b.

D_m	s_1	s_2	S_m	Z_m
0	0, s_1	1, s_2	$\begin{bmatrix}1 & 0\\0 & 1\end{bmatrix}$	$\begin{bmatrix}0\\1\end{bmatrix}$
1	0, s_2	1, s_1	$\begin{bmatrix}0 & 1\\1 & 0\end{bmatrix}$	$\begin{bmatrix}0\\1\end{bmatrix}$

(a)

D_m	s_1	s_2	S_m	Z_m
0	0, s_2	1, s_1	$\begin{bmatrix}0 & 1\\1 & 0\end{bmatrix}$	$\begin{bmatrix}0\\1\end{bmatrix}$
1	0, s_1	1, s_2	$\begin{bmatrix}1 & 0\\0 & 1\end{bmatrix}$	$\begin{bmatrix}0\\1\end{bmatrix}$

(b)

Table 6.1 Maximum-volume $D2S2X2$ structures

The codes from these two tables are easily found to produce the PSD of the conventional NRZ, i.e. the same as that of the Differential. Simple calculations show the volumes of the tetrahedrons in Fig. 6.6 are $\frac{1}{3}$. Assuming this to be the optimum CIC measure for the $D2S2X2$ category, the results confirm the conclusions reached in Chapters 4 and 5 about codes from this complexity level. In other words, the maximum redundancy, which can be introduced through structures from this category, is not sufficient to modify the spectrum of the input sequence, with equiprobable symbols, without also introducing ambiguity in the coded signal. The use of state-dependent decoding, however, allows some modifications of the PSD to be achieved for unequal input symbol probabilities.

There are more than two codes, whose 3D shapes are those from Fig. 6.6. This can be seen by considering that the two pairs of combinations $\{(D_m,s_1,X_2)$, $D_m,s_2,X_1)\}$ and $\{(D_m,s_1,X_1)$, $D_m,s_2,X_2)\}$, corresponding to the vertices of the tetrahedrons in Fig. 6.6a,b, can be positioned in several different ways in columns s_1 and s_2 in the respective 'Code' tables. Not all versions give valid coding schemes and the four which do have been found equivalent to those from Tables 6.1a,b.

In summary, the evaluation of the CIC for the *D2S2X2* category shows that the 3D volume is an adequate integral measure of the coding capabilities of different structures. The use of this measure in the comparative assessment of codes from this complexity group allows the relation between the information redundancy introduced by a code and its ability to modify the spectrum to be revealed. Finally, it has been demonstrated how the evaluation of the CIC makes it possible to determine optimal coding schemes. For this complexity level, four codes have been identified as optimal according to the volume of their 3D shapes, given in Fig. 6.6. As their performance is equivalent to that of the Differential, it can be expected that the larger CIC of the schemes specified in Tables 6.1a,b indicates a certain degree of superiority, which could be utilised in the form of a simpler state-dependent decoding, for example.

6.2.2 Evaluation of the CIC of Codes From the *D2S2X3* Category

The number of coding schemes at this complexity level is much larger than that of the previous category. This is why it is impossible even to summarise a complete investigation of all structures within the limits of this presentation. The selection of codes from the *D2S2X3* category, described in Chapter 5, is used in this section to illustrate the CIC measure applied to codes from a COS of size $V_{\max} = 1 \times 1 \times 2 = 2$.

The usual code to be considered first is AMI. Its 3D shape, with coordinates derived from the (D_m, s_{mi}, X_{mi}) combinations in Table 4.7, is shown in Fig. 6.7. Most predictably, this popular scheme, which achieves the low-frequency suppression, while preserving the unique decodability feature, is represented by a 3D figure with a non-zero volume. The CIC measure of AMI, associated with this figure, is $\frac{1}{3}$ (assuming the size of the COS is 2).

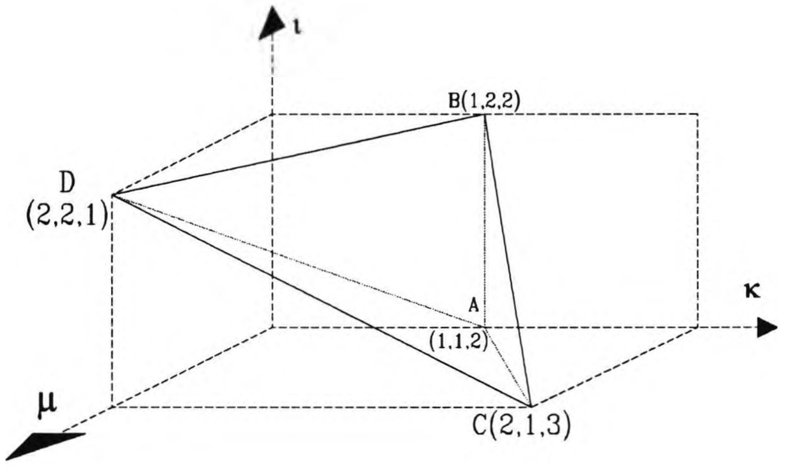


Fig. 6.7 The 3D representation of the AMI code

The assessment of other codes from this category, whose spectral density plots are shown in Fig. 6.8, reveals various relations between the respective geometrical figures and their spectrum shaping capabilities.

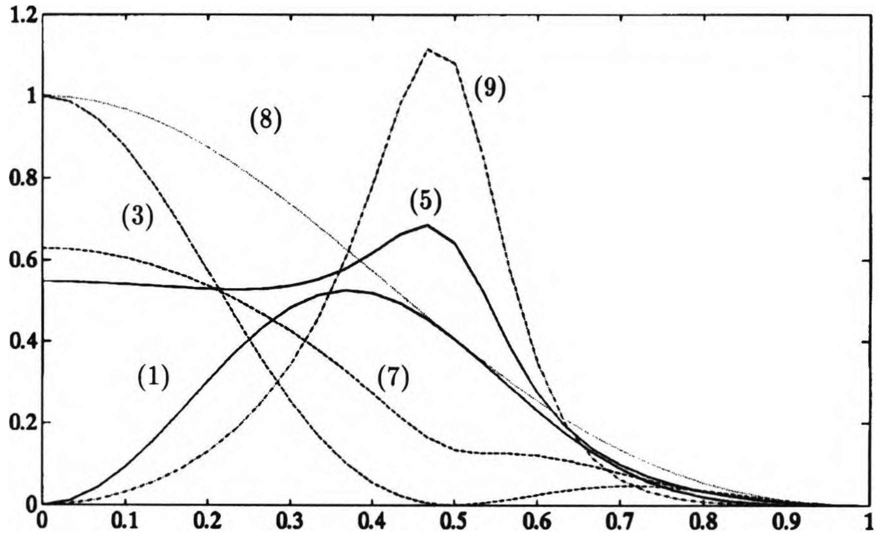


Fig. 6.8 A selection of PSD plots from the $D2S2X3$ category

The shape ABCD in Fig. 6.9a, for example, corresponds to a code producing the NRZ plot, (8) from Fig. 6.8. Its zero volume represents the absence of spectrum modification. All investigated coding structures, which give the NRZ plot, correspond to rectangular or triangular 2D figures. The ones, which introduce some degree of ambiguity, are represented by planar shapes with areas smaller than that of the ABCD rectangle in Fig. 6.9a. One intermediate conclusion from the above results is to omit the lower complexity level structures when applying the CIC evaluation method to a particular category.

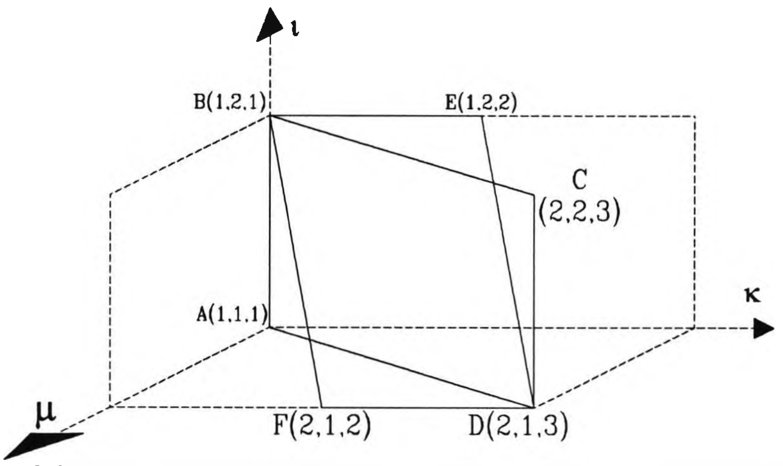


Fig. 6.9a Codes with zero volumes and low spectrum shaping potential

The two triangles, ABC and ADE, shown in Fig. 6.9b, correspond to the codes specified in Tables 5.12a,b, respectively. The zero volume of the two shapes is in accordance with the fact that the redundancy introduced by the two schemes is insufficient to produce any significant modification of the PSD. The noticeable changes of the two spectra (5) and (7) in Fig. 6.8, however, indicates some coding capability of the schemes from Tables 5.12a,b, which can be associated with the area of the two shapes in Fig. 6.9b.

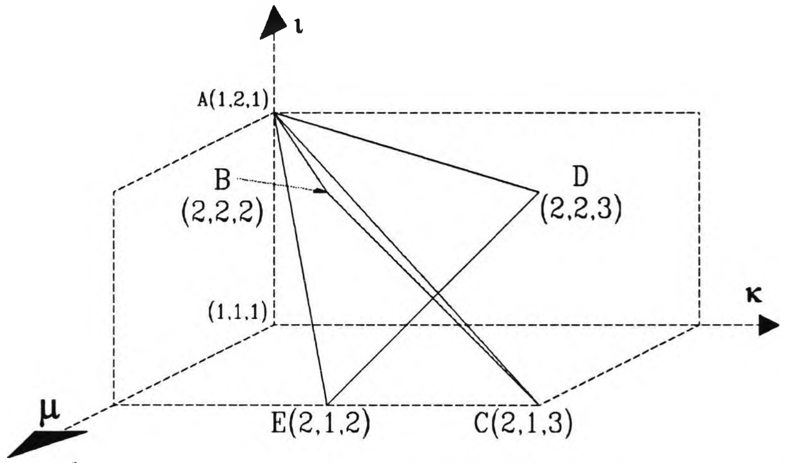


Fig. 6.9b Codes with zero volumes and low spectrum shaping potential

The last two structures from this category to be assessed are Duobinary and the one specified in Table 6.2. The geometrical shape of the latter is shown in Fig. 6.10, while that of Duobinary is determined by points B, E, D and F in Fig. 6.9a.

D_m	s_1	s_2	S_m	Z_m
0	$-1,\ s_2$	$0,\ s_2$	$\begin{bmatrix}0 & 1\\ 0 & 1\end{bmatrix}$	$\begin{bmatrix}-1\\ 0\end{bmatrix}$
1	$-1,\ s_2$	$+1,\ s_1$	$\begin{bmatrix}0 & 1\\ 1 & 0\end{bmatrix}$	$\begin{bmatrix}-1\\ +1\end{bmatrix}$

Table 6.2 A sample $D2S2X3$ code

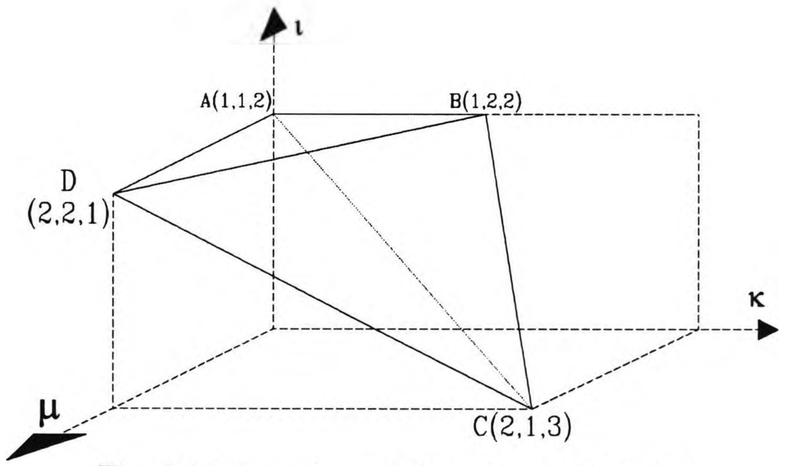


Fig. 6.10 The volume of the code from Table 6.2

The two figures can be associated with the respective CIC through the following considerations. The redundancy introduced by the Duobinary scheme provides for the familiar reduction of the coded signal bandwidth, plot (3) in Fig.

6.8. The spectrum shaping, however is achieved by the use of state-dependent decoding plus the introduction of some ambiguity. This explains the zero volume of the code's geometrical figure and indicates the possibility to find another structure from the same group, which achieves the same bandwidth effect, without the introduction of ambiguity, i.e. by having bigger CIC. The other code represents the maximum possible PSD modification for this category: suppressed low-frequency components and reduced bandwidth combined in plot (9), Fig. 6.8. This is reflected in the volume of its 3D shape, Fig. 6.10, which is $\frac{1}{6}$. The smaller information capacity measure of this code, compared to that of AMI, confirms the superiority of the latter. AMI achieves the familiar spectrum modification, shown as plot (1) in Fig. 6.8, without the ambiguity of the code from Table 6.2.

Despite their relatively small number, the examples selected to illustrate the method for evaluation of the CIC for this group, have demonstrated the most typical relations between the spectrum modification capabilities and the geometrical shapes of various $D2S2X3$ structures. With the increase of complexity the 3D interpretation of the CIC for higher level categories becomes difficult to show as drawings. This does not affect the validity of the evaluation method, but a powerful computational environment is required for the accurate specification of the geometrical figures. To avoid complexity in the presentation of the results, only a few codes from other categories are described in the next subsection.

6.2.3 The CIC of Structures from Higher Complexity Levels

The results from the simulation of $D2S2X4$ codes, discussed in Chapter 5, have indicated that, in spite of the large variety of PSD shapes, only a relatively small number appear to be of practical interest. Plots (4) and (6) in Fig. 5.6b have been mentioned as showing a significant degree of spectrum shaping, achieved with codes, whose specification is given in Tables 6.3a,b, respectively.

D_m	s_1	s_2	S_m	Z_m
0	00, s_2	01, s_2	$\begin{bmatrix}0&1\\0&1\\0&1\\1&0\end{bmatrix}$	$\begin{bmatrix}00\\01\\11\\11\end{bmatrix}$
1	11, s_2	11, s_1		

a

D_m	s_1	s_2	S_m	Z_m
0	01, s_2	00, s_2	$\begin{bmatrix}0&1\\0&1\\0&1\\1&0\end{bmatrix}$	$\begin{bmatrix}01\\00\\11\\11\end{bmatrix}$
1	11, s_2	11, s_1		

b

Table 6.3 Sample code structures with identical 3D volumes

The 3D volume, ABCD , shown in Fig. 6.11, represents the information capacity of these two codes. This example demonstrates the possibility of different codes to have identical volumes, although the spectra they produce are different.

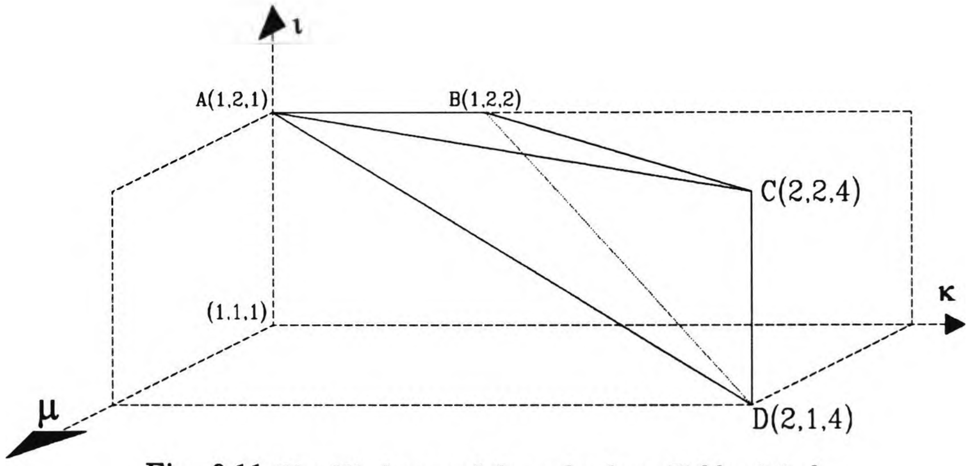


Fig. 6.11 The 3D shapes of the codes from Tables 6.3a,b

Such a result confirms that the CIC is an adequate measure of the coding capability, which can be viewed as producing certain 'amount of change' of the signal spectrum, irrespective of the actual shape of individual spectral density functions. In other words, for a given amount of redundancy, therefore CIC, it is possible to find codes which modify different spectral features of the coded signal. Thus, at a particular complexity level, the coding capacity of a group of structures may allow some of them to reduce the bandwidth only, while others to change the proportion of certain frequency components, without affecting the width of the spectrum. In addition, it is possible to observe the relation between different values of CIC and different degrees of modification of the same spectral feature (e.g. different levels of suppression of the low-frequency components). In fact many of the effects, mentioned above, have been discussed in Chapter 5 with respect to the collections of the PSD plots produced by schemes from the $D2S2X4$ and higher complexity categories.

The coding scheme, specified in Table 6.4a, requires only a brief reference. This structure can be easily recognised as being equivalent to the Manchester code and its familiar PSD is given as plot (8) in Fig. 5.6b.

D_m	s_1	s_2	S_m	Z_m
0	01, s_2	01, s_2	$\begin{matrix}0&1\\0&1\end{matrix}$	$\begin{matrix}01\\01\end{matrix}$
1	10, s_2	10, s_1	$\begin{matrix}0&1\\1&0\end{matrix}$	$\begin{matrix}10\\10\end{matrix}$

Table 6.4a A Manchester type scheme

This scheme is represented by the triangle ACE in Fig. 6.12. The zero volume indicates that the code from Table 6.4a does not belong to $D2S2X4$ category as it requires only one state. Indeed the same output words are assigned to each input symbol regardless of the coder state. This example suggests the possibility to interpret geometrically the inappropriate appearance of a lower category code[4] at a higher complexity level.

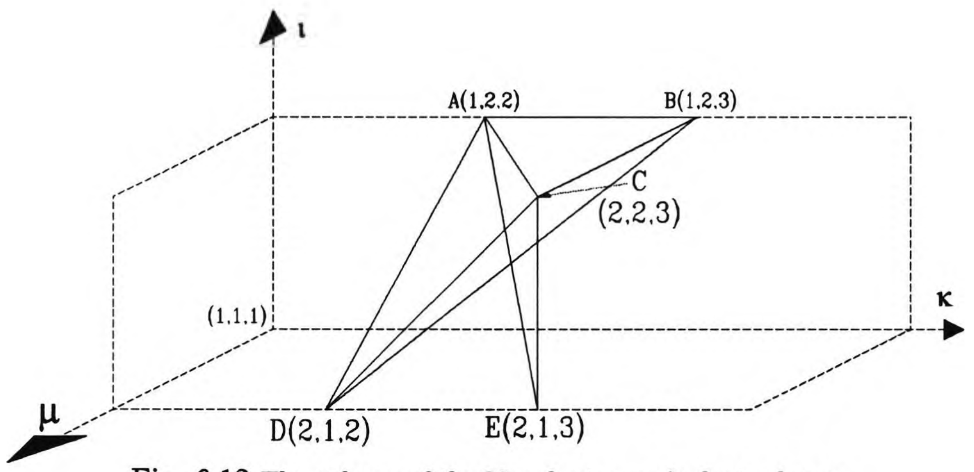


Fig. 6.12 **The volume of the Manchester equivalent scheme**

D_m	s_1	s_2	S_m	Z_m
0	01, s_2	10, s_2	$\begin{bmatrix} 0 & 1 \\ 0 & 1 \\ 0 & 1 \\ 1 & 0 \end{bmatrix}$	$\begin{bmatrix} 01 \\ 10 \\ 10 \\ 01 \end{bmatrix}$
1	10, s_2	01, s_1		

Table 6.4b **A Manchester type scheme**

It can be shown, however, that plot (8) from Fig. 5.6b represents two-state Manchester type schemes, too. One of them is specified in Table 6.4b. The code exhibits properties of a differential scheme. It transforms long sequences of input 0-s into sequences of 01 code words, while long sequences of input 1-s are coded. as alternating pairs of output symbols, 10 and 01. Although the structure from Table 6.4b has the same PSD as that of the Manchester code, it has some additional properties which come into effect for unequal input symbol probabilities. Such a result can be interpreted as some low degree of spectrum modification capability, which at this complexity level is reflected by the CIC represented as the pyramid ABCD in Fig. 6.12. The volume of the 3D figure, which measures $\frac{1}{18}$ of the COS, is relatively small and suggests the possibility to find more efficient schemes in the $D2S2X4$ category or codes of lower complexity, which perform equivalently to schemes from higher categories with non-zero, but very small volumes, relative to the size of the operational space within which the latter are specified.

Two more codes from the $D2S2X4$ category are considered with respect to their information capacity measures. One of them is the CMI scheme, which has been specified in Chapter 4 and the other has been given as Table 5.13 in Chapter 5. The plots produced by the two schemes have been shown in Fig. 5.6c. The two PSD-s have been described as very close to the optimum spectrum modification level achievable within this category[5]. The 3D shapes, representing the CIC of the

[4] The Table 6.4a structure can be transformed into an equivalent $D2S1X4$ scheme.

CMI and the code from Table 5.13 are shown in Fig. 6.13(a and b), respectively.

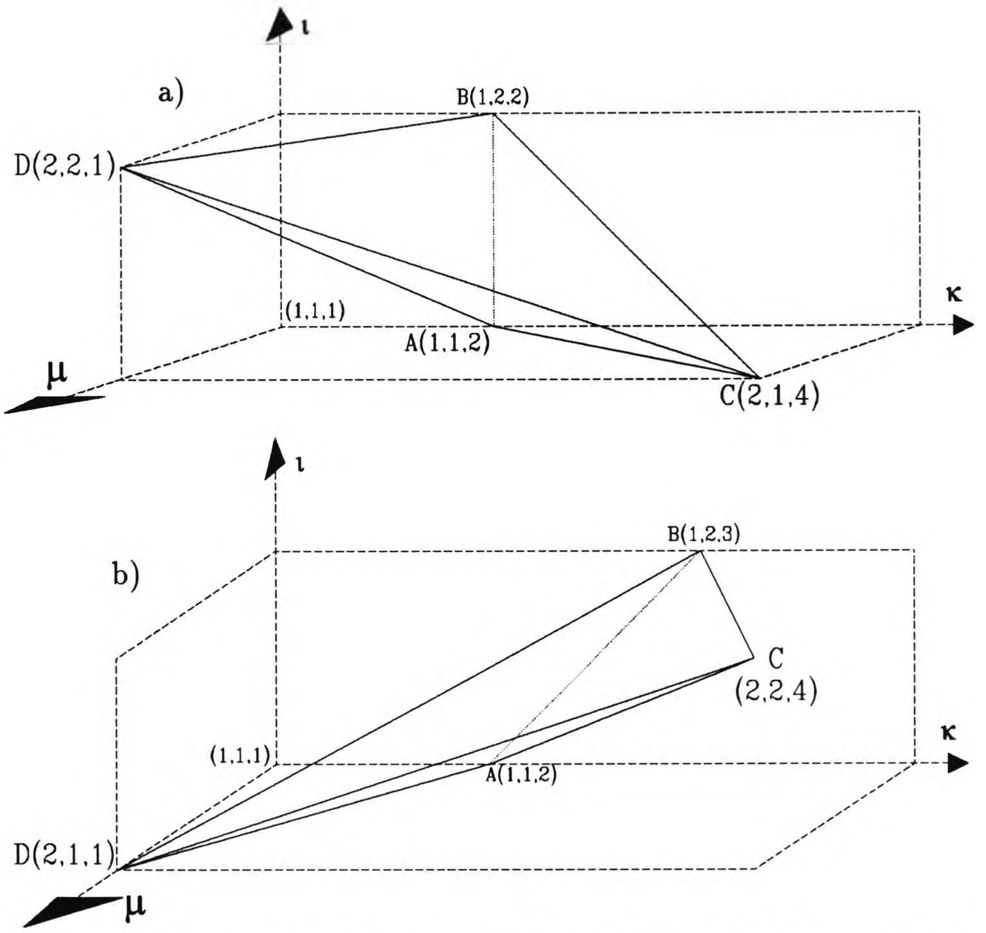


Fig. 6.13 The volumes representing the CIC of the CMI and the Table 5.13 code

The similarity between the spectral shaping effects achieved by the two structures reflects in their geometrical representation. The volumes of the shapes are $\frac{1}{2}$ for the CMI and $\frac{1}{3}$ for the other code. This result complements the conclusions about the PSD shapes from Fig. 5.6c, given at the end of Subsection 5.2.1.2. It confirms the suggestion that the coding scheme from Table 5.13 achieves a better shape[6] of the spectrum, but the effect is produced by redistribution of the code's redundancy, rather than having larger coding capability.

