MODULATION CODES FOR MOBILE COMMUNICATIONS

by

Pieter Gert Wessel van Rooyen

TIMSIS

submitted as partial fulfillment for the

requirements for the degree

MASTER OF ENGINEERING

in

ELECTRICAL AND ELECTRONIC ENGINEERING

in the

FACULTY OF ENGINEERING

at the

RAND AFRIKAANS UNIVERSITY

SUPERVISOR: PROF. H.C. FERREIRA

June 1991

DEDICATION

Dedicated to my God,
who gives me tnlents to develop and the ability to work,
and to my wife, Venessa,
who gives me the freedom and latitude to do so.

SUMMARY

This thesis sets alit to a theoretical and experimental investigation of communication systems, in particular digital communication systems. Finite-state machine representations of modulation codes arc investigated and a procedure to map a class of coding rules into fixed rate state systems arc presented. The performance of binary modulation codes on recording channels have been studied fairly extensively. this is usually not thc case for other bnndlimited channels, and this thesis addresses the gap: various modulation codes were transmitted through filters with various cut off frequencies and slopes to obtain quantitative results for modulation codes through band limited channels in general.

As application of the above results, a selection of constrnined codes were studied for clock extraction when transmitting data over bandwidth-limited, voice band VHF FM mobile communication systems.

OPSOMMING

llicrdie tesis behandel 'n tcoretiesc en ekspcrimentele ondersock van kommunikasiestelscls, in die besonder digitale kommuniknsiestelscls. Eindige-tocstands·masjien-voorstellings van modulasiekodes word ondersock en In metodc om 'n klas vaste-lengte kodes op dié masjienc af te beeld, word aangebicd, Die verrigting van binere modulasickodes op opnemer kanale is alreeds goed ondersoek, wat nie die geval is vir onder bandbeperkte kanale nie. llierdie tcsis spreek dié gaping aan: verskeie modulasiekodes word deur filters met verskillende afsnyfrekwensies en hellings gestuur om kwantitatiewe resultate te vcrkry deur bandbeperktc kanale.

As toepassing van bogenoemde resultate word 'n paar spcsificke modulasickodcs bcstudeer vir klokhcrwinning oar mobiele BHF FM kommunikasiekanale.

ACKNOWLEDGEMENT

I am indebted to many for their advice and assistance during the period of this study; suggestions and help from my fellow students, in particular Francis Swarts, who never hesitated in spending hours to be of assistance to me, and to Prof. Il.C. Ferreira for the constructive suggestions and guidance throughout the project.

CONTENTS PAGE

APPENDICES

UST OF FIGURES

UST OF TABLES

LIST OF SYMBOLS

A_O	Free-space transmission loss
C	Shannon capacity
C	Charge constraint
D	Present data bit
DR	Density ratio
d	Minimum runlength
Δ_0	Durst density
E	Electric field
E_v	State transition matrix
f;	NRZ channel bits
f_s	State characterizing function
f_z	Output characterizing function
r	Output matrix
I	Number of states
K	Number of input words
k	Maximum runlength
l	Distance between two antennas
λ	Eigen value
M	Finite-state machine
m	Input word length
μ	Number of errors
N	Number of output words
$Nd(n)$	Number of distinct d sequences of length n
n	Codeword length
v	Block of v bits

0	Previous codeword
$P(\mu, v)$	Block error probability
R	Code ratc
S	Finite-state set
S_v	State of machine at time t_v
C_1	State in state set
T	State transition matrix
t_v	time at v
V	Approximate eigenvector
X	Finite input alphabet
X_v	Input symbol
ξ	Element of finite input alphabet
Y_i	Channel signals
Z	Finite output alphabet
Z_v	Output symbol
ζ	Element of finite output alphabet

CHAPTER 1

INTRODUCTION

Alexander Graham Bell 0847-1922) invented and patented in 1876 the first telephone that was of any real practical usc. In 1874 he said: *"IfI could make a current of electricity vary in intensity precisely as the air varies in density during the production 01 sound, J should be able to transmit speech telegraphically."* With this remark as principle for a telephone. analog telecommunication as we know it today was on its way.