The interpretation of the information capacity of the two codes, discussed above, can be summarised as follows. The CMI and the Table 5.13 codes introduce a certain amount of redundancy, which is associated with their coding capability through the 3D shapes from Fig. 6.13. The CMI volume of $\frac{1}{2}$ represents CIC, which allows for some degree of spectrum modification plus error detection[7]

[5] The conclusions have been reached empirically, by examining all appropriate spectra.

[6] in terms of low frequency suppression and bandwidth reduction

possibilities. The information capacity of the other code is represented by its volume of $\frac{1}{3}$, which appear to be sufficient for spectrum modification, but cannot provide for error detection capabilities. The above considerations can be extended one step further by suggesting some approximate evaluation of certain spectral and structural features of a code. Comparing the two graphs from Fig. 5.6c it is possible to assume that plot (1) corresponds to a smaller 'amount' of spectrum modification than plot (2). At the same time, if the spectral features of PSD (2) are the only 'achievement' of the code from Table 5.13, then the coding capability, determined by a volume of $\frac{1}{3}$, can be considered the exact CIC required to produce the 'half-bandwidth bell-shape' spectrum. In this respect, the CIC of CMI can be represented as $V_{\text{sp}} = V_{\text{sp}} + V_{\text{er}}$. The first part of the volume, V_{sp}, where $0 < V_{\text{sp}} < \frac{1}{3}$, can be viewed as the amount of information capacity required to produce the shape of the CMI spectrum. The remaining part, $V_{\text{er}} = \frac{1}{2} - V_{\text{sp}}$ can be associated with the error-detection capabilities of CMI.

The evaluation of the CIC for the purposes of comparative assessment of codes can be generalised by assuming the COS volume of any category is $V_{\max} = 1$. As this, in effect, is normalisation of the results, the coding capacity measures of the last two codes become $\frac{1}{6}$ and $\frac{1}{9}$ for the CMI and the Table 5.13 structure, respectively. The normalised volumes allow codes from different categories to be compared with respect to their spectrum modification capability. In this case it is interesting to consider the Modified Duobinary scheme, specified in Table 4.11, Chapter 4. Its PSD graph, shown in Fig. 4.12, is almost identical to that of the code from Table 5.13, regarding the spectrum shaping effects. The important difference, however, is revealed by the CIC of the Modified Duobinary. The corresponding 3D shape is given in Fig. 6.14 and its volume is 2.

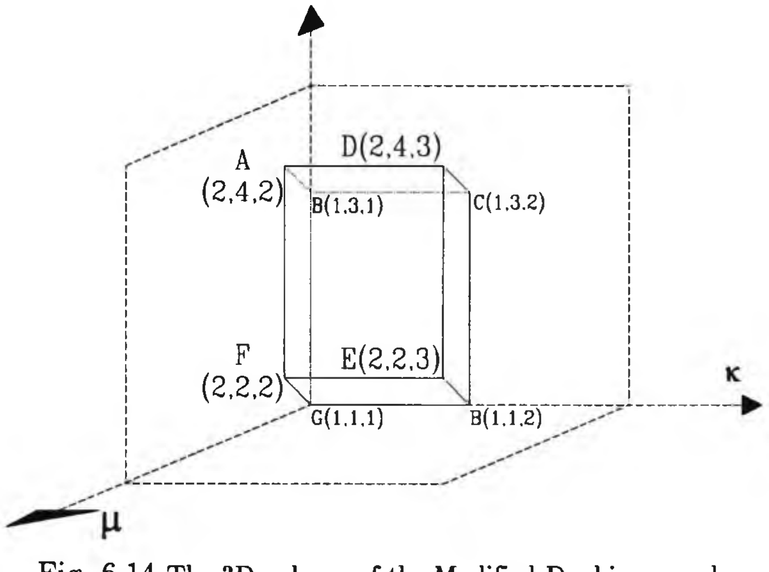


Fig. 6.14 The 3D volume of the Modified Duobinary code

[7] The invalid code word 10 may be used for frame alignment and phase synchronisation.

The full COS capacity for the category $D2S4X3$, which contains the Modified Duobinary structure is $(2-1)\times(4-1)\times(3-1)=6$. This allows the normalised information capacity measure of the code to be determined as $\frac{1}{3}$. The result shows that it is the larger coding capability of the Modified Duobinary, compared to CMI and the Table 5.13 codes, which allows the spectral features of the latter to be achieved without degrading other performance parameters, like the susceptibility to errors, for example. Similar reasoning applies to the sample structures considered next.

In the discussion of various coding schemes from the $D2S2X2$ and the $D2S2X3$ categories (Chapter 5) a few structures have been mentioned to illustrate the spectrum shaping limitations for a particular complexity level. Graphs (4) and (9), from Fig. 5.5b, for example, which feature significant PSD modification effects, have been found to represent invalid code specifications. Based on direct examination of a large number of $D2S2X3$ schemes, it has been concluded that the shape of the two spectra[8] is modified to a degree greater than what is possible at that complexity level, for valid code structures (e.g. plots (1), (3), (5) and (7) from Fig. 5.5b). In other words, the spectral features of PSD-s (4) and (9) have been produced at the expense of essential structural features of the respective schemes, degrading the validity of the latter below the level of practical interest. The applicability of the CIC evaluation method is demonstrated by showing that the conclusions, mentioned above, can be based on accurate estimation of the coding capabilities of different structures. The two schemes discussed below are from the $D2S4X4$ category and their spectra are plots (1) and (2) in Fig. 6.15.

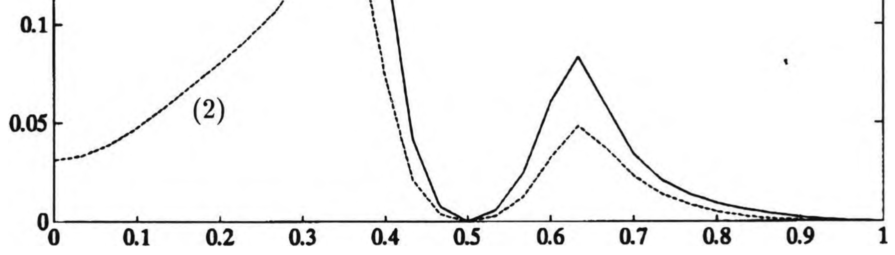


Fig. 6.15 Sample PSD plots of structures from the $D2S4X4$ category

[8] with respect to low frequency suppression and bandwidth reduction

The first one has been specified as version (b) in Table 4.17, Chapter 4 and the second is given by Table 6.5.

D_m	s_1	s_2	s_3	s_4	S_m	Z_m
0	0 0 s_2	0 0 s_4	0 0 s_4	0 0 s_2	0 1 0 0 0 0 0 1 0 0 0 1 0 1 0 0	00 00 00 00
1	0 0 s_3	1 0 s_3	0 0 s_1	0 1 s_1	0 0 1 0 0 0 1 0 1 0 0 0 1 0 0 0	00 10 00 01

Table 6.5 A sample $D2S4X4$ structure

The coding capacity of the structures producing the two PSD-s are represented by the 3D shapes shown in Fig. 6.16 a and b, respectively.

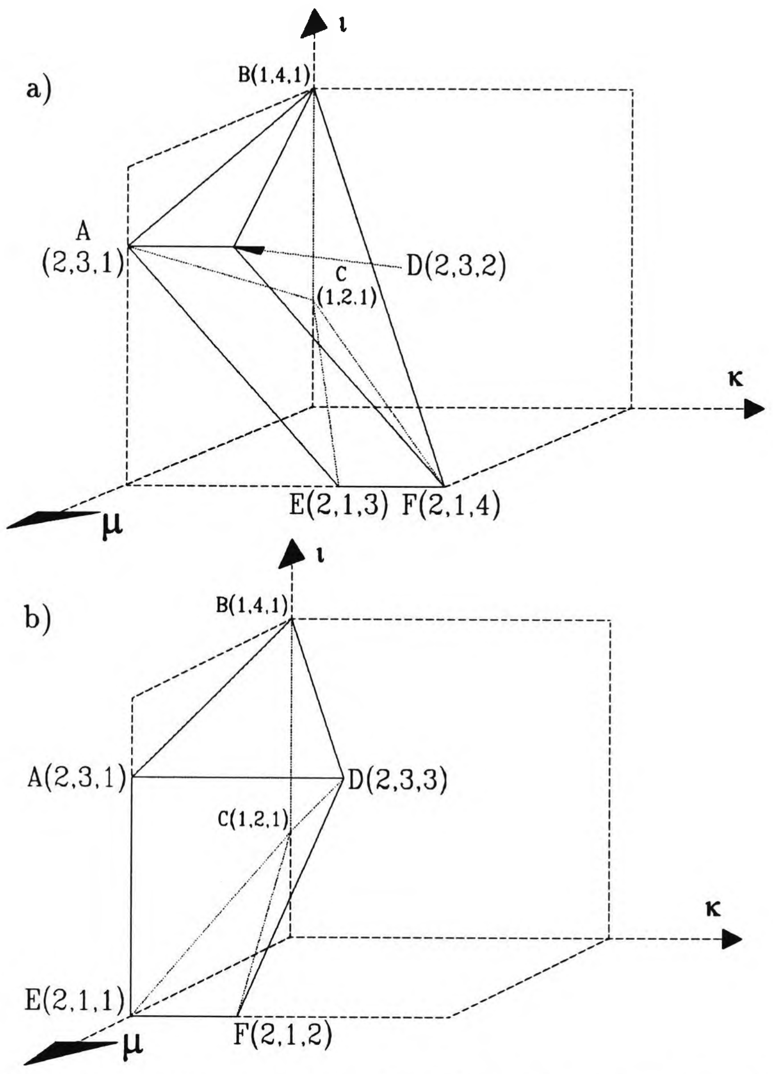


Fig. 6.16 The 3D shapes of the $D2S4X4$ structures from Fig. 6.15

Although the calculation of the volumes of the two figures is more involved than for the previous examples, it is still achievable without the use of computing facilities. The result in normalised form is $\frac{5}{27}$ for both shapes in Fig. 6.16. The code, which gives plot (2) is regarded as producing a higher degree of spectrum modification[9] than the code with plot (1), for the same information capacity measure. The comparison suggests that the spectrum shaping effect of the former is achieved at the expense of its structural characteristics. Indeed, a closer examination of Table 6.5 reveals that it is an invalid structure[10].

The discussion of the last two codes indicates the limitations in presenting the simulation results, imposed by the lack of theoretical basis allowing suitable evaluation of the spectral and structural features of codes. The difficulties in estimating the CIC measures and their relation with the spectrum shaping capabilities of codes increase rapidly for categories of higher complexity levels. This is demonstrated by the 3D shapes representing the new codes proposed in Chapter 4 and shown in Fig. 6.16. The complexity of the shapes, representing the information capacity of coding schemes and the respective PSD plots, requires strict and accurate analytical definition, relating the coding capability volume to the spectral and structural features of codes. Although, in some cases, it may still be difficult to illustrate the CIC of a particular structure with a suitable drawing, in principle there is no restriction to the possibility to evaluate the respective volume, if adequate computational resources are available. The development of the complete theoretical frame, which would allow more precise comparative assessment of the analysis results, is a matter of considerable additional research effort. The purpose of this chapter is to outline the concept of the suggested CIC evaluation method. The main intention is to illustrate the possible development of the uniform assessment and analysis techniques, discussed in the thesis, into a complete systematic approach towards the research and design of digital codes.

[9] Suppression of the low frequency components for a bandwidth of half the code-symbol rate.

[10] States s_1 and s_3 transform consecutive input 1-s into an all-zero sequence, which is indistinguishable from one representing consecutive input 0-s.

7. BINARY MULTIPLEXED CODING FOR MULTICHANNEL TRANSMISSION

The capabilities of the general spectral analysis procedure can be demonstrated through its implementation for the assessment of a new coding scheme suggested and described in this chapter. The most essential features of this scheme are:

- use of Pseudo-Random Binary Sequences (PRBS);
- possibility to combine multiplexing and line coding.

The coding method presented below is based on the principles of Spread Spectrum (SS) and is related to Code-Division Multiplexing (CDM). However there are some major differences in both, the use of SS sequences and the implementation of codes for multiplexing. The main ideas are further revealed after a brief introduction to the principles of spectrum spreading.

7.1 Basics of SS for Coding

SS techniques are employed to achieve advantageous reception of signals in the presence of interference, by the means of selective processing. Prior to transmission the energy of the information signal is distributed over a frequency range, much wider than its original bandwidth, Fig.7.1 below.

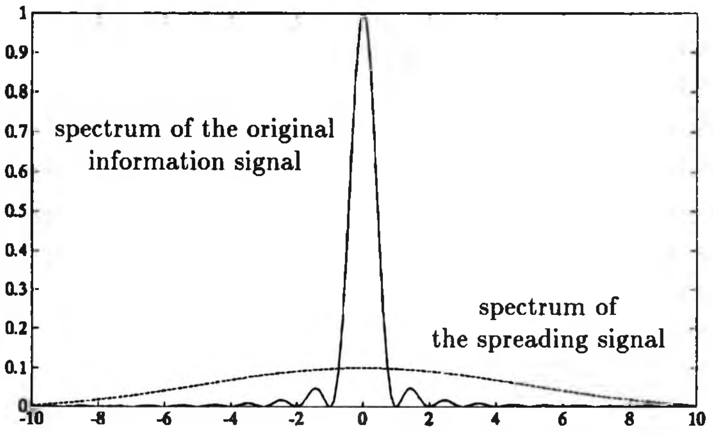


Fig.7.1 Spectral density functions of the information and the spreading signals

The signal at the receiving side undergoes processing which is uniquely related to the spectrum spreading procedure used by the transmitter. As a result the information signal is transformed back to its original frequency range. The same process spreads any noise and interference components of the received signal over a spectrum, much wider than the bandwidth of the information signal. Consequently the amount of unwanted signals within the frequency range of interest is significantly reduced.

Spectrum spreading is accomplished by the use of binary sequences, known as codes. Some of the most popular SS techniques are Direct Sequence (DS) modulation, frequency hopping and time hopping. These and other methods of SS communications have received considerable attention in recent years and the relevant theory has been well documented, [12,13,14,35]. Although the question about the 'extraordinary' capabilities of SS systems is still under debate, [36,37] one thing has become clear: spectrum spreading can be used to reduce significantly the effects of interference, [34]. A general view, which appears to be widely accepted, is that a SS system, which is not designed for any specific purpose, offers no better performance than conventional modulating systems. In some cases, however, it is possible to achieve certain conveniences by implementing SS rather than other modulation techniques.

Suppression of deliberate narrowband interference is a typical area of SS applications. Although the signals in a SS system occupy a very large transmission bandwidth, this is compensated for by the increased ability to reject unwanted signals. As a result it becomes possible to have many signals simultaneously using the same frequency range.

This presentation does not aim to cover all problems related to SS communications. The purpose of the brief overview is to serve as an introduction to some fundamental aspects of spectrum spreading which would facilitate the forthcoming discussion of the suggested code multiplexing technique.

7.1.1 Frequency Distribution by Using DS Modulation

DS modulation is one of the most common methods for spectrum spreading. Its principle can be described in a simple form as follows: The data signal, $d(t)$, which is a binary pulse sequence, is multiplied[1] with a PRBS, $c(t)$ (also known as the coding sequence), whose pulse repetition rate, F_c is many times higher than that of the data signal, F_d. The spectrum of the resulting signal, $s(t)$ has at least the width of the code-sequence spectrum. The main effect of using SS is achieved at the receiver, where the incoming signal, $r(t)$ is multiplied with a replica of the code. This recovers the information signal back into its original bandwidth. All unwanted signals, which can be viewed as noise, $n(t)$ added to $s(t)$, are spread over the spectrum of the code sequence. As a result

[1] In practice the process of multiplication with the code sequence is some suitable operation (usually modulo-2 addition).

only a fraction of the interference will affect the information signal, which is the actual advantage in using SS and is usually evaluated as the processing gain, G_p of a SS system. This parameter is proportional to the ratio $\frac{B_c}{B_d}$, where B_c and B_d are the bandwidths of the code sequence and the data sequence respectively. The processes involved in a DS SS system are illustrated in Fig.7.2. The simple analytical expressions, given below, represent the main signal transformations, describing the principle of spectrum spreading.

$$s(t) = d(t)\; c(t) \tag{7.1a}$$

$$r(t) = s(t) + n(t) = d(t)\; c(t) + n(t) \tag{7.1b}$$

$$r(t)\; c(t) = d(t)\; c^2(t) + n(t)\; c(t) = d(t) + n(t)\; c(t) \tag{7.1c}$$
$$\{c^2(t) = 1 \text{ assumes maximum autocorrelation}\}$$

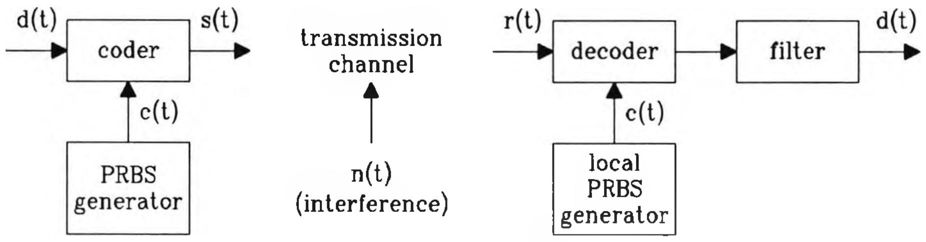


Fig.7.2 A general diagram of a DS SS communication system

The power advantage of the signal over the unwanted interference after decoding at the receiver can be shown through the following simple considerations. The data pulse interval and the code pulse interval can be expressed as $T_d=\frac{1}{F_d}$ and $T_c=\frac{1}{F_c}$, respectively and are related through the number of code pulses, N used for every data bit, i.e. $T_d = NT_c$. Assuming the bandwidths, B_d and B_c are directly proportional to the respective pulse repetition rates, F_d and F_c, the former can be determined from the corresponding spectral density functions for rectangular pulse waveforms.

$$S_d(f) = T_d\left(\frac{sin(\pi f T_d)}{\pi f T_d}\right)^2 \quad \text{and} \quad S_c(f) = T_c\left(\frac{sin(\pi f T_c)}{\pi f T_c}\right)^2 \tag{7.2}$$

The improvement of the system performance due to spectrum spreading can be evaluated by comparing the signal-to-noise ratio at the input, S/N_{in} to that at the output, S/N_{out}. In these expressions S corresponds to the power of $d(t)$, while N_{in} corresponds to the total noise power of the signal $n(t)$, which is distributed over B_c as a result of the process of despreading, given by (7.1c). The fraction of the noise power, N_{out} which remains within the bandwidth of the information signal at the output of the filter in Fig.7.2, can be related to the total noise power through $N_{\text{out}} = \frac{B_d}{B_c} N_{\text{in}}$. Finally the power advantage of the data signal over the unwanted interference, achieved by spectrum spreading, is given by the processing gain of the system as:

$$G_p = \frac{(S/N_{\text{out}})}{(S/N_{\text{in}})} = \frac{N_{\text{in}}}{N_{\text{out}}} = \frac{B_c}{B_d} \tag{7.3}$$

The relation given by (7.3) shows that the performance improvement in a SS system, for a fixed amount of interference, is proportional to the ratio of the transmission rates, $\frac{F_c}{F_d}$. This indicates the need to use sufficiently long code sequences to make the implementation of SS techniques of practical interest. The codes which are in common use at present are of great variety and posses a number of special features. A brief summary of the most typical of the spreading sequences is given in the next subsection.

7.1.2 Spreading Sequences

The best results from spectrum spreading are achievable by using purely random sequences. In practice, however, only pseudo-random codes can be employed. This is due to the requirement of having an exact replica of the spreading sequence available at the receiving side. The code signals in real SS systems are usually generated by shift registers whose most valuable property is the possibility to produce long sequences with relatively simple circuits[2]. Regardless of their origin, the spreading sequences should posses certain characteristics, the most important being:

- features of randomness;
- simplicity of generation.

The requirement for randomness ensures appropriate correlation properties, which provide for the most significant feature of the codes for multiuser SS systems – orthogonality.

[2] Other methods for generating PRBS-s also exist.

Spreading sequences are usually produced by linear Shift-Register Generators (SRG-s). The length and the cycle of an SRG sequence depends on the number of stages and the feedback connections. Particular shift-register circuits generate PRBS-s whose cycle of repetition is $N = 2^n - 1$, where n is the number of stages of the SRG. Such sequences are known as maximal-length codes[3] and they exhibit properties very suitable for spectrum spreading. Unfortunately, the number of different m-sequences for a given n is relatively small, which restricts their use in multichannel communication systems.

Various alternative solutions to this problem have been found with different degrees of compromise with some of the properties of the codes. One of the well known results is the set of Gold sequences, [12,38] which can be generated by simple modulo-2 addition of the outputs of two m-sequence SRG-s. Gold codes have reasonably good correlation and overall randomness properties. The number of different Gold sequences for a given cycle length, N is greater than the corresponding number of m-codes. A simple example, which illustrates the two types of sequences is given below. The circuit diagrams in Fig.7.3 show two possible configurations of a 4-stage maximal-length SRG.

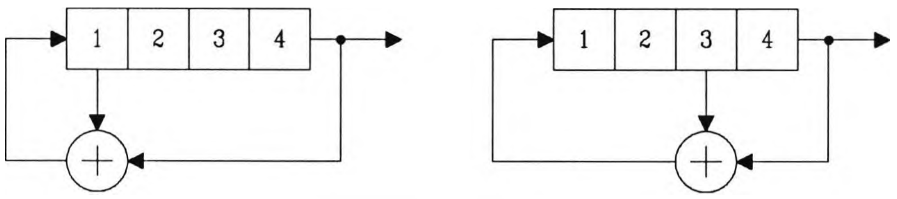


Fig. 7.3 4-stage SRG circuits

The sequences produced by the two circuits are of length $N = 2^4 - 1 = 15$ and the second one is equivalent to the first in reverse order, i.e.:

a) 1 1 1 1 0 1 0 1 1 0 0 1 0 0 0
b) 0 0 0 1 0 0 1 1 0 1 0 1 1 1 1.

There are analytical expressions which gives the exact number of all possible m-sequences for a given size of the SRG, [12]. For $n = 4,5,6$ the respective numbers of m-sequences are $N = 4,6,4$, while at the same time the corresponding number of Gold codes for n = 4 is 272. Three m-sequences are shown as pulse waveforms in Fig. 7.4 for $N = 7, 15$ and 31. (The code-symbol interval is assumed the same for the three samples.)

[3] m-sequences or m-codes

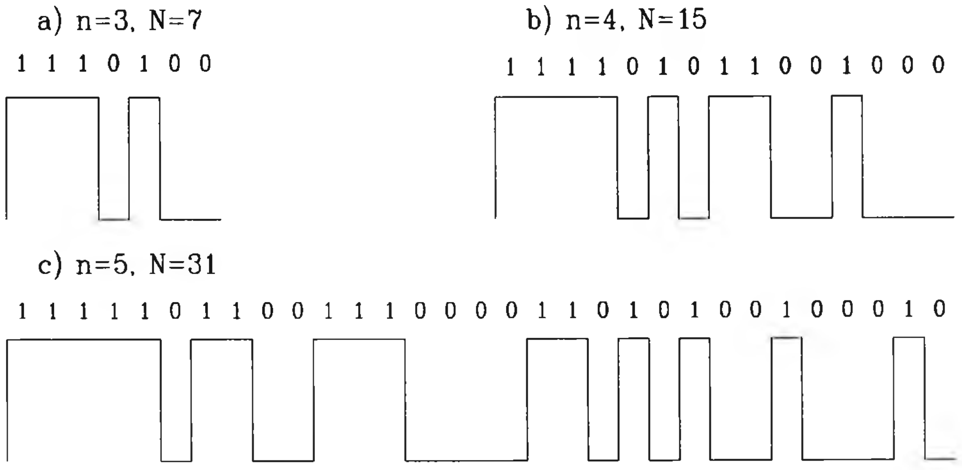


Fig.7.4 Complete cycles of 3 m-sequences of different length

The set of Gold codes produced by modulo-2 addition of sequence a) and all shifted versions of sequence b) above is:

```
1 1 1 0 0 1 1 0 1 1 0 0 1 1 1
1 1 0 1 0 0 1 1 0 0 1 0 1 1 0
1 0 1 1 1 0 0 0 1 1 1 0 1 0 0
0 1 1 0 1 1 1 1 0 1 1 0 0 0 0
1 1 0 0 0 0 0 0 0 1 1 1 0 0 1
1 0 0 1 1 1 1 0 0 1 0 1 0 1 0
0 0 1 0 0 0 1 0 0 0 0 1 1 0 0
0 1 0 1 1 0 1 0 1 0 0 0 0 0 1
1 0 1 0 1 0 1 1 1 0 1 1 0 1 1
0 1 0 0 1 0 0 1 1 1 0 1 1 1 0
1 0 0 0 1 1 0 1 0 0 0 0 1 0 1
0 0 0 0 0 1 0 0 1 0 1 0 0 1 0
0 0 0 1 0 1 1 1 1 1 1 1 1 0 1
0 0 1 1 0 0 0 1 0 1 0 0 0 1 1
0 1 1 1 1 1 0 0 0 0 1 1 1 1 1
```

A full overview of the huge variety of different spreading sequences is not intended in this work. The main purpose of this section is to present briefly the main concepts of spectrum spreading, especially with respect to their implementation for multichannel communications.

7.2 SS for Simultaneous Signal Transmission

The improved interference suppression, achievable through spectrum spreading, has shown a great potential for applications where many users communicate simultaneously, over the same transmission media, occupying the same frequency bandwidth. The main consideration, regarding such multiuser systems, is to provide a set of spreading codes which would allow for reliable detection of the required signal, while ensuring a successful rejection of the combined interference effects from the unwanted signals of the other users. The most significant features associated with spreading sequences, suitable for such applications, are the high autocorrelation and low crosscorrelation values.

The SS methods used for multichannel communications are known as Code-Division Multiple Access, (CDMA). There is a great variety of techniques for simultaneous access of the transmission medium. For the purposes of this presentation, however, the basic principles can be introduced through a brief description of DS CDMA.

The main idea is to provide all users with unique codes which are approximately orthogonal. A simple illustration of the communication process in the case of DS CDMA is given in Fig.7.5.

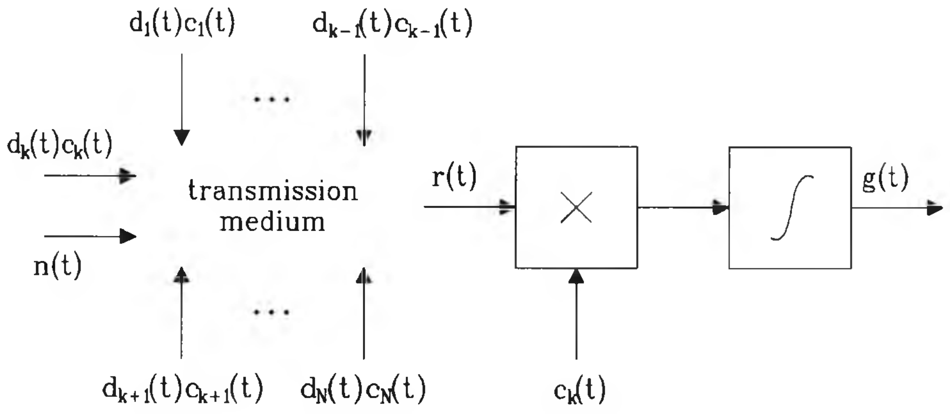


Fig.7.5 Multichannel SS transmission for DS CDMA

The data signals, $d_i(t)$, $(i = 1,\dots,N)$ of N users, communicating simultaneously on the same transmission channel, are individually multiplied with the respective code sequences, $c_i(t)$. The resulting signals combine to produce the signal $r(t)$ which is received by the communicating stations, i.e.:

$$r(t) = \sum_{i=1}^{N} d_i(t-\theta_i)\; c_i(t-\theta_i) + n(t) \tag{7.4}$$

where $n(t)$ is the additive noise in the transmission channel. The random phase parameter, θ_i in (7.4) indicates that in the general case the users of a CDMA system communicate asynchronously. The process of recovery of the data signal,

$d_k(t)$ at the receiving side takes place when the incoming signal is synchronised with the locally generated replica of the code, $c_k(t)$, i.e. $\theta_k = 0$. The result after integration is given by the following expression:

$$g(t) = d_k(t) + \frac{1}{T}\sum_{\substack{i=1 \\ i \neq k}}^{N} d_i(t-\theta_i)c_i(t-\theta_i)c_k(t) + \frac{2}{T}\int_0^T n(t)c_k(t)\, dt \qquad (7.5)$$

The second and the third terms in (7.5) are the interference which is spread over the frequency bandwidth of $c_k(t)$. Most of the interference components from other users and the background noise are filtered out with respect to spectrum of $d_k(t)$.

The implementation of SS techniques for multiuser access of the same transmission medium simultaneously or by occupying a common frequency band, or both is usually referred to as CDM. A considerable number of problems have to be addressed in the design of a SS CDMA system. Some of the most significant are:

- number of codes of a particular length with good
  - autocorrelation,
  - orthogonality and
  - randomness;
- synchronisation;
- near-far effect;
- multipath transmission.

A major condition for solving these problems is the availability of wide transmission bandwidth[4]. Fibre optic systems provide a very suitable environment for multiuser transmission based on CDM. Interesting examples are given in [38,39]. The type and the number of codes available are extremely important and these matters have been well studied and thoroughly investigated in the literature [12-14]. In practical systems the most difficult problems to overcome are related to the near-far effects and synchronisation. The possibility of correct decoding depends heavily on the power contribution of the different sources to the signal in the communication channel. This brings the requirement to ensure careful balance in combining the coded signals on the transmission line. Additional complications arise from asynchronous multiplexing of the SS signals. Various solutions to these problems have been suggested and some original results are given in [39,40].

[4] Required for the use of sufficiently long coding sequences with reasonably high information transmission rates.

7.3 SS Multiplexing for Multichannel Transmission

PRBS-s for spectrum spreading have been used very successfully in some cases of interference suppression and in certain multiple access communication systems. The problems associated with SS applications are much easier to solve for balanced[5] and synchronous[6] transmission. Although the possibilities for useful implementation of SS techniques in point-to-point communications may not seem to be significant, the results from spectrum spreading applications in other areas have often been so surprising and promising, that CDM in this type of communications is still worth some research effort.

A new channel multiplexing method, based on the principles of spectrum spreading is proposed in this section. The method is further enhanced to incorporate some objectives of line coding, which is also discussed below. The main concept is a combination of the processes involved in conventional Time-Division Multiplexing (TDM) and DS CDMA.

In the time domain the method can be described as a simultaneous transmission of binary signals, $d(kT)$ from N different information channels, so that N bits, one from each channel, are transmitted for the duration of one bit interval, T. This is achieved by assigning unique code sequences, each one representing a single bit (according to rules discussed later in this section). For simplicity of the presentation it is assumed that all signals have the same transmission rate, i.e. equal bit intervals. The actual combination of N codes for every bit period can be done in many different ways and the ones, which are of practical interest have to be determined through a careful balance between the complexity of the system and its efficiency.

In a conventional DS SS system the simultaneous transmission of the coded signals from all channels results in a multilevel signal being received by the communicating users. As mentioned earlier, in the case of a FO system, where the only interference comes from the signals of the unwanted channels, the configuration of the communication lines is of crucial importance as a proportional contribution of the signal power by the different sources has to be ensured, [38].

The use of a multilevel signal for the sole purpose of multiplexing a number of binary channels is of no great practical value, especially in a FO communication system. It appears possible, however, to preserve some of the benefits from implementing SS techniques, without compromising the advantage of binary signalling. A Binary-Multiplexed Code (BMC) technique, is suggested in

[5] Equal (or proportional) contributions from all sources to the signal on the communication line.

[6] Coincidence of the symbol intervals in the code sequences of all users.

this chapter to achieve this goal.

Some straightforward ideas about a BMC using PRBS-s are described in [40], (Appendix-B). The technique presented in this reference is based on simple transformations of a multilevel signal, produced by direct addition of the spreading codes into a binary sequence, according to the following rules: If the number of multiplexed data channels is N, then the multilevel pulses, of duration one code symbol period, can assume any integer value from the range $0 - N$. The corresponding two level signal consists of pulses with a period of repetition $\frac{T}{N}$ and duration $\frac{xT}{N^2}$, where $x \in \{0,\ldots,N\}$ is the value of the respective multilevel pulse. Obviously, such a simple transformation results in a significant increase of the required transmission bandwidth. Further processing of the multiplexed signal has been suggested in the same reference to overcome this disadvantage.

The efficiency of the two-level SS multiplexing technique, outlined above, can be improved considerably by adopting a different approach. It is based on the fact that the received signal in a DS CDMA system is usually discriminated against a suitably determined threshold level. Therefore the process of recovery of the original information signal is performed by applying the unspreading code sequence in effect to a two level signal. These considerations suggest that in a communication system, where the signals from all data channels are available at the transmitting side, it is possible to implement a SS BMC which includes the process of threshold discrimination of the multilevel signal, produced by additive combination of the respective code sequences. The resulting line signal is in a binary form which is preferred for most modern communication systems, especially those on optical fibres. The main idea of the suggested method for binary SS multiplexing derives from the fact that in multichannel point-to-point transmission the major interference factor is the signal contributions from the co-channels. Theoretically, for error free communication the principles of spectrum spreading apply to the described method regardless of the transmission process.

7.3.1 The BMC Technique for Two-Level Multichannel Transmission

In the case of multiplexing N binary information channels it is necessary to construct a set of at least N spreading sequences. This number of codes is sufficient if the simplest form of binary data coding is adopted, i.e. a code sequence is sent for a symbol 1 and no sequence for a symbol 0. In most real SS systems, however, another code sequence is used to represent a binary 0. Often the second sequence is the inverse or the reverse of the first one. Thus, for the multiplexing of N channels, N different codes are required to represent the binary 1-s of each channel and, eventually, N complementary sequences for the 0-s. The diagram in Fig. 7.6a illustrates the basic processes involved in binary-multiplexed coding of data signals $d_i(kT)$, $i = 1, \ldots, N$. The PRBS-s $c_i(kT)$ are used for data 1 and their inverse, $\overline{c}_i(kT)$, for data 0.

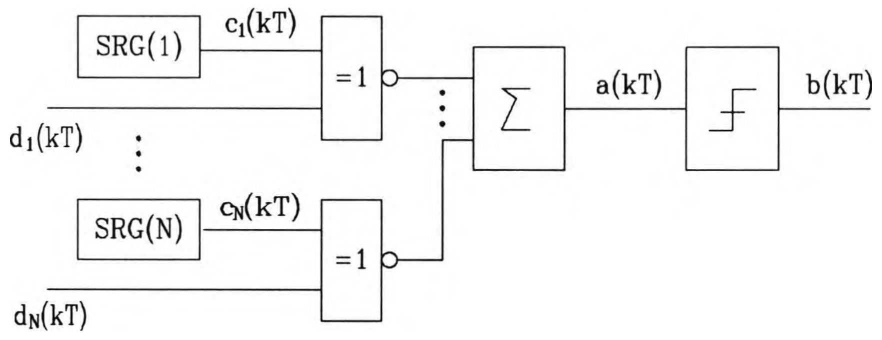


Fig. 7.6a Binary code multiplexer

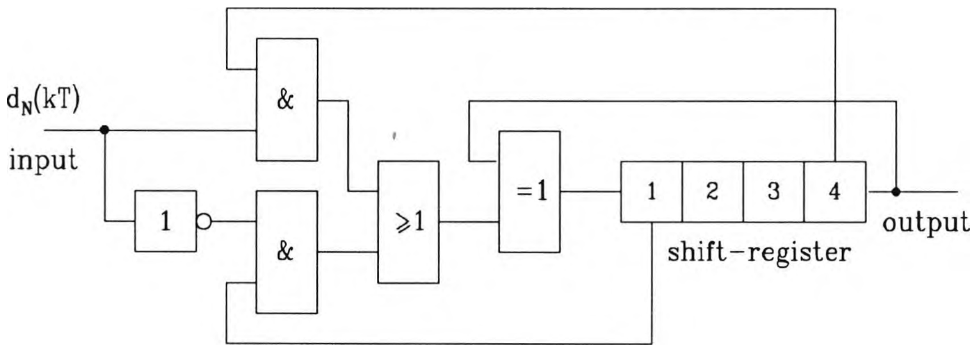


Fig.7.6b A forward and reverse sequence generator

The signal a(kT) at the output of the summation block Σ is multilevel and is transformed into the binary signal b(kT) by the threshold discriminator ⨍. In this particular case the use of the inverted sequences is chosen only for the simplicity of illustration. No technical difficulties exist in constructing a multiplexer, which generates the reverse of the respective code sequences. This can be done by using the data signals to switch between two set of feedback connections for each SRG. The concept is illustrated in Fig.7.6b for a single data channel implementing a 4-stage SRG.

As already mentioned, the performance results of SS systems are of practical value if the code sequences are of sufficient length. Although some of the examples given later are based on relatively short PRBS-s, they are used only to illustrate the principle of the suggested method and do not restrict its generality. The implementation of long spreading sequences does not require directly proportional increase of the code multiplexer complexity. The latter depends on the number of SRG stages which is logarithmically related to the length of the codes.

As the number of m-sequences of length N is smaller than N, Gold sequences are often used for a BMC. The set of Gold codes produced from a pair of maximal PRBS-s of length N is $N+2$ (including the generating pair itself). Their autocorrelation and orthogonality properties are satisfactory, [38] and the simplicity of their generation makes them attractive for various applications. Assuming that the circuit in Fig.7.6a uses codes of length N, one possibility for its practical implementation is to add modulo-2 the m-sequence, produced by one SRG, to the phase-shifts of another m-sequence, produced by a second SRG and $N-1$ delay elements in series.

The BMC schemes, based on SRG-s and combinational logic circuits, provide a considerable potential for optimisation. It is beyond the scope of this work to discuss different code multiplexer designs. A large variety of practical circuits is possible and their performance depends on the specific set of spreading sequences used with a particular BMC.

7.3.2 Analytical Presentation of the Binary-Multiplexed Coding

The multiplexing procedure based on a set of coding sequences can be described analytically in a general form. It is helpful to start with a description of the sets of variables involved in the algebraic relations, as the main requirement is conformity with the arithmetic operations.

The information channels[7], $d_i(kT_d)$, $(i = 1,\ldots,N$ and $k = 0 \pm 1, \pm 2,\ldots)$ are binary signals. It is possible to represent the states of all data inputs, for the duration of one bit interval, T_d as a row vector $d = [d_1,\ldots,d_N]$, where $d_i = 0$ or 1 for $i = 1,\ldots,N$. Every data bit is combined with a sequence, c_i of N code symbols, which is represented as a row vector $C_i = [c_{i1},\ldots,c_{iN}]$. The actual process combining d_i and C_i can be expressed either as a conventional multiplication or as modulo-2 addition. In the first case the spreading of $d_i(kT)$ results in C_i being transmitted for data 1 and 0 for data 0. In the second case data 1 is transformed into C_i, while data 0 becomes the inverse of the code sequence, $\overline{C}_i$. Although multiplication has been adopted in the subsequent presentation (solely for the purpose of notational convenience) the generality of the resulting expressions is not affected. This allows the coding for channel i to be denoted as

$$d_i C_i = \left[d_i c_{i1},\ldots,d_i c_{iN}\right] \tag{7.6}$$

[7] These will be also referred to as data inputs.

The next step is the use of (7.6) in the analytical expression which represents the multiplexing of the set of code sequences $C = [C_i, \ldots, C_N]'$ for a combination of data symbols d. Noting that C is an N by N matrix, the process is given by the following expression:

$$d \times C = \begin{bmatrix} d_1 & d_2 & \ldots & d_N \end{bmatrix} \times \begin{bmatrix} C_1 \\ C_2 \\ \vdots \\ C_N \end{bmatrix} = [d_1 C_1 + d_2 C_2 + d_N C_N] \qquad (7.7a)$$

The result from (7.7a) corresponds to the DS BMC illustrated in Fig.7.6a, as well as to the basic function of any CDM system in which the coded signals are combined additively. The expression in (7.7a) can be further developed by expanding the code matrix, C to determine the components of the signal a at the output of the multiplexer for a given combination of input symbols, d.

$$a = d \times C = \begin{bmatrix} d_1 & d_2 & \ldots & d_N \end{bmatrix} \times \begin{bmatrix} c_{11} & c_{12} & \cdots & c_{1N} \\ c_{21} & c_{22} & \cdots & c_{2N} \\ & & \vdots & \\ c_{N1} & c_{N2} & \cdots & c_{NN} \end{bmatrix}$$

$$= \begin{bmatrix} \sum_{i=1}^{N} d_i c_{i1} & \sum_{i=1}^{N} d_i c_{i2} & \cdots & \sum_{i=1}^{N} d_i c_{iN} \end{bmatrix} \qquad (7.7b)$$

The result for a is a 1 by N vector whose elements, $a_j = \sum_{i=1}^{N} d_i c_{ij}$ ($j = 1, \ldots, $N) are integers from the range of 0 to N and represent the values of the multilevel signal produced through the BMC. A graphical illustration of such a signal is given in Fig.7.7 for the following example values:

$N = 7$, $d = [0\ 1\ 1\ 1\ 0\ 1\ 1]$

$$C = \begin{bmatrix} 1 & 1 & 0 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 1 & 1 & 1 & 1 & 1 \end{bmatrix}$$

$a = [\,2\ 2\ 3\ 4\ 2\ 2\ 3\,]$

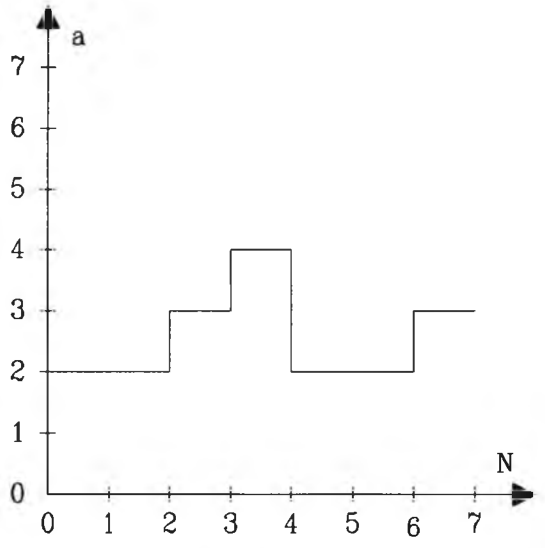


Fig.7.7 Multilevel signal values

The last stage of the proposed BMC technique involves the transformation of the multilevel signal into the binary transmission signal $b(kT)$. This is achieved by a thresholding operation which produces a two level output sequence $b = [b_1, \ldots, b_N]$, for every input sequence a, according to the following condition:

$$b_j = \begin{cases} 0 & \text{for} \quad a_j < \frac{N}{2} \\ 1 & \text{for} \quad a_j > \frac{N}{2} \end{cases} \qquad \text{for j} = 1, \ldots, N$$

The choice of an appropriate threshold value is of great significance and depends on the practical purposes and the set of spreading sequences.

The representation and the analysis of the binary SS multiplexing can be described in a general form by applying the unified assessment approach developed in Chapter 4. First it should be noted that for a given N the number, M of possible combinations of data bits at the inputs of the multiplexer, is finite, i.e. $M = 2^N$. These combinations can be represented as vectors D_i $(i = 1, \ldots, M)$, which form the input symbol matrix D, defined as

$$D = \begin{bmatrix} D_1 \\ D_2 \\ \vdots \\ D_M \end{bmatrix} = \begin{bmatrix} d_{11} & d_{12} & \ldots & d_{1N} \\ d_{21} & d_{22} & \ldots & d_{2N} \\ & & \vdots & \\ d_{M1} & d_{M2} & \ldots & d_{MN} \end{bmatrix} \tag{7.8}$$

Each vector D_i is a different collection of simultaneous states of the input channels defined earlier as a row vector d. The matrix in (7.8) will also be referred to as the input word matrix and in the general case it contains all N-digit binary numbers.

Now the process of code multiplexing can be represented analytically by an expression determining a matrix A, which collects all possible multilevel sequences corresponding to the complete set of words, which represent the data inputs:

$$A = D \times C = \begin{bmatrix} \sum\limits_{j,i=1}^{N} d_{1j}c_{i1} & \sum\limits_{j,i=1}^{N} d_{1j}c_{i2} & \cdots & \sum\limits_{j,i=1}^{N} d_{1j}c_{iN} \\ \sum\limits_{j,i=1}^{N} d_{2j}c_{i1} & \sum\limits_{j,i=1}^{N} d_{2j}c_{i2} & \cdots & \sum\limits_{j,i=1}^{N} d_{2j}c_{iN} \\ \vdots & \vdots & \vdots & \vdots \\ \sum\limits_{j,i=1}^{N} d_{Mj}c_{i1} & \sum\limits_{j,i=1}^{N} d_{Mj}c_{i2} & \cdots & \sum\limits_{j,i=1}^{N} d_{Mj}c_{iN} \end{bmatrix} \tag{7.9}$$

The mathematical operations in (7.9) comply with the conventions adopted at the beginning of this section. The multiplication, in particular, is determined according to the chosen set of codes and the method of their generation.

The last step in compiling the analytical description of the BMC is the presentation of the thresholding function. In effect this function transforms integers from the range of 0 to N into the binary numbers 0 and 1 with respect to a suitably chosen parameter. The latter can be assumed equal to the maximal value of the multilevel signal, N in order to simplify the description of the thresholding function without restricting the generality of the method. In this case the code-multiplexed signal at the output of the system can be represented as a matrix B, which contains all possible binary sequences corresponding to the coded set of input blocks of symbols, and is given by

$$B = Int\left\{ \frac{A}{N} + 0.5 \right\} \tag{7.10a}$$

The function $Int\{*\}$ is defined as the integer part of the argument and is applied on an element-by-element basis as follows:

$$b_{mn} = Int\left\{ \frac{a_{mn}}{N} + 0.5 \right\} \tag{7.10b}$$

where
$$\left.\begin{array}{l} a_{mn} \in A \\ b_{mn} \in B \end{array}\right\} \quad \text{for } m = 1,\ldots,M \text{ and } n = 1,\ldots,N.$$

The expressions given by (7.9) and (7.10) define completely the binary sequence resulting from the suggested technique of multiplexing N information channels by combining the code sequences, which represent uniquely the binary symbols of each channel. By presenting the process in analytical form the multiplexed sequence can be determined for any combination of source signal values. It is possible to show that, if the code matrix C is a set of Gold sequences, the resulting set of BMC sequences B_m $(m = 1,\ldots,M)$ consists of all N-digit binary numbers. The performance of the proposed code-multiplexing operation has been verified numerically for several values of the code length, N. A number of SRG-s with $n = 3,4,5,6$ stages have been simulated analytically and most of the possible sets of Gold sequences have been generated in the form of matrices C_G. The N rows of each matrix are Gold codes produced by a pair of m-sequences of length $N = 2^n - 1$. The simulation assumes representing data symbols 1 by the sequences from C_G and data 0 by those from the inverted elements matrix $\overline{C}_G$. The BMC function in this case is given analytically by the following expression:

$$B = Int\left\{ \frac{D \times C_G + \overline{D} \times \overline{C}_G}{N} + 0.5 \right\} \tag{7.11}$$

where the elements of $\overline{D}$ are the inverted elements of the matrix D, which collects all possible N-digit binary numbers. The calculations, made for $N = 7, 15, 31$ show that the matrix B of the multiplexed sequences also consists of all N-digit binary numbers. This result confirms that the suggested transformation multiplexes unambiguously N information channels, with a symbol repetition period T_d, into a binary sequence with a symbol period $T_c = \frac{T_d}{N}$. The general proof requires involved mathematical transformations, which are not considered essential for the purpose of this discussion. The complexity of the BMC, compared to time-division multiplexing, is compensated for by achieving the flexibility to modify the frequency characteristics of the multiplexed signal. The possibilities to combine code multiplexing and spectrum shaping for the purposes of multichannel binary transmission have been investigated and some of the practical results are presented below. In this respect the objectives of the discussion are:

- to suggest the potential for increasing the efficiency in digital signal transmission by achieving the required spectrum modification by processing of large blocks of symbols, produced from the implementation of the proposed BMC technique;

- to demonstrate the ability of the uniform analysis procedure to perform accurate evaluation of the PSD function of block-coded sequences irrespective of the size of the symbol sets and the complexity of the code transformations.