Until Alexander Graham Bell, the world of communication was a digital world. The preeminent digital device was the telegraph, which communicated by turning an electrical current on and off. It was simple. robust and effective. By pressing down the telegraph key, the operator completed the circuit. Bolding it down for a short time created a "dot"; a longer contact constituted a "dash." 111e presence or absence of gross electrical energy defined a message by means of an accepted code (the Morse code was one of several). The chief constraint was the speed of transmission. A good operator could manage 25 or 30 words, occasionally 40 words per minute [11.

Before long it occurred to a number of inventors. including 1110mas Alva Edison (1874·1931), that the keys could be rigged to transmit mechanically at much higher speeds

than human operators could attain. '11e Morse letters could be punched out on a paper tape ahead of time and the tape could then be drawn through a special automatic keying device to activate the telegraph transmitter at a faster rate. By the 1870's, automatic telegraphs were transmitting at rates in excess of 500 words per minute over test lines and rates of from 100 to 200 words per minute were routinely achieved on ordinary inter city telegraph circuits. This was the wircline corollary of today's goal of "spectrum efficiency."

By utilizing the same wire circuits but transmitting at a much higher rate, the carrying capacity of the telegraph was greatly increased.

Telegraphers encountered a special problem, however, when the first transatlantic cables were laid in the 1850's and 1860's. Here were true long distance circuits, reaching some 2500 kilometers from Ireland to Newfoundland. The laying of these huge cables out across the open Atlantic in waters 3 kilometers deep was a feat of engineering audacity comparable to digging the Panama Canal or landing on the Moon and it captured the public imagination and came to symbolize technological progress for the Victorian era. Yet the physical challenges of laying the cable were matched by the unforeseen electrical challenges involved in actually transmitting signals on these immensely long circuits. Of course regeneration was impossible. For reasons that were not fully understood at the time, the effective transmission rate on the transatlantic circuits was reduced to two or three letters per minute. Here. on the most expensive circuit of all, where wireline "transmission efficiency" was most needed. the transmission rates were so slow, three hours or so for the text contained on one page of the average book, that even after the cable was installed, it was still quicker to transport a typical journal across the ocean by ship than to transmit its contents by undersea telegraph.

The first "multilevel coding" composure was used to improve the situation somewhat. The English scientist William Thomson (who became the first Baron Kelvin) realized that the ordinary Morse code was itself rather inefficient. The symbols only recognized two level of current ("on" and "off") and two intervals of time ("short" and "long") and did not even utilize these code parameters very efficiently. With such symbols it was not necessary to transmit up to five symbols ("dots" and "dashes") to signify a single letter of the alphabet. Thomson constructed an apparatus that was capable of transmitting symbols based on up to five distinct voltage levels. In Thomson's code, two of these five level symbols were sufficient for unique designation of 25 of the 26 letters in the alphabet. Thomson's multilevel code had increased the information density of each symbol. It was the symbol rate that was limited by the electrical and physical characteristics of the cable. If each

symbol could be made to carry more information. the overall throughput of the cable could be increased substantially, to from 16 to 20 words per minute.

In modem terminology, Bell's undulatory current is referred to as *analog* transmission. Instead of transmitting discrete pulses of energy. an analog system like Bell's telephone transmits a complex. unbroken electrical waveform. which corresponds closely to the waveforms produced by the original sounds of human speech.

Analog transmission proved to be simple. reliable and economical for large scale applications. It was heralded as a breakthrough from the older telegraphic thinking. The idea of waveform reproduction guided early radio researchers as well. All the radio systems developed through the first half of the twentieth century utilized the same basic analog signal processing principles of wireline telephony. In general. from 1876 until about 1950. analog transmission reigned supreme among all communication media. with only vestigial survivals of the older telegraphic techniques.

Over the past thirty-five years. digital systems have penetrated every segment of society. Digital switches are revolutionizing telephone central-office and network control functions. Digital transmission systems are now installed on more than half of all interchange telephone trunks, The use of digital microwave is increasing. In the local distribution segment hundreds of thousands of "digital" subscriber loops are being installed every year. Fiber optics. another technological buzzword for the 1990's. utilizes digital technology. Digital techniques are invading other industries, Digital audio systems are rapidly displacing the standard analog media (LP records and analog audio tape cassettes). Digital television is on the horizon. Digital photography. digital x rays. are now being developed. Obviously. the pervasiveness of the computer has transformed the workplace and brought digital electronics into millions of homes.