7.3.3 Practical Implementation of the BMC Technique

The use of Gold sequences for a BMC combined with signal processing for spectrum modification has been simulated analytically. A large variety of schemes, implementing different codes, have been investigated. The spectrum-shaping effects, produced by certain 'Code' specifications, suggest the possibility to achieve virtually any PSD modification, mentioned in the previous chapters as being of practical interest. An example of a circuit, which implements the suggested BMC, is shown in Fig. 7.8. The coding scheme represented by the circuit uses Gold sequences of length N. The actual number of digital signals, which are multiplexed, is $N-1$. This limitation is imposed by the need to provide for frame alignment with respect to the code sequence cycle as the output blocks

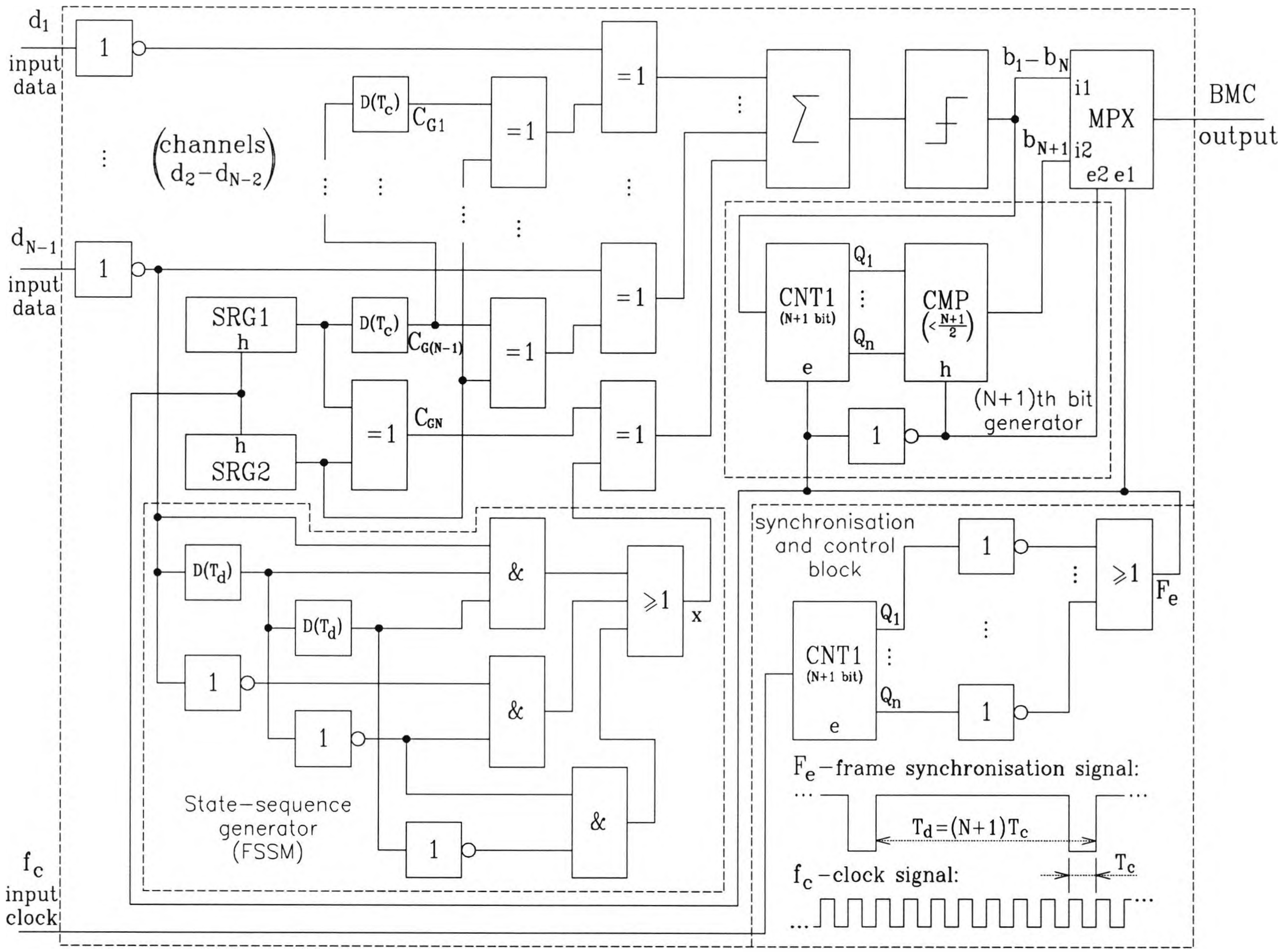


Fig. 7.8 BMC circuit for N-channel coding

of symbols can take all possible values of N-digit binary numbers. Additionally, the coder from Fig. 7.8 performs a block transformation of the type NB($N+1$)B, which provides for the possibility to balance the numbers of different code symbols in the output sequence.

The time relations in the coded signal are based on the data symbol period, T_d which is used as a reference. The clock cycle is taken to be the code symbol period, $T_c = \frac{T_d}{N+1}$. The Gold codes for the data channels d_1 - d_{N-1} are produced by modulo-2 adding different phase-shifts of the sequence generated by SRG1, provided through the delay elements D, to the output of SRG2. A symbol 1 from each data channel is coded as an N-bit Gold sequence and a symbol 0 from the same data signal, as the inverse of that sequence. This is achieved by modulo-2 adding the inverted data signal to the respective Gold sequences.

The Nth Gold sequence, produced by direct adding the outputs of SRG1 and SRG2, is used for frame synchronisation. Its initial and inverted forms are combined with the other sequences according to the code definition given in Table 7.1.

BMC - Code

D_m	s_1	s_2	s_3	s_4	S_m	Z_m
$D_{2\mu-1}$	$B^{(1)}_{2\mu-1}$, s_1	$B^{(0)}_{2\mu-1}$, s_3	$B^{(1)}_{2\mu-1}$, s_1	$B^{(0)}_{2\mu-1}$, s_3	$\begin{bmatrix} S_{2\mu-1} \\ S_{2\mu} \end{bmatrix}$	$\begin{bmatrix} Z_{2\mu-1} \\ Z_{2\mu} \end{bmatrix}$
$D_{2\mu}$	$B^{(1)}_{2\mu}$, s_2	$B^{(0)}_{2\mu}$, s_4	$B^{(0)}_{2\mu}$, s_2	$B^{(1)}_{2\mu}$, s_4		

Table 7.1

where

$$S_{2\mu-1} = \begin{bmatrix} 1&0&0&0 \\ 0&0&1&0 \\ 1&0&0&0 \\ 0&0&1&0 \end{bmatrix}, S_{2\mu} = \begin{bmatrix} 0&1&0&0 \\ 0&0&0&1 \\ 0&0&0&1 \\ 0&1&0&0 \end{bmatrix}, Z_{2\mu-1} = \begin{bmatrix} B^{(1)}_{2\mu-1} \\ B^{(0)}_{2\mu-1} \\ B^{(1)}_{2\mu-1} \\ B^{(0)}_{2\mu-1} \end{bmatrix}, Z_{2\mu} = \begin{bmatrix} B^{(1)}_{2\mu} \\ B^{(0)}_{2\mu} \\ B^{(0)}_{2\mu} \\ B^{(1)}_{2\mu} \end{bmatrix}$$

and $\mu = 1,\ldots,2^{N-1}$. The repetitive structure of the suggested code allows its specification to be presented in general form. The full notation for $N = 7$, for example, requires a table with 64 rows for the different input words, representing combinations of symbols from the different data channels. Every pair of rows of such a table, however, specifies the same coder rules and does not convey more information about the code than Table 7.1.

The coding scheme is implemented in the circuit from Fig. 7.8 by assuming that each of the possible input words, $D=[D_1,D_2,\ldots,D_M]$, where $M=2^{N-1}$, corresponds to the bits from the information channels being considered in the order $d_1,d_2,\ldots,d_{N-1}$. The symbols from channel d_{N-1} represent the parity of the input words, i.e. which of the two rows of Table 7.1 the current combination of bits at the data inputs corresponds to. The sole purpose of this assumption is to simplify the description. The performance of the proposed BMC technique is indiscriminate towards the order of the data symbols and the input words in the analytical presentation of the coding scheme. The result from applying the BMC is the same irrespective of which input blocks the synchronising Gold sequences, C_{GN} and $\overline{C}_{GN}$ are combined with, as long as their sequential order of use is determined by the code specification. In other words, any of the N data channels, d_i, $(i=1,\ldots,N-1)$ can be chosen to represent the parity of the input symbol blocks, half of which correspond to the $D_{2\mu-1}$ row of the BMC-'Code' table, for $d_i=0$ and the other half, to the $D_{2\mu}$ row of that table, for $d_i=1$. Formally, the output sequence for a particular input block is determined by the current coder state according to Table 7.1. In practice the two possible output words $B_\mu^{(0)}$ and $B_\mu^{(1)}$, for every input block D_μ, are generated automatically as the combination of symbols from the different channels uniquely determines the respective set of Gold sequences to be binary-multiplexed.

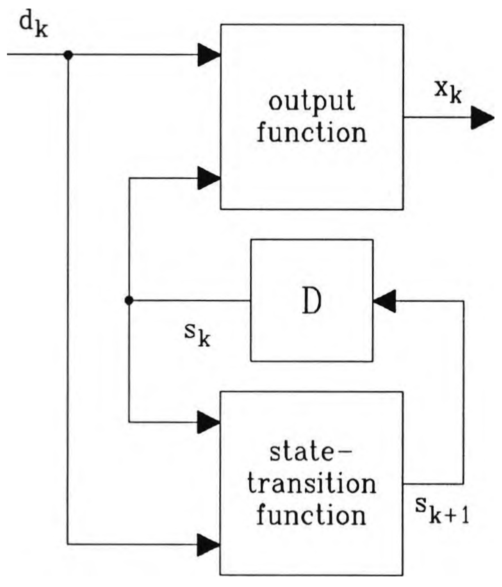


Fig. 7.9 The FSSM model of the BMC

The synchronising sequence, C_{GN} or $\overline{C}_{GN}$, combined with a particular set of Gold codes, is determined by the 'State-Sequence Generator' block of the circuit, which implements the FSSM function of the coder. The selection of the output word $B_\mu^{(x)}$, where $x=0$ or 1, is achieved with the binary sequence x_k, $(k=0\pm1,\pm2,\ldots)$ through the output function of the coder $x_k=\mathfrak{B}[d_k,s_k]$, which is modulo-2 added to the Nth Gold code, so that it switches between C_{GN} and $\overline{C}_{GN}$ according to the state-transition scheme defined by Table 7.1.

The logic circuit, generating the sequence x_k is derived from the general coder model which is represented by the diagram in Fig. 7.9. As it has been shown earlier, channel d_{N-1} is the only significant input for this particular code specification, therefore, the sequence $d_k\equiv d_{N-1}$ completely determines the input

$D_\mu(kT_d)$ of the coder (T_d is the input-symbol period).

The state-transition function $s_{k+1} = \mathcal{A}[d_k, s_k]$ can be synthesised from its specification given in table form as follows. The states s_i, $(i = 1,2,3,4)$ can be represented as the two-digit binary number $(s_{ia}s_{ib}) \in s = [11,\ 01,\ 10,\ 11]$. This allows the relations between d_k, x_k and s_k to be determined from the correspondence between Tables 7.2a and b.

D_m	s_i	$B_m^{(x)}$	s_i
$D_{2\mu-1}$	s_1	$B_{2\mu-1}^{(1)}$	s_1
$D_{2\mu-1}$	s_2	$B_{2\mu-1}^{(0)}$	s_3
$D_{2\mu-1}$	s_3	$B_{2\mu-1}^{(1)}$	s_1
$D_{2\mu-1}$	s_4	$B_{2\mu-1}^{(0)}$	s_3
$D_{2\mu}$	s_1	$B_{2\mu}^{(1)}$	s_2
$D_{2\mu}$	s_2	$B_{2\mu}^{(0)}$	s_4
$D_{2\mu}$	s_3	$B_{2\mu}^{(0)}$	s_2
$D_{2\mu}$	s_4	$B_{2\mu}^{(1)}$	s_4
input	present	output	next

Table 7.2a State-transition scheme

d_k	s_k		x_k	s_{k+1}	
0	0	0	1	0	0
0	0	1	0	1	0
0	1	0	1	0	0
0	1	1	0	1	0
1	0	0	1	0	1
1	0	1	0	1	1
1	1	0	0	0	1
1	1	1	1	1	1
input	s_{ia}	s_{ib}	output	s_{ia}	s_{ib}

Table 7.2b Logical representation of the states

The basic functions, implementing the BMC relations according to the FSSM model from Fig. 7.9, are defined from the following expressions:

$$x_k = d_k(s_{ia})_k(s_{ib})_k + \overline{d}_k\overline{(s_{ib})}_k + \overline{(s_{ia})}_k\overline{(s_{ib})}_k \tag{7.13a}$$

$$(s_{ib})_{k+1} = d_k, \quad (s_{ia})_{k+1} = (s_{ib})_k \tag{7.13b}$$

where the multiplication and the summation are the logical AND and OR operations, respectively. Expressions (7.12) allow the coder functions to be implemented as the circuit given in Fig 7.10. This circuit is incorporated in the complete coder diagram from Fig. 7.8 as the 'State-Sequence Generator' block, where the input d_k is taken from channel d_{N-1}.

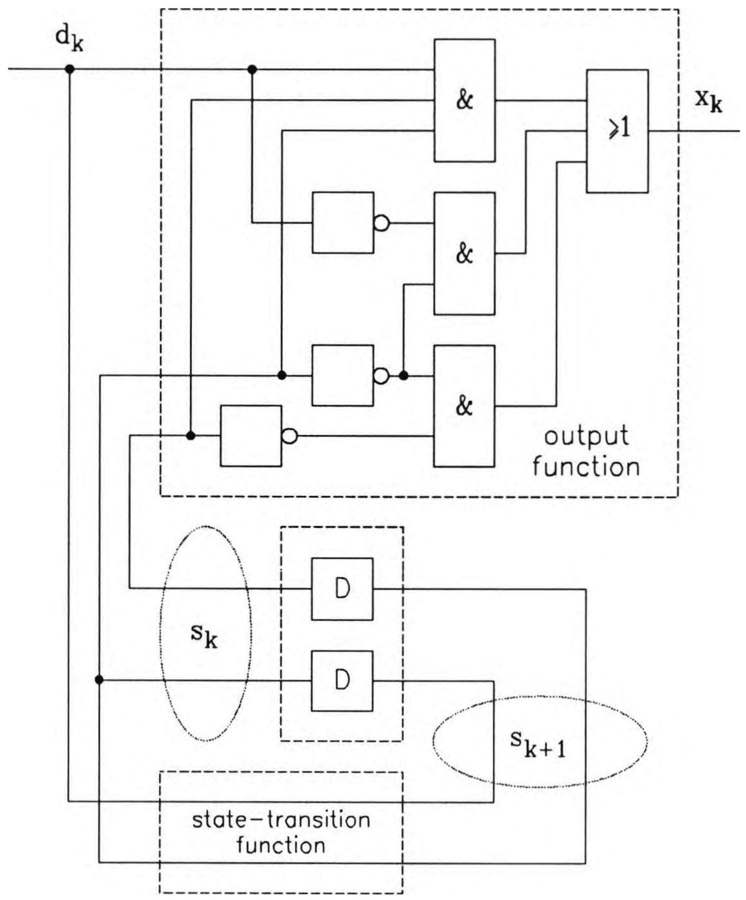


Fig. 7.10 **The circuit implementing the BMC coder functions**

The block structure of the N multiplexed Gold codes is enhanced with an additional $(N+1)$th bit, b_{N+1} which is generated by the counter CNT1 and the digital comparator CMP, according to the following conditions:

$$b_{N+1} = \begin{cases} 1, & \text{if } \sum_{\nu=1}^{N} b_\nu < \frac{N+1}{2} \\ 0, & \text{if } \sum_{\nu=1}^{N} b_\nu \geq \frac{N+1}{2} \end{cases} \tag{7.14}$$

The frame structure is completed by appending b_{N+1} at the end of each coded sequence b_1 - b_N through the multiplexer MPX.

The timing control of the suggested coder circuit is provided by the 'Synchronisation and Control Block'. The input clock sequence f_c is converted into the signal F_e, as shown in Fig. 7.11, through the simple logic circuit transforming the binary output of the counter CNT2.

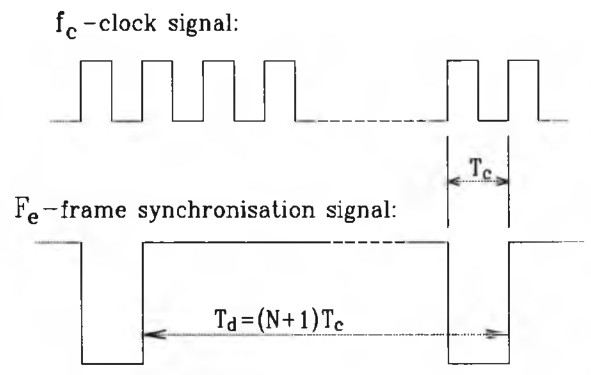


Fig. 7.11 The F_e frame-control signal

The connections of f_c to the SRG-s and the other components, whose operation requires a clock signal, are not shown in the diagram of Fig. 7.8 as its main purpose is to illustrate the principle of the proposed BMC technique. For the same reason only the functional use of the frame-control signal F_e is indicated, without considering the technical specifications of any particular components.

Thus, the high level of the F_e waveform enables:

- the shifting function of the SRG circuits;
- the counting of the 1-s in the first N bits of the code frame by CNT1;
- the input $i1$ of the MPX which outputs symbols b_1-b_N of the BMC sequence.

The low level of F_e is used for the duration of the last code symbol in each frame:

- to suspend the shifting in SRG1 and SRG2;
- to hold the result at the output of CNT1 and comparator CMP;
- to enable the second input, $i2$ of the multiplexer which outputs the symbol b_{N+1} of the BMC frame.

The timing control in the complete practical circuit is implemented in a more precise form, by considering the specific enable and hold functions of particular components, as well as the possibility to use edge triggering.

The coder circuit in Fig. 7.8 is a simple implementation of the proposed BMC structures, which combine the multiplexing of digital sequences into a binary transmission signal with coding for shaping the spectrum of that signal. The adopted four-state structure utilises only a fraction of the flexibility provided by the potentially large size of the code-word set. The spectral analysis of the resulting BMC signal is based on the repetitive pattern of the state-transition matrices, which can be determined from Table 7.1 for every $\mu = 1,\ldots,2^{N-1}$. The respective output matrices are defined through the same table and the expressions (7.9) and (7.10). The analytical form of these matrices is amended in the computational routine with the additional $(N+1)$th bit, for each block of output symbols as follows

$$Z_m = \left[b_{m1}^{(x_i)}, b_{m2}^{(x_i)}, \ldots, b_{mN}^{(x_i)}, b_{m(N+1)}^{(x_i)}\right]$$

where $m = 2\mu - 1$ or 2μ, $x_i = 0$ or 1 for $i = 1, \ldots, 4$ and

$$b_{m(N+1)}^{(x_i)} = \text{Int}\left\{\frac{1}{N}\sum_{\nu=1}^{N} b_{m\nu}^{(x_i)} + 0.5\right\} \tag{7.15}$$

The Int$\{*\}$ function is the same as defined earlier for (7.10). The expression (7.14) is a generalisation of conditions (7.13) and allows the $(N+1)$th symbol of each code word to be determined analytically.

The complete sets, SM and ZM of the state-transition and the output matrices, used in the PSD evaluation procedure have been computed for $N = 7$. This has allowed the spectrum of the binary code-multiplexed signal to be determined as shown in Fig. 7.12a.

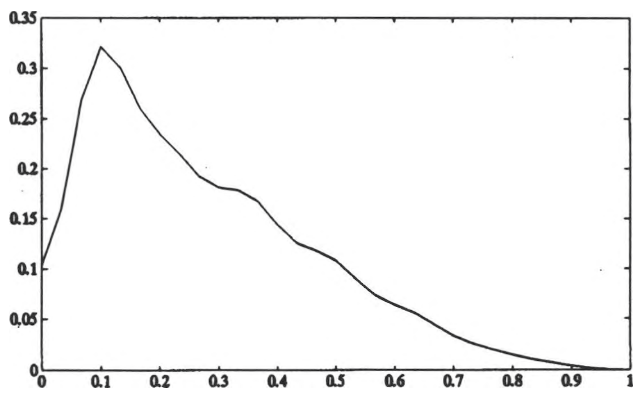


Fig. 7.12a PSD of the BMC signal

The PSD plot reveals a noticeable suppression of the low-frequency components and some bandwidth reduction, which indicate the potential for achieving spectra useful in digital transmission. In addition, the parametrical plot in Fig. 7.12b shows the stability of the frequency characteristics, which remain virtually unaffected by variations of the input symbol probabilities.

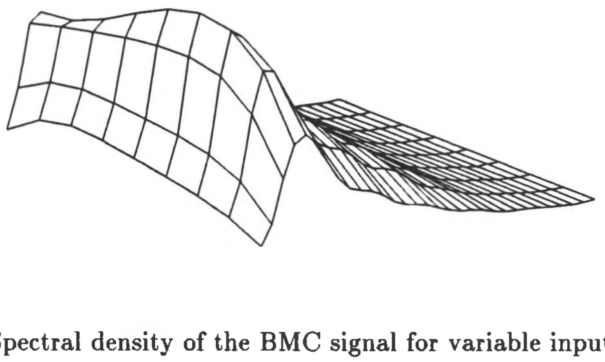


Fig. 7.13b Spectral density of the BMC signal for variable input probabilities

The BMC scheme discussed above is of high a complexity level. It belongs to the $D(2^{N-1})S4X$ (2^N) category ($D64S4X128$, for $N = 7$). This suggests the possibility to create a variety of BMC structures with PSD functions modified to meet most of the requirements in practical communication systems. The computational algorithm, used to generate the analysis results in the thesis, has been further developed into a powerful simulation tool, which allows a systematic and exhaustive search for suitable coding rules to be performed. Various combinations of state-transition schemes and output-word structures have been experimented with. A detailed discussion of the experiments is not considered to be relevant to this report, as particular results are only for specific applications. For the purposes of this presentation it is sufficient to outline the most significant possibilities provided through the unified implementation of the spectral analysis procedure and the proposed method for binary multiplexed coding.

A remarkable feature of the coder from Fig. 7.8 is the intrinsic generality of its structure. There is no limitation in principle for the diagram to represent the same coding scheme with any number of information channels, N. Larger values of N do not require increase in the coder complexity. The main restrictions are imposed by the delay between the beginning of the first and the last phase-shifts of one of the m-sequences, accumulated through a large number of the delay components, $D(T_c)$. This, however, is a purely technical problem which can be solved by a suitable arrangement of more than two SRG-s.

The state-sequence generating block is designed independently of the remaining circuit and its complexity is determined only by the adopted number of states and the specification of the coding rules. Many of the codes discussed in the thesis suggest that the number of states does not have to be very large[8], in order to achieve sufficient spectrum-shaping flexibility.

A significant advantage of the BMC schemes over the conventional block-coding structures is the generation of the code words through shift-register sequences combined with the input symbols. This eliminates the use of large look-up tables with the respective data storage/access requirements and allows the new code to be implemented at very high transmission rates.

In summary

1) The proposed BMC combines the use of SS techniques for the multiplexing of digital sequences into a binary transmission signal with coding for spectrum modification.

[8] Four states give satisfactory results for most practical purposes.

ii) The analytical presentation of the BMC rules (expressions (7.9) - (7.14)) provides for the possibility to evaluate the set of code blocks regardless of its size.

iii) The 'Code' table is specified in general form and the respective matrix presentation of the BMC can be determined for any length, N of the code words.

iv) The result from ii) and iii) allow the unified spectral analysis approach and the developed computational algorithm to be implemented with no restriction on the size of the sets of input words, output words and states.

The main concepts of the SS communications have been briefly discussed in this chapter, in order to introduce the suggested new technique of code-multiplexing for binary multichannel transmission. The integration of the multiplexing and the line coding processes has been demonstrated through the design, the simulation and the analysis of the BMC and the circuit for its implementation. The analytical presentation of the technique in general form suggests the potential for further development and investigation of the method for spectrum modification of binary code-multiplexed signals.

8. SUMMARY AND CONCLUSIONS

The main objectives of this work the creation of a universal algorithm for the analysis and comparative assessment of digital line codes has been achieved in three stages. The first phase was concerned with an extensive survey of the techniques and the related problems in the design and analysis of codes for digital signal transmission. A large number of publications have been studied in order to identify the common issues, as well as the specific achievements in satisfying the stringent requirements for efficient and reliable digital communications. In order to restrict the literature search it was decided to specify certain goals as follows:

- Determine the main areas of and reasons for the implementation of coding in the overall structure of modern digital communication systems.
- Define the importance of line coding with respect to source and error-control coding.
- Compile a comprehensive overview of line coding fundamentals, allowing the basis for subsequent investigations to be established

The outcome of this work is given in the first chapter. It contains an overall presentation of the concepts of digital transformations used to improve the efficiency of communication and to reduce the degrading effects of the transmission channel. The interpretation of the major factors affecting the choice of coding schemes and the description of basic source and error-control codes has revealed the common aspects and the fundamental limitations of coding imposed by the Theory of Information. Finally, the comparative investigation of the problems addressed through the different types of coding has allowed the following conclusion to be reached.

The main objectives of coding, through the adaptation of digital signals to match the transmission media, is to improve the efficiency and reliability of communication. In this respect the transformation of digital sequences into a signal with particular spectral characteristics can be considered an integral part of the channel coding process. Furthermore, line coding is entirely based on the same fundamental principle as coding in general, which requires the introduction of information redundancy, to minimise the distortion of the original signal caused by the restrictions of the transmission channel.

The investigation of the specific effects of line coding on the characteristics of digital signals is presented in Chapter 2. The restrictions on line coding, imposed by line signal considerations, are also investigated. The significant parameters of different types of codes have been outlined from a detailed survey of a considerable number of popular coding techniques. In addition some existing code classification schemes have been studied in order to identify the essential considerations for the suggested general assessment and analysis approach.

The second part of Chapter 2 presents a more detailed investigation of the specific features of typical line codes. The major achievement from this initial analysis of various code structures has been the introduction of common assessment and evaluation criteria, as well as an extensive basis for the subsequent development of the uniform spectral analysis approach. Many popular coding techniques have been investigated and their conventional specifications represented in terms of symbol transformations. Typical spectral characteristics, produced by various codes, have been systematically identified and related to particular code features. The most significant result from the preliminary implementation of uniform assessment criteria is the possibility to specify the general purpose of line coding as follows:

> Modification of the spectral features of the coded signal by changing the statistics of the original signal through structural transformation of input blocks of symbols into output blocks of symbols.

The second stage in the research work comprised the development of the theoretical basis for the unified spectral analysis. The adopted analytical model, described in Chapter 3, is based on the presentation of the line coder through the functional relations of a finite-state sequential machine. The fundamental concepts of the adopted theory involve the computation of the spectral density of the coded signal by applying the standard Fourier transformation. The latter, however, requires the use of second order probabilities in order to express the statistics of the output signal through the statistics of the input sequence, which are related by the non-linear function of the coder. The presentation of the theoretical model of the coder is given in its complete form, redeveloped for the specific case of a discrete sequence of symbols, while preserving the full accuracy and elegancy of the original version, [18].

The major accomplishment from the use of this model is the possibility to specify completely any block coding scheme by the sets of input words, output words and coder states, which are uniquely related through the table and matrix presentation of the coder rules. This allows the continuous and the discrete spectral densities of the coded signal to be computed by specifying the state-transition matrices, the output matrices and the input-symbol probabilities.

Another important achievement is the direct implementation of the spectral analysis formulae into a computational routine, which is described in the second part of Chapter 3. The algorithm for evaluation of the power spectral density has been created within the environment of a high-performance interactive software package, which allows the accuracy and the flexibility of the analysis model to be fully utilised. The computational procedure has been optimised and developed to accommodate various analysis and simulation requirements. A high degree of flexibility and universality has been achieved through the use of a modular structure which allows the spectral analysis technique to be applied to virtually all digital coding schemes with a fixed-length block structure. The results are presented in graphical form and provide a common basis for estimation and comparison of codes and their frequency characteristics.

In Chapter 4 the uniform spectral analysis approach, based on the theoretical model of the coder, is applied to the coding schemes assessed in Chapter 2. The results demonstrate the viability of using a general and systematic approach to the problems of the design and the analysis of line codes. It permits a direct comparative evaluation of various coding schemes, not possible with the frequently used application specific form of definition. It also offers the potential for further theoretical investigation.

Additionally, several new codes are analysed, which show how the structure of the code may be deliberately selected or manipulated to provide desirable features in the line signal spectrum.

A standard method is proposed to define any fixed-length, finite-memory block code in the form of a code specification table, necessary for the uniform spectral analysis routine.

Chapter 5 takes the computational algorithm, derived in the previous chapter, and extends it to a completely general routine for the design of digital line codes. It is based on a further enhancement of the computational algorithm, used to analyse the frequency characteristics of coded sequences. This is accomplished through a generalised specification of the initial parameters (the sets of input/output symbols and coder states) and provides spectral analysis of all the coding schemes which are determined as valid according to predefined conditions. The method allows for the systematic generation and assessment of a large variety of code structures, based on the general definition of code categories. The modified computational algorithm is described and the results from the spectral analysis of several groups of codes are presented. These results are summarised as the basis for a method of creating new coding schemes.

The results shown in Chapter 5 illustrate the fundamental concepts of the suggested uniform assessment and analysis approach. Most of the important stages in the generation and classification of new code structures have been demonstrated through the systematic investigation of some low-complexity level categories. Many of the popular coding techniques have been shown to appear precisely in the predicted categories, exhibiting the expected structural and spectral features typical of the respective group of schemes. These results prove the general applicability of the suggested classification method.

The investigation of complete categories is another significant achievement. All codes from $D2S1X4$, $D2S1X8$, $D4S1X4$ and $D2S2X2$ have been generated and analysed. This in effect precludes the possibility to 'invent' a new coding scheme with parameters specified within these categories. Some higher complexity level categories have also been exhaustively simulated and assessed, but their complete investigation will be presented in a future extension of this work.

The ability of different codes to achieve particular spectral features has been determined. Of equal importance is the efficiency of a coding structure in terms of its information capacity. A new method for the evaluation of coding efficiency is suggested, which derives directly from the uniform analysis approach. It is based on the system of general classification of codes, developed in Chapter 5, through a 3D volume representation of the coding capacity. Using this method, it is demonstrated how an optimal combination of a code structure, spectral features and information capacity may be achieved.

The thesis concludes with a practical example of the application of the generalised analysis procedure. The example chosen was a particularly demanding one, involving a spread spectrum communication channel. The use of a Binary-Multiplexed Code (BMC) is investigated and evaluated. The analysis suggests some significant advantages may be derived from the BMC approach and a simple circuit is proposed for its implementation.

In conclusion one of the most useful results, accomplished with the development of the spectral analysis procedure, is the possibility to apply the coder model and the software routine to any finite state symbol transformation scheme of theoretically unlimited complexity, as long as it is appropriately specified[1]. Despite the relatively sophisticated probability and FSSM theories involved in deriving the final expressions for the PSD functions, their practical

[1] According to the requirements of the FSSM model of the coder, presented in Chapter 3.

implementation is simple and straightforward[2]. This is demonstrated in the thesis by a comparison of the computational algorithm with some of the conventional techniques for spectral analysis of coded signals. A typical example is given in [2], where the spectral density function of the AMI code is evaluated through a lengthy and tedious procedure, involving manual calculation of probabilities for numerous combinations of coded symbols. This method has an acceptable degree of accuracy but it becomes impractical for more complex codes. In many publications concerning frequency analysis of coded signals very little, if anything, is mentioned about the techniques which had been used to evaluate the spectral density. Often the graphical results are only approximations and some of the spectral characteristics are deduced empirically and by analogies.

The frequency analysis algorithm and the uniform assessment method suggested in this work provide a powerful tool for investigation and design of virtually any fixed-length block-coding scheme of practical interest. The computational procedure can produce an exhaustive set of results corresponding to a variety of initial conditions, as well as any amount of detail related to specific frequency characteristics of coded signals. There is considerable scope for further work. The implementation of well defined analytical selection criteria is required as a basis for further work on the general classification method and the use of the suggested information capacity evaluation technique. The generality and the comparison capabilities of the technique offer enormous potential for the development of a whole range of new coding schemes with the possibility of matching them efficiently not only to the spectral characteristics of a particular communication channel, but also to its information capacity.

[2] Depending on the available computing resources and software environment.

APPENDIX - A

This appendix contains the code of the computational algorithm, which is written in the application-specific language of the Matlab package, [15]. This is a command based interactive programming environment with a comprehensive set of instructions performing powerful mathematical operations on matrices and allowing for suitable graphical presentation of the results. The software routine is developed in a modular form, which makes it compact, flexible and easily adaptable to various analysis and simulation requirements.

All modules can be divided into several types on the basis of functional similarity. The first type represents the main blocks for the different codes or groups of codes. These modules are used to specify the code symbol sets, the matrix presentation of the coding rules, the ranges of the computational parameters and the subroutines, required for the main analysis procedure. The structure of the main blocks is similar for all codes and will be described in detail only for one of the schemes with some additional comments given for the others where more substantial differences are encountered. The computational blocks are the second type of modules. They implement the analytical expressions developed in Chapter 3 and are common for all coding schemes. The modules processing the analysis results and representing them in the required output format comprise the third type of program blocks.

The complete set of programming modules is presented in two parts. The first part contains all the main blocks of code and the subroutines used in the evaluation of the spectral density functions of the conventional coding techniques, discussed in Chapter 4. The second part consists of the enhanced computational structures which have been used to simulate and analyse the full range of coding schemes, according to the classification categories, suggested in Chapter 5.

The adopted notation is based on the Matlab programming conventions. It is given below in a different font to make it distinguishable from the comments. Explanatory notes are given before or after the command lines of the program modules, except where a reference to a specific line is necessary. The comments, which appear within the body of a software block, are delimited by a '%' symbol and the beginning of a new line. All the modules of the software routine are used as Matlab ∗.M files. The numerical and the graphical results are stored as ∗.MAT files and ∗.MET data files, respectively. Finally, it should be noted that the program blocks are stripped from the lines providing for the interactive mode of operation, to avoid confusion of the parts comprising the main computational procedure with unessential interface and presentation lines of software code.

Part I - Modules and Subroutines for the Analysis of the Conventional Codes

The main software blocks of the conventional coding schemes are given in this section. The first group of lines[1] specify the symbol and the word sets as matrices, from which the sizes of these sets are determined. The state-transition and the output matrices, as well as their sizes are determined by the second group of lines. The two lines of the third group are used to specify the ranges of the computational parameters (as described in Chapters 3 and 4). The fourth group determines the range of the input probabilities (if variable) and performs the complete analysis for each value of the probability parameter. The main mathematical computations are executed in this part of the module, which calls a number of subroutines. The graphical display of the intermediate and the final analysis results is also produced by this part, except for the 3D parametrical plot which appears after the full parametrical cycle is completed.

Main modules of the spectral analysis procedure:

```
% Computes the PSD of the NRZ scheme for unipolar input symbols

d=[0,1];   D=[d(1);d(2)];    [M,N]=size(D);          % the input symbol and word sets
x=[0,1];   X=[x(1); x(2)];  [J,L]=size(X);          % the output symbol and word sets

S1=[1]; S2=[1];                 % the state-transition matrices (this is a one-state scheme)
Z1=[X(1,:)]; Z2=[X(2,:)];                                            % the output matrices
[I,I]=size(S1); SM=[S1;S2]; ZM=[Z1;Z2];

j=sqrt(-1);  Fx=1; nFx=1;  nf=50;  npr=1;                % the computational parameters
Tx=1/Fx;  T=L*Tx; Td=(L/N)*Tx;  F=Fx/L; Fd=(N/L)*Fx;

for p = 1 : npr                                       % starts the variable probabilities cycle
  q0=p/(npr+1); q1=1-q0;                                  % the input symbol probabilities
  GPSD01;   GPSD02; axis([-.01,1,0,.3]);  PSDPLOT;                    % the subroutines
end

if npr>1
  mesh(yc)                                          % displays the parametrical PSD plot
else
end
```

[1] A number of lines separated from the rest of the code by a blank line.

```
% Computes the PSD of the NRZ scheme for polar input symbols

d=[0,1];    D=[d(1);d(2)];      [M,N]=size(D);
x=[-1,1];   X=[x(1); x(2)];   [J,L]=size(X);

S1=[1]; S2=[1];
Z1=[X(1,:)]; Z2=[X(2,:)];
[I,I]=size(S1); SM=[S1;S2]; ZM=[Z1;Z2];

j=sqrt(-1);  Fx=1; nFx=1;   nf=50; npr=9;
Tx=1/Fx;  T=L*Tx; Td=(L/N)*Tx;  F=Fx/L; Fd=(N/L)*Fx;

for p = 1 : npr
   q0=p/(npr+1); q1=1-q0;
   GPSD01;    GPSD02; axis([-.01,1,0,1.1]);   PSDPLOT;
   title('POLAR'); text(.4,.7,'yc'); text(.4,1.03,'Yxc');
   text(.05,.15','Yxd=0');  text(.1,.1,'yd=0');
end

if npr>1
  mesh(yc)
else
end
```

```
% Computes the PSD of the Manchester code

d=[0,1];   D=[d(1);d(2)];                    [M,N]=size(D);
x=[0,1];  X=[x(1), x(2); x(2), x(1)];    [J,L]=size(X);

S1=[1];  S2=[1];
Z1=[X(1,:)]; Z2=[X(2,:)];
[I,I]=size(S1); SM=[S1;S2]; ZM=[Z1;Z2];

j=sqrt(-1);   Fx=1; nFx=1;   nf=30;  npr=1;
Tx=1/Fx;  T=L*Tx; Td=(L/N)*Tx;  F=Fx/L; Fd=(N/L)*Fx;

for p = 1 : npr
    q0=p/(npr+1); q1=1-q0;
    GPSD01;    GPSD02;    PSDPLOT;    title('MANCHESTER');
   text(.5,1,'yc'); text(.7,1.4,'Yxc');
   text(.01,1,'Yxd=0');  text(.01,.5,'yd=0');
end

if npr>1
   mesh(yc);  title('MANCHESTER');
else
end
```

```
% Computes the PSD of the Differential code

d=[0,1];     D=[d(1);d(2)];    [M,N]=size(D);
x=[-1,+1]; X=[x(1); x(2)];  [J,L]=size(X);

S1=[1 0                        % the state-transition matrices (this is a two-state scheme)
     0 1]; S2=[0 1
                 1 0];  Z1=[X(1,:)
                            X(2,:)]; Z2=[X(2,:)
                                         X(1,:)];
[I,I]=size(S1); SM=[S1;S2]; ZM=[Z1;Z2];

j=sqrt(-1);   Fx=1; nFx=1;   nf=50; npr=1;
Tx=1/Fx;  T=L*Tx; Td=(L/N)*Tx;  F=Fx/L; Fd=(N/L)*Fx;

for p = 1 : npr
   q0=p/(npr+1); q1=1-q0;
   GPSD01;    GPSD02;  axis([-.01,1,0,1.1]);
   PSDPLOT;  title('DIFFERENTIAL (NRZ-mark/space)');
   text(.4,.7,'yc');      text(.4,1.1,'Yxc');
   text(.05,.15,'Yxd=0'); text(.1,.1,'yd=0');
end
if npr>1
  mesh(yc)
else
end
```

```
% Computes the PSD of the AMI code

d=[0,1];    D=[d(1);d(2)];                [M,N]=size(D);
x=[-1,0,1]; X=[x(1); x(2); x(3)];   [J,L]=size(X);
S1=[1 0
      0 1]; S2=[0 1
                     1 0];   Z1=[X(2,:)
                                    X(2,:)]; Z2=[X(1,:)
                                                    X(3,:)];
[I,I]=size(S1); SM=[S1;S2]; ZM=[Z1;Z2];

j=sqrt(-1);  Fx=1; nFx=1;   nf=50;  npr=9;
Tx=1/Fx;  T=L*Tx; Td=(L/N)*Tx;  F=Fx/L; Fd=(N/L)*Fx;

for p = 1 : npr
   q0=p/(npr+1); q1=1-q0;
   GPSD01;    GPSD02;    PSDPLOT;
end

if npr>1
  mesh(yc);  title('AMI');
else
end
```

```
% Computes the PSD of the Duobinary code

d=[0,1];          D=[d(1); d(2)];          [M,N]=size(D);
x=[-1,0,+1];    X=[x(1); x(2); x(3)];    [J,L]=size(X);
S1=[0 1
      0 1]; S2=[1 0
                   1 0];  Z1=[X(2,:)
                              X(1,:)]; Z2=[X(3,:)
                                           X(2,:)];
[I,I]=size(S1); SM=[S1;S2]; ZM=[Z1;Z2];

j=sqrt(-1);  Fx=1; nFx=1;  nf=50;  npr=1;
Tx=1/Fx;  T=L*Tx; Td=(L/N)*Tx;  F=Fx/L; Fd=(N/L)*Fx;

for p = 1 : npr
   q0=p/(npr+1); q1=1-q0;
   GPSD01;   GPSD02;   PSDPLOT;
   title('DUOBINARY');
   text(.1,.7,'yc');     text(.3,.4,'Yxc');
   text(.05,.5,'Yxd=0');  text(.05,.3,'yd=0');
end

if npr>1
  mesh(yc);  title('DUOBINARY');
else
end
```

```
% Computes the PSD of the CMI code

d=[0,1];     D=[d(1);d(2)];                                   [M,N]=size(D);
x=[-1,+1]; X=[x(1), x(2); x(2), x(2); x(1), x(1)];   [J,L]=size(X);

S1=[1 0
     0 1]; S2=[0 1
                  1 0];  Z1=[X(1,:)
                                 X(1,:)]; Z2=[X(2,:)
                                                  X(3,:)];
[I,I]=size(S1); SM=[S1;S2]; ZM=[Z1;Z2];

j=sqrt(-1);  Fx=1; nFx=1;   nf=50;  npr=1;
Tx=1/Fx;  T=L*Tx; Td=(L/N)*Tx;  F=Fx/L; Fd=(N/L)*Fx;

for p = 1 : npr
   q0=p/(npr+1); q1=1-q0;
   GPSD01;    GPSD02;    PSDPLOT;    title('CMI');
   xy=.4; xY=xy;  % this and the next four lines place text at a computed position in the plot
   fiy=round((xy*nf)/(nFx*Fx)); yy=yc(p,fiy+1)+.05;
   fiY=round((xY*nf)/(nFx*Fx)); yY=Yxc(p,fiy+1)+.05;
   text(xy,yy,'yc');    text(xY,yY,'Yxc');
   text(.53,.25,'Yxd'); text(.53,.05,'yd');
end

if npr>1
  mesh(yc);
else
end
```