Against this backdrop. cellular mobile radio in its current form will almost certainly be the last major analog communication system ever deployed. Also. analog cellular radio finds itself in the early 1990's in a situation similar to that of the LP record industry a few years back. faced with the onset of digital compact laser disk media. Compact disk (CD) market share has rocketed from almost nothing in 1980 to overtake analog LP's by 1990. Communication engineers are beginning to talk about *digital cellular* in terms which suggest a similar upheaval of current industry patterns in mobile telephony.

An interesting observation in the history of analog vs digital, is that digital was the first

and last. Digital was replaced by analog and with the might of a battleship regained its position as the supreme. The processes of digital communication are awe-inspiring, in speed, in precision, in terms of sheer technical accomplishment, yet these processes are clearly much more complex than Bell's simple telephone! A question therefore arises: Why bother? Or, to put it another way, why does digital, with all its expensive overhead, payoff?

111is question and other interesting aspects of digital communication, digital cellular radio and the radio channel itsclf will be discussed in chapter two.

Alan Mathison Turing's (t912-1954) claim to fame is as one of the fathers of the electronic computer. His 1936 paper, "On Computable Numbers, with an application to the Entscheidungsproblem" is the classic in its field. The *Entscheidungsproblem*: *"to find a method for deciding whether or not a given formula is a logical consequence of some other given formulae."* lie dreamed of making a "brain", his Universal Machine, essentially what we would now recognize as an automatic digital computer with internal storage. This "brain" of Turing could be realized with a finite-state machine with millions of states. (The formal definitions for a finite-state machine and a state can be found in chapter three).

In chapter three a formal or "computer scientist" discussion concerning *finite-state machines* (FSM) is presented. Although some of the material presented in this chapter will not have any direct relation to work done in this thesis, it will serve as an introduction to the theory of finite-state machines for students to follow and finite-state-machine-enthusiasts alike.

Modulation codes are also known as runlength limited codes, constrained codes, line codes or *dk* codes and are employed on self-synchronizing digital communication systems. These codes usually find application on digital magnetic and optical recorders [3]. As another (Jesser known) application for these codes, a selection of modulation codes were investigated for clock extraction when transmitting data over bandwidth-limited, voice band VHF FM mobile communication systems. Chapter four presents the necessary background on the relevant information theory and its implications on digital communication systems.

Chapter five deals with applications of finite-state machines, maybe less impressive and with a million or so states less than Turing's "brain", in digital communication systems.

Furthermore, finite-state machine representations of modulation codes are investigated and a procedure is presented to map a class of coding rules into fixed rnte systems. This procedure to set up state systems for codes with memory, described by fixed length rules, is based on an engineering approach to sequential design [4J. A method for converting between Mealy and Moore representations of finite-state machines is also presented in this chapter, together with a method to obtain a minimum Moore machine.

A special class of modulation codes with only a maximum nmlength constraint, were considered for use on mobile communication channels. The results obtained is presented in chapter six.

Experiments were conducted and custom designs were undertaken in this study. Chapter seven gives a system-based description of the various facets of the experimental set up which was developed.

Codes with various runlength and charge constraints were generated and transmitted through highpass, lowpass and bandpass filters with various cut off frequencies and slopes to obtain quantitative results for these codes through bandlimited channels.. The parameters of the code selected also determine the shape of the power spectral density and the bandwidth requirements of the modulated FM signal on the channel. Chapter eight deals with these results.

Chapter nine is the "proof of the pudding": the results obtained for modulation codes transmitted over mobile VHF radio channels.

Since the topics investigated in this study are still new, some meaningful suggestions for future research arc made in chapter ten together with some concluding remarks.

Supplementary material including circuit diagrams, programs written in Turbo C and TMS Assembler are included in the appendices at the back of the thesis, and since we arc living in the "digital" age, an IBM compatible floppy disk, with some of the programs developed. is presented in the cover of this thesis.

CHAPTER 2

MOBILE COMMUNICATIONS

The telephone was introduced to the public in 1876 at the Centcnnial Exposition of the United States in Philadelphia. Alexander Graham Bell was able to transmit speech electrically, in one direction only, over a copper wire circuit of several hundred meters in length. Although not everyone present that day could immediately perceive its commercial value, the "speaking telegraph" was quickly perfcctcd for adequate two-way communication and was offered for business and residential service the following ycar. Within a short time there were thousands, then tens of thousands and soon hundreds of thousands of paying customers.

TIle story of the evolution of the wire network is in large part the story of a long struggle against thc burden and expense of the physical wire plant. Scientists were led deeper and deeper into the study of electrical transmission, to understand in detail the characteristics of wire media and to search (or ways to utilize this costly transmission facility more and more efficiently.

Toward the end of the nineteenth century, while this struggle was only beginning, a young Gcnnan scientist named lIeinrich Rudolf Hcrtz discovered a strange and wonderful phenomenon: from an electric spark of sufficient intensity there seemed to emanate

invisible waves of force which could be captured at a distant location by a suitable constructed receiving device. It seemed to be a realization of the ancient concept of "action at a distance." Classical physicists found this philosophically disturbing and postulated the existence of some impalpable intervening medium, the Ether, which actually transmitted these strange waves. Hertz's own experiments extended only over a few meters. A few years later, Guglielmo Marconi transmitted these waves over several kilometers, and began to call it Radio.

The early telephone engineers, caught up in the heroic and unrelenting struggle with copper wire physics and economics, looked upon the new phenomenon with awe. *"/1 has been shown,"* wrote one John J. Carty in 1891, *"that longer waves may be generated which arc capable of electrical action ami which can be propagated Ihrough the densest jog and even through a stone wall with just as much ease as through the clearest atmosphere [1].*

Communication devices which exploited these "Jlertzian" waves experienced a much slower technological gestation but a more rapid market development than the telephone. Radio broadcasting was introduced commercially in the United States in 1921. Within ten years, more than 50% of all American households boasted a radio set. Within twenty years, it was over 90%. The growth of television, another technology based on radio transmission, was even more rapid. Once it became readily available to the American public in 1946 it took only nine years to reach the 50% level and only fourteen years to reach 90%. Such inventions, inherently far more complex and costly than the telephone instrument, were able to spread so much more rapidly because they did not require a massive investment in wireline infrastructure.

John J. Carty, however, writing in 1891, had not foreseen either radio or TV broadcasting as we know it today. For him, the promise of Hertz's discoveries lay in another direction: itA *system of telephony without wires seems one of the interesting possibilities..."*

The possibility for utilizing radio devices for communicating with moving vehicles was quickly appreciated. The earliest commercial application of radio had been for communication with ships at sea. As early as 1921, the Detroit Police Department was conducting experiments with "mobile" radio. Throughout the 1930's, experience with mobile communication accumulated. It was World War II, however, and the sudden and pressing need for two-way mobile communication on a large scale, that gave the real impetus to mobile radio technology. It is impossible to imagine any of the characteristic tactical operations of that war functioning effcctively without radio.

At the end of the war, the first licenses were granted for the provision of true mobile telephone services; in other words, to allow a user calling by radio from a moving vehicle to be interconnected into the public telephone network. TIle car phone was conceived. The war had brought to prominence a new type of radio technology, frequency modulated radio or FM, which permitted superior mobile voice communication. Interfaces to telephone switches had been established. Americans were buying automobiles in record numbers. Prosperity had returned. Along with television, mobile radio seemed poised for a postwar boom.

But, something went wrong. From the promise of those early commercial systems in the late 1940's, the actual deployment of mobile telephone systems proved painfully slow. More than forty years Inter, even after the development of "modem" cellular radio systems, the actual development of the market for mobile telephony was abysmal. In the densest traffic centers in the US, the penetration in the mid-1980's is considerably less than 1%!1 Moreover, based on current technology the available spectrum is loaded to near capacity. Even to double this penetration will apparently involve very substantial technical and economic challenges.

Mobile telephony [t] has undoubtedly set the record for the slowest penetration by any technology to the mass marketplace.

- *Cost* is not the whole explanation. The cost of ordinary wireline telephony in the early years of the twentieth century was, in relative terms, much higher than the cost of mobile service today;
- *Spectrum shortage* is no explanation at all, but a symptom;
- *Insufficient demand* is most decidedly *not* the explanation. If anything, the indications have all been strongly in the other direction, tending to show a very large demand for affordable mobile communication services.