% Computes the PSD of the conventional 3B4B code
% A new Matlab, function *allbin(n)* has been created to compute
% all possible n-digit binary numbers, from which the input and the output
% word sets are formed.

```
d=[0,1];  D=allbin(3);    [M,N]=size(D);
x=[0,1];  X=allbin(N+1);  [J,L]=size(X);

S1=[1 0
    0 1]; S2=[0 1
              1 0];
Z1=[X(6,:)                % the output matrices are specified according to a look-up table, [18]
    X(6,:)]; Z2=[X(10,:)
                 X(10,:)]; Z3=[X(15,:)
                               X(5,:) ]; Z4=[X(14,:)
                                             X(9,:) ];
Z5=[X(8,:)
    X(3,:)]; Z6=[X(12,:)
                 X(2,:) ]; Z7=[X(7,:)
                               X(7,:)]; Z8=[X(11,:)
                                            X(11,:)];
[I,I]=size(S1); SM=[S1;S1;S2;S2;S2;S2;S1;S1];
              ZM=[Z1;Z2;Z3;Z4;Z5;Z6;Z7;Z8];

j=sqrt(-1); Fx=1; nFx=1;  nf=30; npr=9;
Tx=1/Fx;  T=L*Tx; Td=(L/N)*Tx;  F=Fx/L; Fd=(N/L)*Fx;

for p = 1 : npr
  q0=p/(npr+1); q1=1-q0;
  GPSD01;   GPSD02;   PSDPLOT;   title('3B4B');
  xy=.5; xY=.7;
  fiy=round((xy*nf)/(nFx*Fx)); yy=yc(p,fiy+1)+.01;
  fiY=round((xY*nf)/(nFx*Fx)); yY=Yxc(p,fiy+1)+.01;
  text(xy,yy,'yc'); text(xY,yY,'Yxc'); text(.05,.2,'Yxd');  text(.06,.25,'yd');
end

if npr>1
  mesh(yc);  title('3B4B');
else
end
```

```
% Computes the PSD of the 5B6B code (the look-up table is taken from [24])

d=[0,1];   D=allbin(5);      [M,N]=size(D);
x=[0,1];   X=allbin(N+1);   [J,L]=size(X);


S1=[1 0
      0 1]; S2=[0 1
                  1 0];
 Z1=[X(23,:); X(40,:)]; Z2=[X(39,:); X(24,:)]; Z3=[X(27,:); X(36,:)];
 Z4=[X( 7,:); X( 7,:)]; Z5=[X(43,:); X(20,:)]; Z6=[X(11,:); X(11,:)];
 Z7=[X(13,:); X(13,:)]; Z8=[X(14,:); X(14,:)]; Z9=[X(51,:); X(12,:)];
Z10=[X(19,:); X(19,:)];Z11=[X(21,:); X(21,:)];Z12=[X(22,:); X(22,:)];
Z13=[X(25,:); X(25,:)];Z14=[X(26,:); X(26,:)];Z15=[X(28,:); X(28,:)];
Z16=[X(45,:); X(18,:)];Z17=[X(29,:); X(34,:)];Z18=[X(35,:); X(35,:)];
Z19=[X(37,:); X(37,:)];Z20=[X(38,:); X(38,:)];Z21=[X(41,:); X(41,:)];
Z22=[X(42,:); X(42,:)];Z23=[X(44,:); X(44,:)];Z24=[X(53,:); X(10,:)];
Z25=[X(49,:); X(49,:)];Z26=[X(50,:); X(50,:)];Z27=[X(52,:); X(52,:)];
Z28=[X(51,:); X( 6,:)];Z29=[X(56,:); X(56,:)];Z30=[X(46,:); X(17,:)];
Z31=[X(54,:); X( 9,:)];Z32=[X(58,:); X( 5,:)];
[I,I]=size(S1);
SM=[S2;S2;S2;S1;S2;S1;S1;S1;S2;S1;S1;S1;S1;S1;S1;S2;
     S2;S1;S1;S1;S1;S1;S1;S2;S1;S1;S1;S2;S1;S2;S2;S2];
ZM=[Z1;Z2;Z3;Z4;Z5;Z6;Z7;Z8;Z9;Z10;Z11;Z12;Z13;Z14;Z15;Z16;Z17;
     Z18;Z19;Z20;Z21;Z22;Z23;Z24;Z25;Z26;Z27;Z28;Z29;Z30;Z31;Z32];


j=sqrt(-1); Fx=1; nFx=1;   nf=50; npr=1;
Tx=1/Fx;  T=L*Tx; Td=(L/N)*Tx;  F=Fx/L; Fd=(N/L)*Fx;


for p = 1 : npr
   q0=p/(npr+1); q1=1-q0;
   GPSD01;   GPSD02;  PSDPLOT;   title('5B6B');
   xy=.5; xY=.7;
   fiy=round((xy*nf)/(nFx*Fx)); yy=yc(p,fiy+1)+.01;
   fiY=round((xY*nf)/(nFx*Fx)); yY=Yxc(p,fiy+1)+.01;
   text(xy,yy,'yc');    text(xY,yY,'Yxc');
   text(.05,.2,'Yxd');  text(.06,.25,'yd');


end
if npr>1
   mesh(yc); title('5B6B');
else
end
```

```
% Computes the PSD of the Modified Duobinary code

d=[0,1];       D=[d(1);d(2)];          [M,N]=size(D);
x=[-1,0,1];  X=[x(1); x(2); x(3)];  [J,L]=size(X);

S1=[ 1 0 0 0                        % the state-transition matrices (this is a four-state scheme)
     0 0 1 0
     1 0 0 0
     0 0 1 0]; S2=[  0 1 0 0
                     0 0 0 1
                     0 1 0 0
                     0 0 0 1]; Z1=[  X(2,:)
                                     X(2,:)
                                     X(1,:)
                                     X(1,:)]; Z2=[     X(3,:)
                                                       X(3,:)
                                                       X(2,:)
                                                       X(2,:)];
[I,I]=size(S1); SM=[S1;S2]; ZM=[Z1;Z2];

j=sqrt(-1);  Fx=1; nFx=1;   nf=50;  npr=1;
Tx=1/Fx;  T=L*Tx;  Td=(L/N)*Tx;  F=Fx/L;  Fd=(N/L)*Fx;

for p = 1 : npr
   q0=p/(npr+1); q1=1-q0;
   GPSD01; GPSD02; PSDPLOT;  title('MODIFIED DUOBINARY');
   text(.3,.3,'yc');       text(.4,.5,'Yxc');
   text(.05,.8,'Yxd=0');  text(.05,.6,'yd=0');
end

if npr>1
  mesh(yc);  title('MODIFIED DUOBINARY');
else
end
```

```
% Computes the PSD of the Miller code

d=[0,1];   D=[d(1);d(2)];              [M,N]=size(D);
x=[0,1];   X=[x(1), x(1); x(1), x(2);
               x(2), x(1); x(2), x(2)];  [J,L]=size(X);

S1=[ 0 1 0 0
     0 0 0 1
     0 0 0 1
     0 1 0 0]; S2=[ 0 0 1 0
                    0 0 1 0
                    1 0 0 0
                    1 0 0 0]; Z1=[ X(1,:)
                                   X(4,:)
                                   X(4,:)
                                   X(1,:)]; Z2=[  X(2,:)
                                                  X(2,:)
                                                  X(3,:)
                                                  X(3,:)];
[I,I]=size(S1); SM=[S1;S2]; ZM=[Z1;Z2];

j=sqrt(-1); Fx=1; nFx=1;  nf=30; npr=9;
Tx=1/Fx;  T=L*Tx; Td=(L/N)*Tx;  F=Fx/L; Fd=(N/L)*Fx;
for p = 1 : npr
  q0=p/(npr+1); q1=1-q0;
  GPSD01;  GPSD02;  PSDPLOT;   title('MILLER');
  text(.3,1,'yc');     text(.7,3,'Yxc');
  text(.05,3,'Yxd=0');  text(.05,2,'yd=0');
end

if npr>1
  mesh(yc);  title('MILLER');
else
end
```

```
% Computes the PSD of the 3B2T-RBS scheme
% for different specification of the input symbol probabilities (see Chapter 4)
% FOR BINARY PROBABILITIES MAKE choice=2
% FOR TERNARY PROBABILITIES  choice=3

choice=3;
d=[0,1];   D=allbin(3);     [M,N]=size(D);
x=[0,1];   X=allbin(N+1);   [J,L]=size(X);

R1=[1 0 0 0]; R2=[0 1 0 0]; R3=[0 0 1 0]; R4=[0 0 0 1];
        % R1-R4 specify the possible rows of the state-transition matrices for a four-state scheme
                                 % this allows the matrices to be presented in a more concise form
S1=[R3;R4;R1;R2]; S2=[R2;R3;R2;R3]; S3=[R4;R1;R4;R1];
S4=[R1;R2;R3;R4]; S5=[R1;R4;R1;R4]; S6=S3; S7=S3; S8=S1;
Z1 = [X(10,:); X(10,:); X( 7,:); X( 7,:)];          % the look-up table of the output words
Z2 = [X(15,:); X( 2,:); X(15,:); X( 2,:)];                             % has been computed
Z3 = [X(12,:); X( 9,:); X( 8,:); X( 5,:)];
Z4 = [X( 7,:); X( 7,:); X(10,:); X(10,:)];
Z5 = [X(13,:); X( 4,:); X(13,:); X( 4,:)];
Z6 = [X(14,:); X( 3,:); X(14,:); X( 3,:)];
Z7 = [X( 8,:); X( 5,:); X(12,:); X( 9,:)];
Z8 = [X( 6,:); X( 6,:); X(11,:); X(11,:)];
[I,I]=size(S1); SM=[S1;S2;S3;S4;S5;S6;S7;S8];
             ZM=[Z1;Z2;Z3;Z4;Z5;Z6;Z7;Z8];

j=sqrt(-1); Fx=1; nFx=1;   nf=50; npr=1;
Tx=1/Fx;  T=L*Tx; Td=(L/N)*Tx;  F=Fx/L; Fd=(N/L)*Fx;

for p = 1 : npr
   q0=p/(npr+1); q1=1-q0;
   if choice==2
   GPSD01;  % the subroutine which computes the PSD, using the binary symbol probabilities
   elseif choice==3
   PSD01RBS;          % subroutine, computing the PSD with the ternary symbol probabilities
   end
   GPSD02;   PSDPLOT;
end
if npr>1
  mesh(yycc);
else
end
```

```
% Computes the PSD of the 1T2B code, based on the 3B2T-RBS scheme
% for different specification of the input symbol probabilities (see Chapter 4)
% FOR EQUAL PROBABILITIES -   choice=2.1
% FOR UNEQUAL PROBABILITIES -   choice=2.2

choice=2.1;

d=[0,1,2];   D=d';     [M,N]=size(D);
x=[0,1];   X=allbin(N+1);   [J,L]=size(X);

S1=[ 1 0 0 0
     0 1 0 0
     0 0 1 0
     0 0 0 1]; S2=[ 0 0 0 1
                    1 0 0 0
                    0 0 0 1
                    1 0 0 0]; S3=[ 0 0 1 0
                                   0 0 0 1
                                   1 0 0 0
                                   0 1 0 0];
Z1=[ X(3,:)
     X(3,:)
     X(2,:)
     X(2,:)]; Z2=[  X(4,:)
                    X(1,:)
                    X(4,:)
                    X(1,:)]; Z3=[   X(2,:)
                                    X(2,:)
                                    X(3,:)
                                    X(3,:)];
[I,I]=size(S1); SM=[S1;S2;S3];  ZM=[Z1;Z2;Z3];

j=sqrt(-1); Fx=1; nFx=1;   nf=50; npr=1;
Tx=1/Fx;  T=L*Tx; Td=(L/N)*Tx;  F=Fx/L; Fd=(N/L)*Fx;

for p = 1 : npr
   PSD01RBS;   GPSD02;   PSDPLOT;
end

if npr>1
   mesh(yc);
else
end
```

```
% Simulates the mB1C code for m=3 and 5
% The C bit is added to different columns of D to investigate the effect

d=[0,1];  D=allbin(5);    [M,N]=size(D);
x=[0,1];  X=[D';(~D(:,5))']'; [J,L]=size(X);
        % The set of output words is formed by adding a complementary bit to the input words

S1=[1]; [I,I]=size(S1);
for sm=1:M
   SM(sm,:)=S1;       % Compiles the matrix SM which collects the state-transition matrices
end                                                              % for all input words
ZM=X;

j=sqrt(-1); Fx=1; nFx=1;   nf=50; npr=1;
Tx=1/Fx;  T=L*Tx; Td=(L/N)*Tx;  F=Fx/L; Fd=(N/L)*Fx;

for p = 1 : npr
   q0=p/(npr+1); q1=1-q0;
   GPSD01;   GPSD02;  PSDPLOT;
   xy=.5; xY=.7;
   fiy=round((xy*nf)/(nFx*Fx)); yy=yc(p,fiy+1)+.01;
   fiY=round((xY*nf)/(nFx*Fx)); yY=Yxc(p,fiy+1)+.01;
   text(xy,yy,'yc');    text(xY,yY,'Yxc');
   text(.05,.2,'Yxd');  text(.06,.25,'yd');
end
if npr>1
  mesh(yc);
else
end
```

The following four modules are the subroutines called by the main blocks of the spectral analysis algorithm. Their structures are different for each module and special comments are given separately for the essential groups of lines.

The subroutines for the spectral analysis procedure

```
% The subroutine GPSD01 computes the TPM, S and the components
% Vx, H0, H1, H2 of the expression, (3.10b) for the code word PSD.
% The subroutine is constructed assuming binary input symbols

for m = 1 : M                        % The input-word probabilities are computed in this loop
   Nm = sum(D(m,:));    Q(m) = (q0^(N-Nm))*(q1^Nm);
end 1

S=zeros(I,I);
for m = 1 : M                                      % The TPM S computed in this loop
   mi=[(m-1)*I+1:m*I];    S=S+Q(m)*SM(mi,:);
end

U=eye(S); E=ones(S); u=diag(U);          % Computes the additional parameters, required
P=u'/(S+E-U);  Sinf=u*P;                        % in the expression for the PSD function
v=diag(P);  USinf=U-Sinf;  SSinf=S-Sinf;                % as defined in section 3.2.3.2

Vx=zeros(1,L); H0=zeros(L,L);
H1=zeros(L,I); H2=zeros(I,L);
for m = 1 : M               % The components Vx, H0, H1, H2, defined by expressions (3.9),
   mi=[(m-1)*I+1:m*I];                                 % are computed in this loop
   Vx=Vx+Q(m)*P*ZM(mi,:); H0=H0+Q(m)*ZM(mi,:).'*v*ZM(mi,:);
   H1=H1+Q(m)*ZM(mi,:).'*v*SM(mi,:); H2=H2+Q(m)*ZM(mi,:);
end
H0Vx=.5*(H0-Vx.'*Vx);                   % additional variable introduced for concise notation
```

% The subroutine *GPSD02* - computes the general PSD for of the particular
% coder sets and the specified range of computational parameters, Fx, nFx, nf

```
for fi=0:nf;          % The values of the continuous PSD function are computed in this loop for
   fx=fi*(nFx*Fx)/nf; f=fx/L;     fd=(N/L)*fx;            % normalised frequency values
   xf(fi+1)=fx;        df(fi+1)=fd; z=exp(j*2*pi*f*T);
   Jz=USinf*(z*U-SSinf)^(-1); Jiz=USinf*((1/z)*U-SSinf)^(-1);
   Wz=H0Vx+H1*Jz*H2; Wiz=H0Vx+H1*Jiz*H2;
   Wxc=T*(Wz+Wiz.');                       % The continuous PSD of the code-word sequence
   for l=1:L
      wc(l,:)=(exp(-j*2*pi*fx*(l-1)*Tx))/L;       % The DEFRAMING transfer function
   end
   Yxc(p,fi+1)=wc'*Wxc*wc;            % The continuous PSD of the code-symbol sequence
   if fi > 0
     sincc(fi+1)=(sin(pi*fx*Tx)/(pi*fx*Tx));         % The modulator transfer function
   else
     sincc(fi+1)=1;
   end
   yc(p,fi+1)=sincc(fi+1)'*Yxc(p,fi+1)*sincc(fi+1);              % The continuous PSD
end                                                                    % of the line signal

nF=(Fx/F)*nFx;
for k=0:nF          % The values of the continuous PSD function are computed in this loop for
   Fk=k*F; kF(k+1)=Fk;                                        % normalised frequency values
   for l=1:L
      wd(l,:)=(exp(-j*2*pi*k*(l-1)/L))/L;          % The DEFRAMING transfer function
   end
   Yxd(p,k+1)=(abs(Vx*wd))^2;               % The PSD of the code-symbol sequence
   if k>0
   sincd(k+1)=(sin(pi*Fk*Tx)/(pi*Fk*Tx));           % The modulator transfer function
   else
   sincd(k+1)=1;
   end
   yd(p,k+1)=(abs(sincd(k+1))^2)*Yxd(p,k+1);    % The discrete PSD of the line signal
end
```

% For constant input-probabilities the PSD functions Wxc, Yxc, yc, Yxd, yd are row vectors
% containing the value of the PSD for each value of the frequency from specified range.
% For variable input probabilities these functions are matrices whose rows are as defined above
% for each set of probability values.

```
% The subroutine PSDPLOT - plots the continuous parts, yc(p,:); Yc(p,:)
% and the discrete parts, yd(p,:);Yd(p,:)
% of the line signal and the code-symbol sequence PSD functions

plot(xf,Yxc(p,:),xf,yc(p,:)), hold   on

% The plot of the discrete parts of the PSD functions are preceded
% by modification of the respective row vectors to achieve plotting
% of the discrete components (the spectral lines) as vertical bars
% for improved readability of the graphs.

plYkF=[kF(1),kF(1),kF(1)+xf(2)/2,kF(1)+xf(2)/2];
plykF=[kF(1),kF(1),kF(1)+xf(2),kF(1)+xf(2)];
plYd=[0,Yxd(p,1),Yxd(p,1),0];
plyd=[0,yd(p,1),yd(p,1),0];
for i=2:nF
   plYkF=[plYkF,kF(i)-.005,kF(i)-.005,kF(i)+.005,kF(i)+.005];
   plykF=[plykF,kF(i)-.01,kF(i)-.01,kF(i)+.01,kF(i)+.01];
   plYd=[plYd,0,Yxd(p,i),Yxd(p,i),0];
   plyd=[plyd,0,yd(p,i),yd(p,i),0];
end
plot(plYkF,plYd,plykF,plyd), hold off
```

% The subroutine - *PSD01RBS*. A modified version of the subroutine *GPSD01*.
% The module is constructed for ternary probabilities as required
% by the main block for the analysis of the 3B2T-RBS code and its 1T2B version,
% selected by specifying the value of the parameter *choice*

```
if choice==2.1
  Q=[0.333 0.333 0.333];              % equal probabilities for the ternary symbols 0,1,2
elseif choice==2.2
     Q=[0.25 0.375 0.375];            % unequal probabilities for the ternary symbols 0,1,2
  elseif choice==3       % selects unequal ternary probabilities for the 2T4B transformation
     a=0.0938; b=0.1406;          % the probabilities for the blocks of two ternary symbols
     Q=[a b a b b b a b a];                   % Q collects the probabilities for all valid
end                              % blocks of ternary symbols, used in the 3B2T-RBS code

S=zeros(l,l);
for m = 1 : M
   mi=[(m-1)*l+1:m*l];    S=S+Q(m)*SM(mi,:);
end
U=eye(S); E=ones(S); u=diag(U);
P=u'/(S+E-U); Sinf=u*P;
v=diag(P);   USinf=U-Sinf; SSinf=S-Sinf;
Vx=zeros(1,L); H0=zeros(L,L);
H1=zeros(L,l); H2=zeros(l,L);
for m = 1 : M
   mi=[(m-1)*l+1:m*l];
   Vx=Vx+Q(m)*P*ZM(mi,:); H0=H0+Q(m)*ZM(mi,:).'*v*ZM(mi,:);
   H1=H1+Q(m)*ZM(mi,:).'*v*SM(mi,:); H2=H2+Q(m)*ZM(mi,:);
end
H0Vx=.5*(H0-Vx.'*Vx);
```

Part II - Modules and Subroutines for the Simulation and the Analysis of Coding Structures from the Different Classification Categories

The main software blocks, specifying the codes from the different categories, are given in this section. Their structure is similar to the main spectral analysis modules from Part I, except that the block for each category specifies all possible state-transition and output matrices for a given number of states and a set of output words. Each block calls a main subroutine which is designed for a given size of the input-word set. The main subroutines generate all combinations of state-transition and output matrices which correspond to valid codes. For each of these combinations the computational subroutines are called, which perform the evaluation of the respective PSD functions, display the results in graphical form and store the data to disks.

Main modules of the code generating algorithm:

% This module computes the PSD of all possible codes from the D2S1X4 category

```
d=[0,1];  D=[d(1);d(2)];  [M,N]=size(D);                 % the input symbol and word sets
x=[0,1];  X=[x(1), x(1); x(1), x(2);
              x(2), x(1); x(2), x(2)];                    % the output symbol and word sets
                          [K,L]=size(X);

S1=[1];                     % one possible state-transition matrix (this is a one-state category)
Z1=[X(1,:)]; Z2=[X(2,:)]; Z3=[X(3,:)]; Z4=[X(4,:)];              % all possible output
                                                                  % matrices
SMm=[S1];                [SMl,l]=size(SMm); numSM=SMl/l;
ZMm=[Z1;Z2;Z3;Z4]; [ZMl,L]=size(ZMm); numZM=ZMl/l;
      % SMm and ZMm are matrices collecting all possible state-transition and output matrices
                              % The SM and ZM matrices are derived from SMm and ZMm

j=sqrt(-1);  Fx=1; nFx=1;  nf=30;                        % the computational parameters
Tx=1/Fx;  T=L*Tx; Td=(L/N)*Tx;  F=Fx/L; Fd=(N/L)*Fx;

D2SIXK;                                                   % the main computational subroutine
```

```
% This module computes the PSD of all possible codes from the D2S1X8 category

d=[0,1];  D=[d(1);d(2)];  [M,N]=size(D);
x=[0,1];  X=allbin(3);    [K,L]=size(X);

S1=[1];
Z1=[X(1,:)]; Z2=[X(2,:)]; Z3=[X(3,:)]; Z4=[X(4,:)];
Z5=[X(5,:)]; Z6=[X(6,:)]; Z7=[X(7,:)]; Z8=[X(8,:)];
SMm=[S1];                                [SMl,l]=size(SMm); numSM=SMl/l;
ZMm=[Z1;Z2;Z3;Z4;Z5;Z6;Z7;Z8]; [ZMl,L]=size(ZMm); numZM=ZMl/l;

j=sqrt(-1);  Fx=1; nFx=1;  nf=30;
Tx=1/Fx;  T=L*Tx; Td=(L/N)*Tx;  F=Fx/L; Fd=(N/L)*Fx;

D2SlXK;
```

```
% This module computes the PSD of all possible codes from the D4S1X4 category

d=[0,1];  D=allbin(2);    [M,N]=size(D);
x=[0,1];  X=allbin(2);    [K,L]=size(X);

S1=[1];
Z1=[X(1,:)]; Z2=[X(2,:)]; Z3=[X(3,:)]; Z4=[X(4,:)];

SMm=[S1];                  [SMl,l]=size(SMm); numSM=SMl/l;
ZMm=[Z1;Z2;Z3;Z4;]; [ZMl,L]=size(ZMm); numZM=ZMl/l;

j=sqrt(-1);  Fx=1; nFx=1;  nf=30;
Tx=1/Fx;  T=L*Tx; Td=(L/N)*Tx;  F=Fx/L; Fd=(N/L)*Fx;

D4SlXK;
```

% This module computes the PSD of all possible codes from the D2S2X2 category

```
d=[0,1];   D=[d(1);d(2)];  [M,N]=size(D);
x=[0,1];   X=[x(1); x(2)]; [K,L]=size(X);


S1=[ 0 1
     0 1]; S2=[  0 1
                 1 0]; S3=[  1 0
                             0 1]; S4=[  1 0
                                         1 0];
Z1=[ X(1,:)
     X(1,:)]; Z2=[    X(1,:)
                      X(2,:)]; Z3=[    X(2,:)
                                       X(1,:)]; Z4=[    X(2,:)
                                                        X(2,:)];

SMm=[S1;S2;S3;S4]; [SMl,l]=size(SMm); numSM=SMl/l;
ZMm=[Z1;Z2;Z3;Z4]; [ZMl,L]=size(ZMm); numZM=ZMl/l;


j=sqrt(-1);   Fx=1; nFx=1;   nf=30;
Tx=1/Fx;  T=L*Tx; Td=(L/N)*Tx;  F=Fx/L; Fd=(N/L)*Fx;


D2SlXK;
```

% This module computes the PSD of all possible codes from the D2S2X3 category

```
d=[0,1];       D=[d(1);d(2)];            [M,N]=size(D);
x=[-1,0,1];  X=[x(1); x(2); x(3)];  [K,L]=size(X);


S1=[ 0 1
      0 1]; S2=[  0 1
                  1 0]; S3=[  1 0
                              0 1]; S4=[  1 0
                                          1 0];
Z1=[ X(1,:)
      X(1,:)]; Z2=[    X(1,:)
                       X(2,:)]; Z3=[    X(1,:)
                                        X(3,:)];
Z4=[ X(2,:)
      X(1,:)]; Z5=[    X(2,:)
                       X(2,:)]; Z6=[    X(2,:)
                                        X(3,:)];
Z7=[ X(3,:)
      X(1,:)]; Z8=[    X(3,:)
                       X(2,:)]; Z9=[    X(3,:)
                                        X(3,:)];

SMm=[S1;S2;S3;S4];                        [SMl,l]=size(SMm); numSM=SMl/l;
ZMm=[Z1;Z2;Z3;Z4;Z5;Z6;Z7;Z8;Z9];
                                           [ZMl,L]=size(ZMm); numZM=ZMl/l;


j=sqrt(-1);   Fx=1; nFx=1;   nf=30;
Tx=1/Fx;  T=L*Tx; Td=(L/N)*Tx;  F=Fx/L; Fd=(N/L)*Fx;

D2SlXK;
```

% This module computes the PSD of all possible codes from the D2S2X4 category

```
d=[0,1];   D=[d(1);d(2)];                   [M,N]=size(D);
x=[0,1];   X=[x(1), x(1); x(1), x(2);
               x(2), x(1); x(2), x(2)];     [K,L]=size(X);

S1=[ 0 1
     0 1]; S2=[  0 1
                 1 0]; S3=[  1 0
                             0 1]; S4=[  1 0
                                         1 0];
Z1=[X(1,:); X(1,:)]; Z9 =[X(3,:); X(1,:)];
Z2=[X(1,:); X(2,:)]; Z10=[X(3,:); X(2,:)];
Z3=[X(1,:); X(3,:)]; Z11=[X(3,:); X(3,:)];
Z4=[X(1,:); X(4,:)]; Z12=[X(3,:); X(4,:)];
Z5=[X(2,:); X(1,:)]; Z13=[X(4,:); X(1,:)];
Z6=[X(2,:); X(2,:)]; Z14=[X(4,:); X(2,:)];
Z7=[X(2,:); X(3,:)]; Z15=[X(4,:); X(3,:)];
Z8=[X(2,:); X(4,:)]; Z16=[X(4,:); X(4,:)];

SMm=[S1;S2;S3;S4];                 [SMl,l]=size(SMm); numSM=SMl/l;
ZMm=[Z1;Z2;Z3;Z4;Z5;Z6;Z7;Z8;Z9;Z10;Z11;Z12;Z13;Z14;Z15;Z16];
                                    [ZMl,L]=size(ZMm); numZM=ZMl/l;

j=sqrt(-1);   Fx=1; nFx=1;   nf=30;
Tx=1/Fx;  T=L*Tx; Td=(L/N)*Tx;  F=Fx/L; Fd=(N/L)*Fx;

D2SlXK;
```