The realization is growing among industry observers that interconnected mobile radio, *in its current configuration,* cannot become the mainstream mobile service that its designers once hoped for. However, we have advanced to the point where no policeman, fireman, taxi-driver or sccurity guard would be able to fulfill his designated role efficiently without suitable radio equipment.

Today "cellular radio" stands in the spotlight, with its hopes illuminated and its deficiencies exposed, still pretending to a degree of technological permanence that once

seemed more valid than it docs today, The "mobile revolution" is, however, much larger than the current generation of cellular radio and its immediate problems, although severe in some respects, should be recognized as developmental rather than fundamental. Consider a parallel case. Thirty years ago computers were bulky monstrosities: expensive, power-hungry, slow, finicky. Viewed from today's perspective, the basic technology was inadequate. Computers were too slow for many tasks and too fragile for most environments. It would have been inconceivable to put such computers in a car, aboard an orbiting satellite, or on someone's desk. Yet, the breakthroughs came and today observers would agree that computer applications are no longer restricted by technology-hardware capability and availability as much as by the economics, architecture and ingenuity of the software implementations. At least as far as conventional data-processing (nonreal-tlme) applications are concerned, the hardware is fast enough, cheap enough and durable enough to go anywhere and to do almost anything that we arc willing to pay programmers to develop and debug.

Mobile radio today is in the same situation as the computer of the 1950's. Our goals for mobile communication, in terms of performance, cost, capacity, spectrum efficiency and portability, seem far beyond the reach of today's hardware. This, it is believed, will prove to be a very temporary state of affairs [5]. Imminent technical breakthroughs, some already unfolding, will completely change our thinking about mobile radio and trnnsfonn our sense of the possible. This study is hoped to contribute in a decisive way to this course.

2.1) ANALOG VS DIGITAL

Digital systems are increasingly favored because they enjoy certain general advantages over analog techniques, the most important of which have to do with the emerging plans for a digital network that will possess capabilities far beyond those of today's telephone and radio systems. A point wise comparison, advantages and disadvantages, of a digital system to an analog system will follow.

AnVANTAGFS OF DlOrrAL RADIO SYSTEMS:

- Digital equipment is transparent to the type of traffic which it carries, i.e. the traffic may originate from telephone, computer, facsimile, telex, ete., it may be integrated into one bit stream for transmission over the same radio bearer (or other transmission systems), and it may be switched together. As the data stream is always present, whether infonnation is contained in it or not, the load on the

radio system is therefore constant. Thls is not true for analog radio systems, where the loading depends upon the amount and the type of traffic being carried at anyone time;

- TIle accumulated noise of analog systems which posed serious constraints on equipment design is avoided in a digital system by regenerating the data stream;
- Digital radio systems are less prone to interference and can operate satisfactorily with a carrier-to-interference ratio of 15 to 30 dn. This permits the same frequency to be re-used on the orthogonal polarization, thus conserving bandwidth and effectively doubling the spectrum efficiency or doubling the capacity which the RF channel can carry. This feature is one of the best advantages of digital radio over analog, for in the analog case the requirement for carrier-to-interference ratio is much higher: approximately 45 to 60 dB [6];
- TIle output power of a digital radio transmitter can be less than that of the analog radio for a given transmission quality. This lowers the cost of the equipment, increase reliability and saves on power and air-conditioning costs. Smaller output power also has the advantage that smaller interference levels are produced, making it possible for higher co-channel frequency re-use within a given geographical area and coexistence of digital systems with existing analog systems;
- Depending on the modulation system in use, digital systems are able to produce a voice channel of acceptable quality with a carrier-to-noise ratio of as little as 15 dB, whereas for an analog system this may be designated as 30 dB or more, again depending upon the system. Figure 2.1 shows the comparison of the signal-to-noise ratio (SNR) versus input receive level *ireceivcr quieting curve)* of analog and a digital radio system. Under *flat fading* conditions, the analog system shows a gradual decline of SNR (dB for dB) for a decrease in received signal level. The digital system is unaffected until a threshold is reached. This characteristic is due to the regeneration process, in which the digital signal can be regenerated to its pristine fonn as long as the system is operated above a predetermined threshold. This threshold behavior thus permits a constant transmission quality to be achieved, which is independent of the received field variations, path length and the number of repeaters.