% This module computes the PSD of all possible codes from the D2S3X2 category

```
d=[0,1];  D=[d(1);d(2)];   [M,N]=size(D);
x=[0,1];  X=[x(1); x(2)];  [K,L]=size(X);


SR1=[0 0 1]; SR2=[0 1 0]; SR3=[1 0 0];

   % SR1-SR4 specify the possible rows of the state-transition matrices for a three-state scheme
                                   % this allows the matrices to be presented in a more concise form

S1 =[SR1; SR1; SR1];
S2 =[SR1; SR1; SR2];  S16=[SR2; SR3; SR1];
S3 =[SR1; SR1; SR3];  S17=[SR2; SR3; SR2];
S4 =[SR1; SR2; SR1];  S18=[SR2; SR3; SR3];
S5 =[SR1; SR2; SR2];  S19=[SR3; SR1; SR1];
S6 =[SR1; SR2; SR3];  S20=[SR3; SR1; SR2];
S7 =[SR1; SR3; SR1];  S21=[SR3; SR1; SR3];
S8 =[SR1; SR3; SR2];  S22=[SR3; SR2; SR1];
S9 =[SR1; SR3; SR3];  S23=[SR3; SR2; SR2];
S10=[SR2; SR1; SR1];  S24=[SR3; SR2; SR3];
S11=[SR2; SR1; SR2];  S25=[SR3; SR3; SR1];
S12=[SR2; SR1; SR3];  S26=[SR3; SR3; SR2];
S13=[SR2; SR2; SR1];  S27=[SR3; SR3; SR3];
S14=[SR2; SR2; SR2];
S15=[SR2; SR2; SR3];

Z1=[X(1,:); X(1,:); X(1,:)];  Z5=[X(2,:); X(1,:); X(1,:)];
Z2=[X(1,:); X(1,:); X(2,:)];  Z6=[X(2,:); X(1,:); X(2,:)];
Z3=[X(1,:); X(2,:); X(1,:)];  Z7=[X(2,:); X(2,:); X(1,:)];
Z4=[X(1,:); X(2,:); X(2,:)];  Z8=[X(2,:); X(2,:); X(2,:)];

SMm=[S1;S2;S3;S4;S5;S6;S7;S8;S9;S10;S11;S12;S13;S14;S15
        S16;S17;S18;S19;S20;S21;S22;S23;S24;S25;S26;S27];
                                        [SMl,l]=size(SMm); numSM=SMl/l;
ZMm=[Z1;Z2;Z3;Z4;Z5;Z6;Z7;Z8]; [ZMl,L]=size(ZMm); numZM=ZMl/l;


j=sqrt(-1);  Fx=1; nFx=1;  nf=50;
Tx=1/Fx;  T=L*Tx; Td=(L/N)*Tx;  F=Fx/L; Fd=(N/L)*Fx;


D2SlXK;
```

```
% This module computes the PSD of all possible codes from the D2S4X2 category

d=[0,1];   D=[d(1);d(2)];    [M,N]=size(D);
x=[0,1];   X=[x(1); x(2)];   [K,L]=size(X);

Sr=[0 0 0 1; 0 0 1 0; 0 1 0 0; 1 0 0 0];    % a supplementary matrix containing all possible
                                            % rows for a four-state state-transition matrix
for r1=1:4                        % this cycle compiles the matrix SMm collecting all possible
  for r2=1:4                                          % four-state state-transition matrices
    for r3=1:4
      for r4=1:4
  R=(r1-1)*4^3+(r2-1)*4^2+(r3-1)*4+r4;
  SMm(R*4-3:R*4,:)=[Sr(r1,:);Sr(r2,:);Sr(r3,:);Sr(r4,:)];
      end
    end
  end
end
for r1=1:2                        % this cycle compiles the matrix ZMm collecting all possible
  for r2=1:2                                          % one-symbol output-word matrices
    for r3=1:2
      for r4=1:2
  R=(r1-1)*2^3+(r2-1)*2^2+(r3-1)*2+r4;
  ZMm(R*4-3:R*4,:)=[X(r1,:);X(r2,:);X(r3,:);X(r4,:)];
      end
    end
  end
end

[SMI,I]=size(SMm); numSM=SMI/I;
[ZMI,L]=size(ZMm); numZM=ZMI/I;

j=sqrt(-1);   Fx=1; nFx=1;   nf=30;
Tx=1/Fx;  T=L*Tx; Td=(L/N)*Tx;  F=Fx/L; Fd=(N/L)*Fx;

D2SIXK;
```

The subroutines for the code generating algorithm

% The module D2SIXK computes the PSD for all categories
% specified with a set of two source words D=[0,1]

```
for s1 = 1 : numSM              % this cycle generates all combinations of indices which specify
   s1i = [(s1-1)*l+1 : s1*l];                 % a set of two (M=2) state-transition matrices
   for s2 = 1 : numSM                     % which are selected from the matrix SMm to form
      s2i = [(s2-1)*l+1 : s2*l];          % the computational matrix SM for a generated code
      SM=[SMm(s1i,:);SMm(s2i,:)];

      npr=1; p = 1;   q0=p/(npr+1); q1=1-q0;        % determines the input probabilities
      GPSD01A;                                   % calls the subroutine computing the TPM S

      U=eye(S); E=ones(S); u=diag(U);   % Computes some of the additional parameters
                    % required in the expression for the PSD function as defined in section 3.2.3.2
      Seig = 1; %(sum(abs(eig(S))+eps>=1)==1);          % The validity of the TPM is
      Srow = (all(sum(S.'))==1);                          % verified by this group of lines
      SEUdet = (det(S+E-U)~=0);                        % to eliminate invalid code structures
      Scheck = (Seig & Srow & SEUdet);
      if Scheck == 1
         P=u'/(S+E-U);  Pcheck = ~(any(P<=0));
      end
      if (Scheck & Pcheck) == 1          % Computes the rest of the additional parameters
         Sinf=u*P;  v=diag(P);   USinf=U-Sinf;  SSinf=S-Sinf;          % for a valid TPM

         for z1 = 1 : numZM  % this cycle generates all combinations of indices which specify
            z1i = [(z1-1)*l+1 : z1*l];                   % a set of two (M=2) output matrices
            for z2 = z1 : numZM          % which are selected from the matrix ZMm to form
               z2i = [(z2-1)*l+1 : z2*l];                % the matrix ZM for a generated code
               ZM=[ZMm(z1i,:);ZMm(z2i,:)]; nX=1;
               for zr1=1:M*l      % this loop skips matrices which produce ambiguous codes
                  for zr2=(zr1+1):M*l
                     neX=any(ZM(zr1,:)~=ZM(zr2,:));
                     nX=nX+neX;
                  end
               end

               if nX>=M
```

```
                eS1=int2str(s1); eS2=int2str(s2);       % prepares the character string for
                Zet1=int2str(z1); Zet2=int2str(z2);      % the name of the data-store file
                GPSD01B;  GPSD02; PSDPLOT;             % the PSD evaluation subroutines
                title(['S',eS1,'S',eS2,'Z',Zet1,'Z',Zet2]);
                D2ORDR% the subroutine producing the data storage order (Section 5.2.1.1)
              else
              end
            end
          end
        else
        end
        D2SAVE;                              % the subroutine saving the spectral analysis data

    end
  end
```

```
% The module D4SIXK computes the PSD for all categories
% specified with a set of four source words D=[00,01,10,11]
% The same as module D2SIXK except that the loops generating
% the state-transition and the output matrices are modified for M=2

for s1 = 1 : numSM
   s1i = [(s1-1)*l+1 : s1*l];
   for s2 = 1 : numSM
      s2i = [(s2-1)*l+1 : s2*l];
      SM=[SMm(s1i,:);SMm(s2i,:)];
      for s3 = 1 : numSM
         s3i = [(s3-1)*l+1 : s3*l];
         for s4 = 1 : numSM
            s4i = [(s4-1)*l+1 : s4*l];
 SM=[SMm(s1i,:);SMm(s2i,:);SMm(s3i,:);SMm(s4i,:)];

      npr=1; p = 1;   q0=p/(npr+1); q1=1-q0;
      GPSD01A;

      U=eye(S); E=ones(S); u=diag(U);
      Seig = (sum(abs(eig(S))+eps>=1)==1);
      Srow = (all(sum(S.'))==1);
      SEUdet = (det(S+E-U)~=0);
      Scheck = (Seig & Srow & SEUdet);
      if Scheck == 1
         P=u'/(S+E-U);  Pcheck = ~(any(P<=0));
      end
      if (Scheck & Pcheck) == 1
         Sinf=u*P;  v=diag(P);   USinf=U-Sinf;  SSinf=S-Sinf;

   for z1 = 1 : numZM
      z1i = [(z1-1)*l+1 : z1*l];
      for z2 = z1 : numZM
         z2i = [(z2-1)*l+1 : z2*l];
         for z3 = z2 : numZM
            z3i = [(z3-1)*l+1 : z3*l];
            for z4 = z3 : numZM
               z4i = [(z4-1)*l+1 : z4*l];
 ZM=[ZMm(z1i,:);ZMm(z2i,:);ZMm(z3i,:);ZMm(z4i,:)];
```

```
                    eS1=int2str(s1); eS2=int2str(s2);
                    eS3=int2str(s3); eS4=int2str(s4);
                    Zet1=int2str(z1); Zet2=int2str(z2);
                    Zet3=int2str(z3); Zet4=int2str(z4);
                    GPSD01B;  GPSD02; PSDPLOT;
               title(['S',eS1,eS2,eS3,eS4,'Z',Zet1,Zet2,Zet3,Zet4]);
                    D4ORDR;
                end
              end
            end
          end
          D4SAVE;
             else
             end
               end
             end
           end
        end
```

The subroutines for the spectral analysis procedure

```
% The subroutine GPSD01A computes the TPM, S (similar to GPSD01)
% It is constructed assuming binary input symbols

for m = 1 : M
   Nm = sum(D(m,:));
   Q(m) = (q0^(N-Nm))*(q1^Nm);
end
S=zeros(I,I);
for m = 1 : M
   mi=[(m-1)*I+1:m*I];
   S=S+Q(m)*SM(mi,:);
end
```

```
% The subroutine GPSD01B (similar to the second part of GPSD01)
% computes the components Vx, H0, H1, H2
% of the expression, (3.10b) for the code word PSD.

Vx=zeros(1,L); H0=zeros(L,L);
H1=zeros(L,I); H2=zeros(I,L);
for m = 1 : M
   mi=[(m-1)*I+1:m*I];
   Vx=Vx+Q(m)*P*ZM(mi,:); H0=H0+Q(m)*ZM(mi,:).'*v*ZM(mi,:);
   H1=H1+Q(m)*ZM(mi,:).'*v*SM(mi,:); H2=H2+Q(m)*ZM(mi,:);
end  %3
H0Vx=.5*(H0-Vx.'*Vx);
```

```
% The subroutines GPSD02 and PSDPLOT are the same as those given in Part I
```

% The subroutine D2ORDR constructs matrices containing the values
% of yxd and yxc for different PSD plots only and
% compiling the matrix ORDz, used to derive the table of equivalence
% which associates codes with the respective PSD functions

```
si=(s1-1)*numSM+s2;              % generate the sequential numbers of the state-transition
zi=(z1-1)*numZM+z2;                                          % and the output matrices
if ~(exist('ycc'))
  ydd=yd; Yxdd=Yxd;  ycc=yc; Yxcc=Yxc;         % initiates the PSD collecting matrices
  zi=1; ORDz(zi,:)=[si,zi,s1,s2,z1,z2,1];    zi=zi+1;
else
  for l=1:indx % this loop checks for equivalent PSD functions (the continuous parts are used)
     Ryc=(abs(yc-ycc(l,:))<eps);     % Ryd=(yd-ydd(l,:)<eps);          % the discrete
     if (all(Ryc))==1                % & all(Ryd))==1           % parts can be added, too
       ORDz(zi,:)=[si,zi,s1,s2,z1,z2,l]; zi=zi+1;
       break
     elseif l==indx
          ydd=[ydd;yd]; Yxdd=[Yxdd;Yxd];
          ycc=[ycc;yc]; Yxcc=[Yxcc;Yxc];
          ORDz(zi,:)=[si,zi,s1,s2,z1,z2,indx+1]; zi=zi+1;
     end
  end
end
[indx,nff] = size(ycc);
```

```
% The subroutine D4ORDR is functionally the same as D2ORDR
% except that the indices, specifying the state-transition and
% the output matrices are computed for M=4

si=(s1-1)*numZM^3+(s2-1)*numZM^2+(s3-1)*numZM+s4;
zi=(z1-1)*numZM^3+(z2-1)*numZM^2+(z3-1)*numZM+z4;
if ~(exist('ycc'))
   ydd=yd; Yxdd=Yxd;  ycc=yc; Yxcc=Yxc;
   ORDz(zi,:)=[zi,z1,z2,z3,z4,1];
else
   for l=1:indx
       Ryc=(abs(yc-ycc(l,:))<eps);  %  Ryd=(yd-ydd(l,:)<eps);
       if (all(Ryc))==1             %  & all(Ryd))==1
          ORDz(zi,:)=[zi,z1,z2,z3,z4,l];
          break
       elseif l==indx
             ydd=[ydd;yd]; Yxdd=[Yxdd;Yxd];
             ycc=[ycc;yc]; Yxcc=[Yxcc;Yxc];
             ORDz(zi,:)=[zi,z1,z2,z3,z4,indx+1];
       end
   end
end
[indx,nff] = size(ycc);
```

% The subroutine D2SAVE saves the PSD data for the D2SIXK codes

```
if ((Scheck & Pcheck) == 1)|(z1==numZM)        % checks if a valid code exists
  eM=int2str(M); al=int2str(l); Kei=int2str(K);  % produces characters to label
  zeti=int2str(zi); esi=int2str(si);       % the files according to the codes category
  DR=['d',eM,'s',al,'x',Kei];                     % the store directory name
  FL=['d',eM,'s',esi,'z',zeti];                        % the store file name
  SAVEa = ['save a:\',DR,'\',FL];
  YydcfF=[' ydd',' Yxdd',' ycc',' Yxcc',' xf',' kF'];
  eval(['!md a:\',DR])
  eval([SAVEa,YydcfF,' ORDz'])
  clear ydd Yxdd ycc Yxcc ORDz z1
  save cdtemp; clear; load cdtemp                 % cleans the working space
  eval(['dir a:\',DR])
else
end
```

% The subroutine D4SAVE is the same as D2SAVE but is used for M=4

```
if (Scheck & Pcheck) == 1
  eM=int2str(M); al=int2str(l); Kei=int2str(K);
  zeti=int2str(zi); esi=int2str(si);
  DR=['d',eM,'s',al,'x',Kei];
  FL=['d',eM,'s',esi,'z',zeti];
  SAVEa = ['save a:\',DR,'\',FL];
  YydcfF=[' ydd',' Yxdd',' ycc',' Yxcc',' xf',' kF'];
  eval(['!md a:\',DR])
  eval([SAVEa,YydcfF,' ORDz'])
  clear ydd Yxdd ycc Yxcc ORDz
  save cdtemp; clear; load cdtemp
  eval(['dir a:\',DR,'*.mat'])
else
end
```

APPENDIX - B

A SPREAD SPECTRUM TECHNIQUE FOR TWO–LEVEL OPTICAL LINE SIGNALLING

Georgi P. Petkov, City University, London

Abstract : This paper describes a method for implementing spread spectrum techniques in binary signal transmission over optical fibres. The presentation of the proposed method is preceded by an overview of the essential arguments in support of spectrum spreading applications in fibre-optic systems.

I. INTRODUCTION

There have been many publications showing the potential offered by the Spread Spectrum (SS) techniques for multi-user communication systems. The rapid increase in the development of Fibre-Optic Communication Systems (FOCS) during recent years created new areas for investigating the benefits of spectrum spreading. Multiple access communication networks on optical fibres appear to be quite promising.

Section II summarises most of the essential arguments for developing SS Systems on optical fibres and contains an overview of some of the published achievements in this area. Some results reported in [8, 9] are introduced in section III. The basic idea of combining coded data signals into a multi-level line signal is presented. It is suggested in section IV that code multiplexing and multiple access operation could be efficiently achieved through two-level line signalling.
A technique for transmission of SS information in a binary coding format is proposed, which is more appropriate for optical-fibre systems than a multi-level format. The purpose of the extended preliminary discussion is to provide the background to the development of the suggested method.

II. SS IN OPTICAL COMMUNICATIONS

SS techniques have been extensively developed for multiple-access mobile communications and other radio transmission systems. The development of many successful applications for SS Systems in general and those for Code Division Multiple Access (CDMA) implies even better opportunities for using spectrum spreading in FOCS. The advantages of the optical-fibre transmission medium are its constant and predictable parameters the inherent noise immunity. The significant interfering signals are only those from different stations of the same communication system.

Combining the advantages of SS Systems and FOCS seems very attractive. It is suggested even by a general outline of the main features of both types of systems. For transmission over optical fibres these are :
– high information capacity due to the extremely broad bandwidth of the optical fibres;
– high energy efficiency (power per unit information) due to extremely low transmission losses.
The basic advantage of SS Systems is :
– the processing gain achieved by the controlled

redistribution and the subsequent recovery of the signal throughout some very wide frequency band. Thus the SS techniques are efficient at high bit rates and the optical fibres are one of the best media to provide this. More arguments for the application of SS techniques in FOCS can be developed by going into the detailed theory of both areas [1, 2].

Some reported results show the main directions of developing highly efficient optical communication networks with SS implementations. An example of a straightforward solution is given in [3]. The ground terminals of a satellite communication system can cope with an increased number of user-distribution stations by using optical fibre cables for direct retransmission of high frequency SS signals. Other publications [4,5] give just a general view of possible implementations. The idea that is common to most reports is achieving a substantial increase in either performance or in the number of users when suitable spreading sequences are used in multiple access networks. A basically different approach is discussed in [6,7]. It concerns applying spectrum spreading methods directly in the optical frequency domain.

III. THE MULTI-LEVEL LINE SIGNALLING

The most explored area of SS applications in FOCS is the design of high capacity information networks based on CDMA. Good examples of practical results are given in [8, 9]. Each station is asigned a suitable code sequence. The spectrum spreading is achieved by sending that code sequence for data bit '1' and its inverse for data bit '0'. All stations can transmit simultaneously through optical coupling. The result is a multi-level optical line-signal. Maximal-length pseudo-random sequences are considered unsuitable for the proposed code multiplexing because the lower bound of the maximal crosscorrelation is restricted,

$$\max_m \left|R_{a,b}(m)\right| > -1+2^{(n+1)/2},$$

for any two maximal sequences a and b of length $N=2^n-1$ (n is the number of stages of a shift register generator). An additional disadvantage is the relatively small number of maximal sequences of a given periodic length. Much better results have been achieved by using Gold sequences. Although their out-of-phase autocorrelation contains small non-zero peaks, their crosscorrelation function takes only three values. The number of Gold sequences of given period N is N+2 which is sufficient for the purposes of code multiplexing. It has also been shown in [8] that the number of stations which allow for error free performance depends on the period of the code sequences. With sufficiently long codes there are conditions under which N+1 stations can transmit simultaneously with no error.

The ideas from [8] have been applied in a Local Area Network experiment with 'optical processing' which is reported in [9]. It is suggested that an optical correlator is used at the receiving side. Such a correlator would consist of optical delay lines whose combined outputs yield pulses corresponding to the autocorrelation peak of a certain coding sequence. This can greatly reduce the operational speed of the optical receiver.

The main problem with the optical processing is the number of the delay lines which should be equal to the number of ones in a particular random sequence.

The results of both reports described above show quite a good potential for practical achievements. However there are conditions which may restrict to some extent the area of applications. One of them is the necessity of securing a very careful balance in the optical 'mixing' of the signals on the line.
The disproportionate contribution of the separate stations could result in uncontrollable interference levels. Another problem is that multi-level optical signals are quite susceptible to the noise from optical components.

IV. THE SS BINARY LINE SIGNALLING

The main objective of this section is to suggest some possibilities for transforming the SS techniques to the time domain in order to avoid the need for a multi-level optical signal. The 'Time Spreading' approach is equivalent in principle to the normal spectrum spreading. By this method a binary, as oposed to a multi-level, signal is generated for transmission over the optical fibre.

A very general model is adopted to describe the transition from the conventional SS Systems to the proposed method for code multiplexing. It is assumed that the binary signals of N information channels are to be transmitted simultaneously. Each signal is spread by a unique coding sequence of length N and a bit rate $F_c=NF_d$, where F_d is the information bit rate.

The total channel capacity for baseband digital transmission is given by

$$C=\sum_{1}^{N}C_i=N[F_c(1+\frac{s_i}{\nu})],$$

where $\frac{s_i}{\nu}$ is the signal/noise ratio for the i-th channel. When all channels come from separate sources and share the same media a multi-level signal is formed. For the purposes of
the presentation the values of this signal are considered to range over all integers from 0 to N. Thus within a code-bit period the value k=0 ... N of the multi-level signal equals the number of channeles whose code bits for that period are 'one'. This is equivalent to approximately an N times lower $\frac{s_i}{\nu}$ which at the same time, is compensated for by a processing gain approximately equal to $(F_c/F_d)=N$. In terms of information capacity this is also equivalent to N channels over separate transmission lines.The multi-level type of line signal can be visualized by simulating a 7 channel system with Gold codes of length 7. This is not a practical case but it can illustrate the general features of multi-level signalling. (The simulation for 31 bit Gold sequences looks the same but is difficult to illustrate in one figure.) A 3-D perspective-image of the line signal values for all possible 7-tuples of binary channel data is shown in fig.1a. When the conventional decoding is preceded by optoelectronic conversion and hard-limiting of the line signal, determination of the threshold level is quite critical. An illustration of the of the hard-limited line-signal values from fig.1a for the 7 channel model is given in fig.1b. The 'high' and 'low' values of the two-level signal correspond to

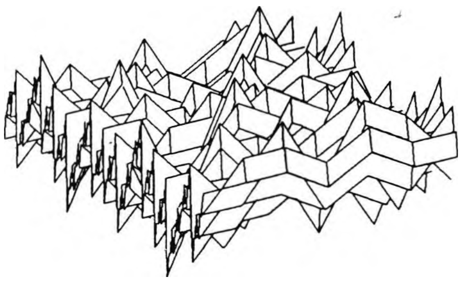
the signal values above and below the threshold respectively. Finally the picture in fig.1c shows the decoded original data after processing the line signal from fig.1b with all code sequences.

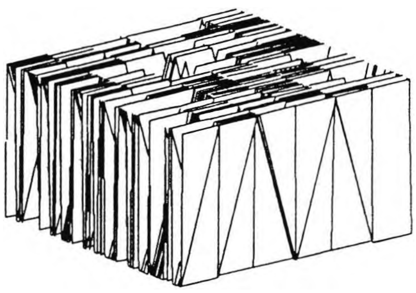
It is claimed by the results of [8] the suggested code multiplexing can be better than the conventional time-domain multiplexing .

— possibility to multiplex information bits asynchronously as long as the code bits are in synchronization;

— the receiver does not have to demultiplex all channels as it can decode only those of interest.

To achieve the same effect in code multiplexing by using two-level signalling it is


a) the space of all signal values


b) the hard-limited signal

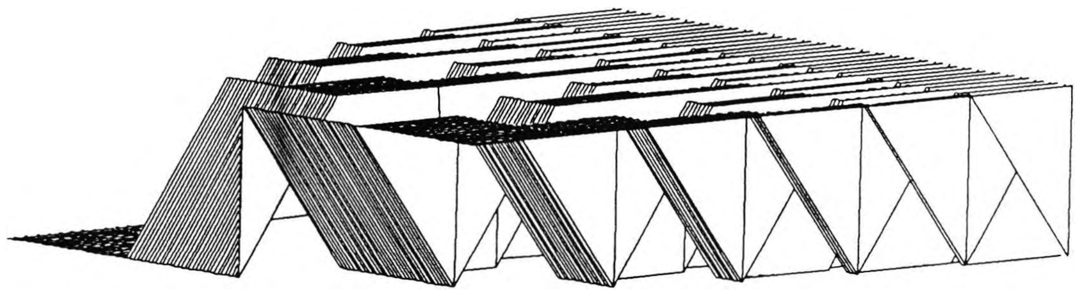
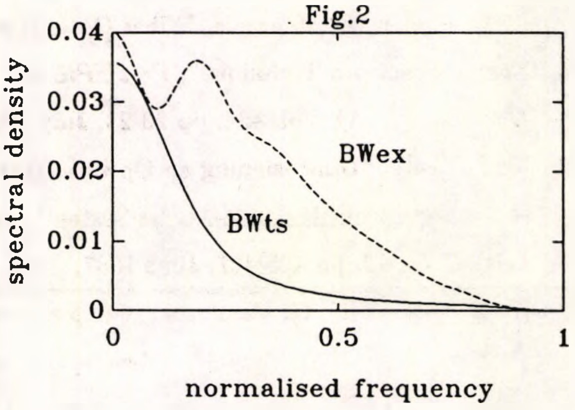


Fig. 1 c) the space of the decoded values

This can be considered an improvement for a hypothetical communication system where the transmitting side multiplexes the information channels by producing a multi-level signal identical to the one resulting from separate sources using the same line. It is obvious that such a code multiplexing technique could be preferred provided a suitable optical source exists. The important advantages would be:

obvious from the expression for C that a bit rate of the order of NF_c is required. Sending the coding sequences of all channels sequentially is a direct binary equivalent of transmission over N separate lines. (If this is done within one information bit period the bit rate becomes N^2F_d, which is far from efficient signalling.) However, as this is the extreme in bandwidth, the power spectral density of such a coding format has been calculated to define the upper boundary of the increase in the necessary frequency band. The graph is shown in fig. 2 as BW_{ex}. Some small improvement could be achieved if the code bits with equal numbers from the different spreading sequences are transmited sequentially. This introduces a uniform distribution of pulses in the line signal regardless of the number

Fig.2



of channels multiplexed which is essential to the amount and the distribution of the timing content. This is still not very practical especially with longer spreading sequences.

A substantial improvement can be achieved by transmitting the SS information embedded in the multi-level signal in binary form. A straight forward example is for each code-bit period to send a pulse of duration

$$t_c = \frac{k}{N} T_c = \frac{k}{N^2} T_d \text{ , where}$$

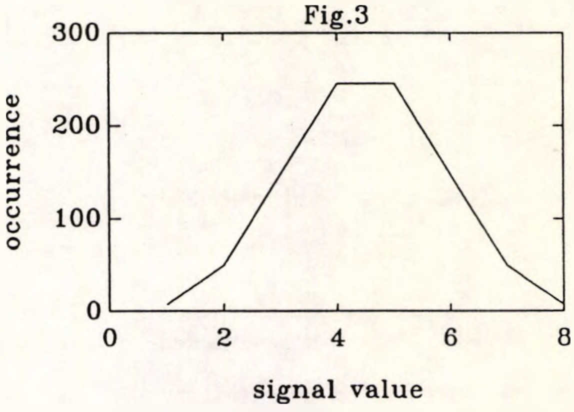
$k = 1 \ldots N$ is the corresponding value from the multi-level signal. $T_c = 1/F_c$ and $T_d = 1/F_d$ are the code and information bit periods respectively. The coding rules for the time-spreading technique are directly related to the multi-level model simulation. This implies that the decoding of the binary signal is achievable through processing stages similar to those illustrated above.

The coding can be technically simple in spite of the big set of possible substitutions. The solution is to find the optimal ratio between the states and the code words in a finite state sequential machine model of the line coder.This technique has been modelled to compute the power spectral density of the resulting code-multiplexed line signal.

The graph is presented in fig. 2 as BW_{ts} and shows a noticeable reduction in the occupied bandwidth.

Further development of the proposed coding is to invert the coding pulses for k smaller and bigger than some predetermined values. It can be set for example to $k < 0.25(N+1)$ and $k > 0.75(N+1)$.

Fig.3



A particular threshold value for pulse inversion may be determined from the statistical distribution of the average number of values in the multi-level signal. An example of such a distribution is shown in fig.3. It can be seen that a binary signal produced by the time-spreading method contains the whole SS information of the multi-level optical signal and its bandwidth is much smaller than the one of direct time-domain multiplexed code sequences. However, the increase in the transmission bit rate makes the proposed technique practical mainly for FOCS.

V. SUMMARY AND CONCLUSIONS

The main advantages in using SS techniques in optical-fibre communications have been discussed. Some reported results have been presented in brief to illustrate code multiplexing for multi-level optical-fibre transmission.

A method has been proposed for data-channel multiplexing through producing a binary equivalent of the multi-level SS signal. It has been shown that an increase in the transmission bit-rate can be maintained within reasonable limits. The time-spreading technique requires higher bandwidth but this is easily compensated for by the use of two-level line signalling, which is more appropriate for fibre-optic communications than a multi-level format.

There are still many unexploited possibilities for improvement of the proposed technique. The necessary bandwidth can be further reduced by constructing codes which would have the pulse contents of a data period T_d optimised without altering the amount of SS information.

REFERENCES

[1] H. B. Killen, "Digital Communications with Fiber Optics and Satellite Applications", *Prentice-Hall Inc*, 1988.

[2] R. L. Pickholtz et al, "Theory of Spread-Spectrum Communications — A tutorial", *IEEE Trans. on Communications*, Vol COM-30, pp 855-884, May 1982.

[3] J. J. Pan et al, "Applications of Fiber Optics for MILSATCOM Earth Terminals", *IEEE MILCOM 1982*, pp 18.3-1/18.3-6, 1982.

[4] C. A. Laber, M. Kaveh, "Spread Spectrum signalling for Distributed Fiber Optic Systems", *IEEE Trans. on Aerospace and Eln. Syst.*, Vol. AES-16, pp 440-445, July 1980.

[5] P. Pfieffer, P. Mayrueis, "Fiber Optic Bus with Spread Spectrum Technique", *Proc SPIE Int. Soc. Opt. Eng. (USA)*, Vol. 434, pp 20-23, July 1984.

[6] P. Healy, "Dimensioning an Optical-Fiber Spread-Spectrum Multiple-Access System", *Optics Letters*, Vol 12, pp 425-427, June 1987.

[7] G. J. Faschini, G. Vannucci, "Using Spread-Spectrum in A High-Capacity Fiber-Optic Local Network", *IEEE J. of Lightwave Technology*, Vol 6, pp 370-379, March 1988.

[8] S. Tamura et al, "Optical Code-Multiplex Transmission by Gold Sequences", *IEEE J. of Lightwave Technology*, Vol LT-3, Feb. 1985.

[9] P. R. Prucnal et al, "Spread Spectrum Fiber-Optic Local Area Network Using Optical Processing", *IEEE J. of Lightwave Technology*, Vol LT-4, May 1986.

A LOW COST OPTICAL LINK FOR INTEGRATED VOICE AND DATA TRANSMISSION OVER DISTRIBUTED COMPUTER NETWORKS

N. Nicolaou, G. Petkov, R. Comley

Centre for Information Engineering, City University, Northampton Square, London EC1V 0HB, England

Abstract: The possibility for integrated voice and data transmission through a mesh type of local area network has been investigated. A model of 20Mbps fibre-optic link has been designed to connect two nodes. Because the distances to be covered are relatively short and the number of tranceivers is large, a low-cost solution has been adopted. This was achieved by a very simple circuit design and off-the-shelf components. A three level line code carrying the full timing information was used to increase the reliability of transmission. The design considerations concerning the transmitter and receiver circuits and results of measuring the transmission performance of the fibre-optic link are presented in the following paper.

INTRODUCTION

The rapid proliferation of personal computers and the desire for the distribution of computing power and resources has made Local Area Networks (LANs) an important feature in areas such as industry, business, university campuses, military installations, etc. LANs have been developed to provide a wide range of services many of which have widely differing demands. Some applications require only data or voice communication, some require both and more recently video transmissions have begun to appear. This diversity of applications has led to the establishment of many different standards and techniques.

For some applications the LAN is not a critical component while for others a LAN failure could be potentially catastrophic. A LAN for these latter applications has been suggested in [1]. It has a regular mesh (lattice) topology with its nodes situated on the cross-over points and connected to their immediately adjacent neighbours via bi-directional links. The LAN was designed to offer high levels of reliability (achieved through its multipath structure) and voice-data integration. It can support up to 128 nodes and uses 20 Mbps point-to-point fibre-optic links.

Conventional fibre-optic transmitters and receivers are generally expensive making the cost of the mesh topology prohibitively high. This paper describes a simple transmitter/receiver unit suitable for use in the above network but with potential for many other general purpose applications.

LINE CODING OOBJECTIVES

In the case of point-to-point links the transmitter and the receiver must be synchronised either by using a separate line carrying the clock signal or by timing information contained within the transmitted data. This latter solution is the one preferred in almost all cases except where the distances involved are very small. There are many different line codes in use today [2,3].

The most important characteristics affecting the selection of a line code are its spectral distribution, codec complexity, bit timing content, error detection capability and transmission efficiency [3]. In the case of LANs, achieving optimum or maximum line code efficiency is generally of less importance than reducing hardware complexity. Also, for short lines where the Bit Error Rate (BER) may be reduced to very low levels ($\leq 10^{-9}$ typ.), a line code with an inherent error detection capability is not considered necessary. The bit rate of the line signal implies that the fibre-optic (F-O) link is not dispersion limited and the spectral distribution of the line code will not raise the problems of variable frequency response and propagation delay, typical for metallic lines. From the above, it would therefore be reasonable to suggest that line codes, suitable for LANs based on optical fibres, should contain adequate timing information and have simple coder and decoder circuits.

Most of the line codes used with optical fibres are two-level codes. This was necessary because of the non-linearity of early optical sources and the fact that two-level codes would allow longer distances between repeaters. However, optical sources are now being made with much more linear characteristics, and since long distances are not generally a feature of LANs, multi-level codes can be seriously considered. The simplest multi-level code is a three-level code, an example of which is shown in Fig.1.

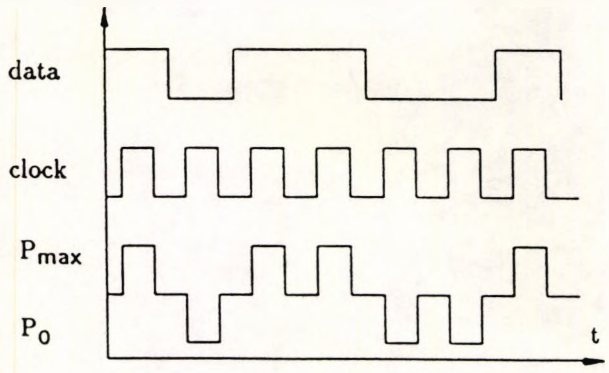


Fig.1 Data and line-signal formats

This code is similar to the Polar RZ code [3] the only difference being that the signal is not a voltage but light and the levels are not +V, 0 and −V but P_0, 0.5 P_{max} and P_{max} (P is the light intensity). In this code 1's are represented by a pulse from 0.5 P_{max} to P_{max} and 0's by a pulse from 0.5 P_{max} to P_0. Both pulses are half a bit-cell wide and are located in the middle of the bit-cell.

Since there are two transitions in every bit-cell it means that the line code contains the full clock information. Of course, using three levels to transmit binary data is not the most efficient way but this is outweighted by the savings in other areas. For example, there is no need for any additional synchronisation bits and it offers the possibility for continuous line monitoring with simple circuitry and immediate line fault detection. Also, the decoder circuit for the three level line code is very simple compared with other codes offering similar timing information. The circuit diagram of a coder and decoder for such a code is shown in Fig.2.

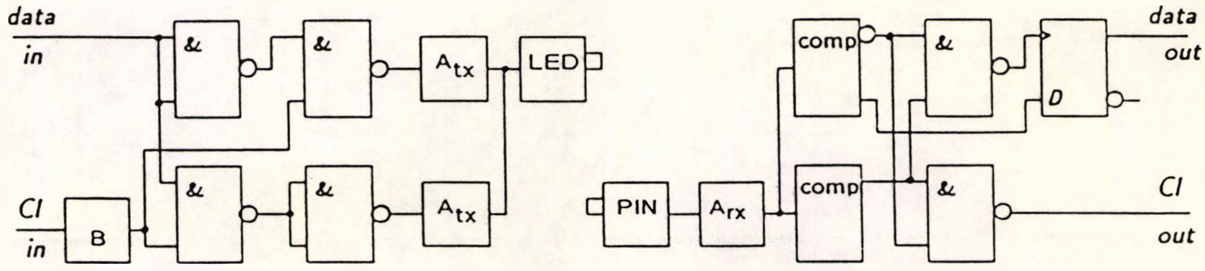


Fig.2 The optical transmitter and receiver circuits
(A_x - amplifier; LED - light emitting diode; PIN - photodetector; B - buffer; comp - comparator)

DESIGN CONSIDERATIONS

The purpose of the F-O link is to provide simple and reliable connection between adjacent nodes of the proposed network. Although the distances between connected nodes are relatively small, a F-O link is still a better choice than a conventional metallic line, because it can easily provide the necessary 20Mbps bit rate and take advantage of the cheap plastic coated silica type of fibres. Also the immunity of F-O links to external electrical interference can be very significant for certain LAN installations.

Simple calculations on the optical power budget for our link show that the maximum distance covered is around 1.5Km, assuming a photodiode sensitivity of -25dBm, typical connector loss of 2dB and a 3dB span margin. The above distance is more than adequate for the needs of the proposed network.

The parameters of the fibre optic components presented in Table 1 show that the optical power of the LED and the photodiode sensitivity can be easily matched with appropriate fibre characteristics. As a result, most of our attention has been directed towards achieving high reliability through good timing characteristics of the transmitted signal.

Table 1

General characteristics of the F-O link	20 Mbps clock rate; NRZ/TTL level data input and output: Three-level optical pulse format; 850nm transmission wavelength
Transmitter	LED optical power source mounted in an SMA bulk type connector (160W into 200m PCS); 6ns typical rise and fall times
Receiver	SMA connectorised PIN Photodiode; Transimpedance preamplifier stage (A_{rx})

A LED driving circuit (A_{tx}) was designed to provide the necessary response time for the conversion of the high speed TTL output into fast high-current driving pulses at the required bit rate. Using a single IC (4x2 NAND gates) and two high-frequency transistors, a very simple circuit for converting the two-level NRZ data into a three-level optical signal was achieved. As each network node incorporates four Transmitter-Receiver units, the simplicity of the circuitry is of great importance. The advantages of transmitting full timing information allow a very simple decoding circuit to be used at the receiver site. There is no need for a local clock generator or synchronising circuit.

CIRCUIT DESCRIPTION

The very simple TTL logic circuit of the transmitter (Fig.2) combines the clock and the NRZ pulses to produce separate streams of pulses for the 1's and 0's respectively. These signals switch the two driving transistors to conduct the maximum or zero value of LED current for 50% of each clock period.

The receiver utilises the broad-band frequency response of a transimpedance preamplifier followed by a high speed 5539 opamp to produce the necessary levels for the discriminating circuit. Both 1 and 0 pulse streams are detected by comparators and a simple circuit is used to recover the clock and the NRZ data signals. The relative time shifting between these two sequences is proportional to the variation of the DC component around the zero (mid-level) value when long sequences of consecutive 1's and 0's are transmitted. This does not present a serious problem for our network because:

— the construction of the data packages is such that long sequences of 1's or 0's will never exceeo the measured limit
— the line signal is formed so that when clock pulses or both clock and data pulses are absent, a constant half- maximum level optical signal is fed into the line.

The constant middle level optical power transmission ensures that the transmitter/receiver circuits are permanently biased which speeds up response times and allows a constant line integrity check to be performed.

TEST RESULTS

Two basic parameters of the F-O link were measured by the Digital Transmission Analyser ME520B. The 20Mbps transmission rate was achieved by applying an external clock signal to the ME520B transmitter unit (ME520A-Tx). The BER and timing jitter were measured with the ME520B receiver unit (ME520A-Rx).

For any of the TTL level test signal, i.e. $(2^{10}-1)$, $(2^{15}-1)$ and $(2^{23}-1)$ pseudorandom sequences or different conditions of manually set 16 bit words, the F-O link showed error-free performance over measuring intervals from 10 to 180 minutes duration.

The values of the measured timing jitter were well below the CCITT recommendation limits which is taken to be quite acceptable for the designed network. This level of performance is achieved through the excellent synchronisation characteristics of the receiver circuit, which locks onto the input signal immediately after it crosses either detection level.

The only conditions which degraded the F-O link performance occurred when long sequences of zeros were added to the data bit stream. The results show that 64 consecutive zeros in every $2^{23}-1$ bit pseudorandom sequence bring the BER up to 10^{-3}. Occasionally repetitive loss of synchronisation occurred although the values of the measured timing jitter did not exceed 0.2UI. This phenomenon can be explained by the relatively high level of base-line drift (Fig.3) which may be easily eliminated through the use of appropriate data formats.

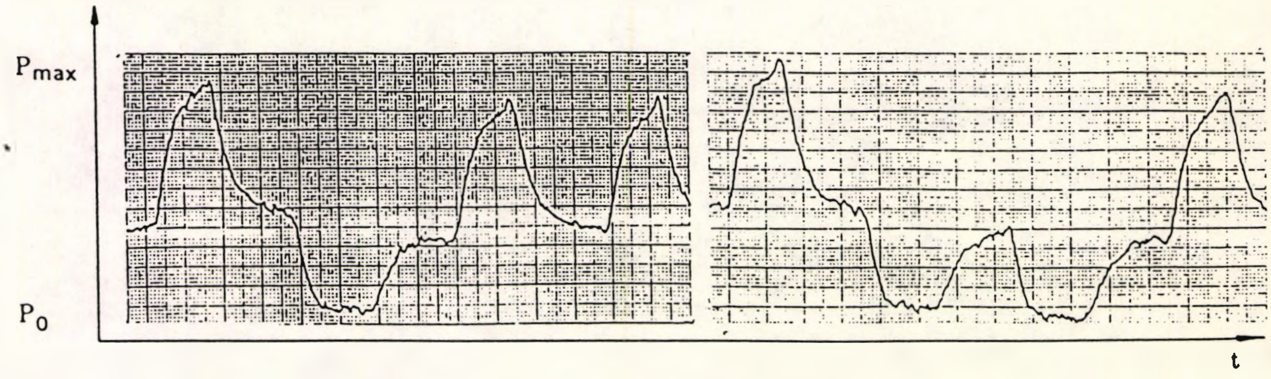


Fig. 3 Optical pulse-waveform

CONCLUSIONS

This paper shows that fibre optic links for moderate transmission rates can be easily built using cheap "off-the-shelf" components. A 20Mbps optical link suitable for transmission over distances of up to 1.5Km has been successfully demonstrated.

Although not used extensively with optical fibres, three-level codes with full timing information and simple transmitter-receiver circuits are shown to offer a very cost-effective and reliable solution for LANs which employ point-to-point links of relatively short length.

Availability of suitable optical components (e.g. the tri-stable laser diode [4]) should allow significant improvements in performance to be achieved with three-level optical codes. Further plans include the investigation of optical transmission formats and line codes suitable for integrated voice, data and video communications.

ACKNOWLEDGEMENTS

Two of the authors, G. Petkov and N. Nicolaou, would like to acknowledge the support of the U.K. Science and Engineering Research Council by the provision of ORS awards. We would also like to thank the ISRT of The Bulgarian Posts for providing test facilities.

REFERENCES

[1] Nicolaou, N.C. and Comley, R.A. "High Reliability LAN supporting Speech and Data Communications using a Mesh Topology." Proceedings of the IASTED International Symposium on Computers and Their Applications for Development, Taormina, Italy, Sept.1986, pp. 188-192

[2] Sorencen, H.O. "Use of Standard Modulation Codes for Fiber Optic Link Optimisation", Proceedings of the Eighth International Fiber Optics Communications & Local Area Networks Exposition, Las Vegas, Sept. 1984, pp. 55-58

[3] Miller and Ahamed, Digital Transmission Systems and Networks, Volume 1, Chapter 2, 1987, Computer Science Press.

[4] Watanabe, M. et al "Optical tristability and associated ternary digital functions using a twin-stripe laser diode", IEE Proceedings, Vol.135, Part J, No.1, Feb. 1988, pp. 74-78

REFERENCES

1. MARSHALL, G. J.; Principles of Digital Communications; McGraw-Hill, 1980.
2. MILLER, M. J., AHAMED, S. V.; Digital Transmission Systems and Networks (vol. I: Principles); Computer Science Press, 1988.
3. KILLEN, H. B.; Digital Communications with Fibre Optics and Satellite Applications; Prentice-Hall International Editions, 1988.
4. PEEBLES, Jr. P. Z.; Digital Communication Systems; Prentice-Hall International Editions, 1987.
5. TRÖNDLE, K. and SÖDER, G.; Optimisation of Digital Transmission Systems; Artech House, 1987.
6. LYNN, P. A.; An Introduction to the Analysis and Processing of Signals; Macmillan Education ltd., 1989.
7. CATTERMOLE, K. W., O'REILLY, J. J.; Mathematical Topics in Telecommunications (vol. 2: Problems of Randomness in Communication Engineering); Pentech Press Limited, 1984.
8. SHANNON, C.; The Mathematical Theory of Communication; Illinois UP, 1949.
9. HAMMING, R. W.; Coding and Information Theory; Prentice-Hall, 1980.
10. LIN, S., COSTELLO, Jr. D. J.; Error Control Coding: Fundamentals and Applications; Prentice-Hall, Inc., Englewood Cliffs, N. J. 07632, 1983.
11. BERLEKAMP, E. R.; Algebraic Coding Theory; McGraw-Hill Book Company, 1968.
12. DIXON, R. C.; Spread Spectrum Systems; John Wiley & Sons Inc., 1984.
13. SIMON, M. K., et al; Spread Spectrum Communications (vol. I); Computer Science Press, Inc., 1985.
14. SKAUG, R., HJELMSTAD, J. F.; Spread Spectrum in Communication; Peter Peregrinus Ltd., 1985.
15. "MATLAB for 80386-based MS-DOS Personal Computers"; User's Guide; The MathWorks Inc., 1989.

16. HAWKER, I.; Future trends in digital telecommunication transmission networks; *Electronics & Communication Engineering Journal*, 1990, vol. 2, No. 6, 251-260.
17. CATTERMOLE, K. W.; Principles of Digital Line Coding; *Int. J. Electronics*, 1983, vol. 55, No. 1, 3-33.
18. CARIOLARO, G. L., el al; Analysis of Codes and Spectra Calculations; *Int. J. Electronics,* 1983, vol. 55, No. 1, 35-79.

19. PETERSON, W. W.; Encoding and Error-Correction Procedures for the Bose-Chaudhuri Codes; *IRE Trans. Inf. Theory,* 1960, IT-6, 459-470.
20. INGRAM, D. G. W.; Digital line codes - an introduction and overview; *Digest of Colloquium on Analytical foundations for digital line codes held at University of Essex,* 4th of June 1981, 1-11.
21. BROOKS, R. M.; Design of a 7B8B line code with improved error monitoring capabilities; *Digest of Colloquium on Analytical foundations for digital line codes held at University of Essex,* 4th of June 1981, 87-103.
22. SHARLAND, A. J., STEVENSON, A.; A Simple In-Service Error-Detection Scheme Based on the Statistical Properties of Line Codes for Optical Fibre Systems; *Int. J. Electronics,* 1983, vol. 55, No. 1, 141-158.
23. WINSTEN, C. B.; Transition matrices and Markov chains; *Digest of Colloquium on Analytical foundations for digital line codes held at University of Essex,* 4th of June 1981, 22-33.
24. POO, G. S.; Computer aids for code spectra calculations; *Digest of Colloquium on Analytical foundations for digital line codes held at University of Essex,* 4th of June 1981, 58-86.
25. BROOKS, R. M. and JESSOP, A.; Line coding for optical fibre systems; *Int. J. Electronics,* 1983, vol. 55, No. 1, 81-120.
26. POROKHOV, O. N., RADEV, P.; 3B2T-RBS - An efficient transmission method for digital fibre-optic communications; *Electronic Letters,* Oct. 1986, Vol. 22, No. 21, 1125-1126.
27. GARDNER, W. A., FRANKS, L. E.; Characterisation of cyclostationary random signal processes; *IEEE Transaction on Information Theory,* 1975, Vol. 21, 4-14.
28. CARIOLARO, G. L., TRONCA, G. P.; Spectra of block coded digital signals; *IEEE Transactions on Communications,* 1974, Vol. 22, 1555-1564.
29. FARREL, P. G.; Coding as a cure for communication calamities: The successes and failures of error control; *Electronics & Communications Engineering Journal,* 1990 Vol.2, No. 6, 213-220.
30. TAKASAKI, Y., et al; Two-level AMI line coding family for optical fibre systems; *Int. J. Electronics,* 1983, vol. 55, No. 1, 121-131.
31. KRZYMIEN W. A.; Transmission Performance Analysis of a New Class of Line Codes for Optical Fiber Systems; *IEEE Transactions on Communications,* 1989, Vol. 37, No 4, 402-404.
32. PETROVIC, R.; Low Redundancy Optical Fiber Line Code; *Journal of Optical Communications,* 9 (1988) 3, 108-111.
33. ALI A. M.; Multiplexing and Line Coding Schemes for Broadband-ISDN; *IEEE CH3214-3/86/0000- 0371,* 1986, 371-375.