'Thesc points together with the fact that a digital radio network can be established by using existing towers and antennas mokc digital systems attractive. With the reduction in large scale integrated (LSI) circuit *costs,* and thc large scale production of digital integrated circuits, equipment costs arc reduced, making digital systems cost competitive with thnt of analog systems. Indeed the total cost for a digital system is at present gcnerally lower than

for an analog system.

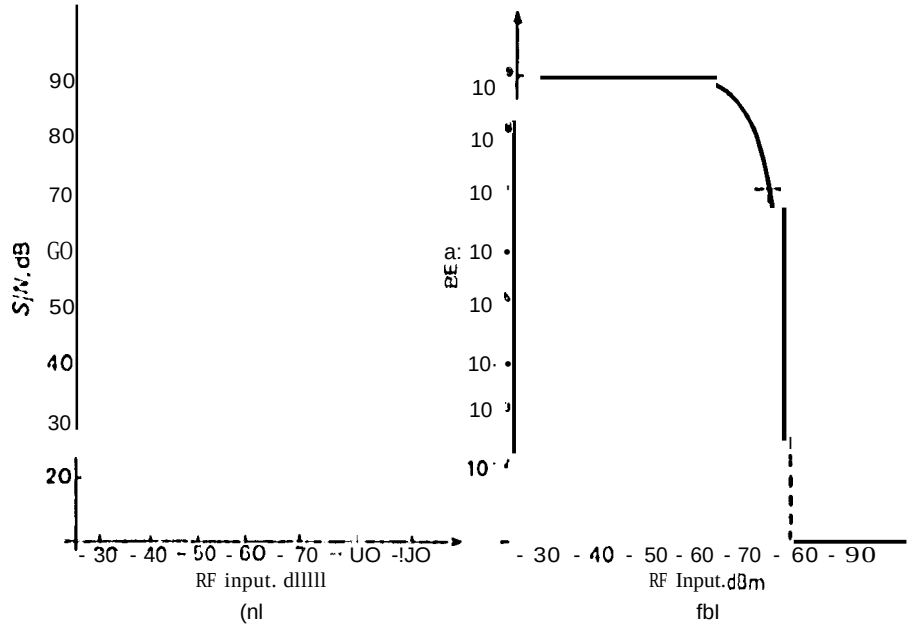


FIGURE 2.1

RECEIVER QUIETING CURVE: (a) ANALOG SYSTEM; (b) DIGITAL SYSTEM

DISADVANTAGE..5 OF DIGITAL RADIO SYSTEMS:

- Digital radio and digital systems have to be integrated in the early stages with an existing analog network. The associated interfacing problems, costs and interference problems make the situation difficult:
- Present digital radio systems are not as RF bandwidth efficient as analog systems of 600 voice channels or more. These problems will be resolved or not be that important, as the operating carrier frequencies are pushed higher;
- To carry the same traffic as an optical fiber, a digital radio requires to operate with a larger RF bandwidth. This large bandwidth makes the radio link susceptible to selective fading, which results in the consequent loss of all traffic. A wideband analog system by the same token would suffer only an increase in noise:
- The introduction of new and many varied techniques over those required in the design, testing and maintenance of analog systems poses a problem in the retaining of personnel and in the training of new engineering staff. There is in reality a threefold problem. The first aspect is that there is already a heavy capital investment in analog systems, which have proven to be reliable, Personnel arc thus still heavily committed in this area and will be for some years to come. Secondly, digital radio or systems require that new skills be learnt and acquired, which arc not easily accepted in general. Finally, transmission technology has strong competitors in the information technology and computer tcchnology areas,

which may be more immediotely appealing to prospective new engineers.

The decisive advantages of digital radio over analog are: economy, better interference immunity and better quality circuits over long distances.

It should be clear by now that the choice of analog or digital technology is a fundamental, irreversible decision that will define the next generation of cellular systems. Is there any rl'aSOI1 to consider analog technology for the next generation of mobile radio? Admittedly, I believe the answer is obvious. The next generation will be digital, for all the reasons presented. Since it is evident that digital is the future, this study will only consider digital radio systems.

2.2) DIGITAL SIGNALING OVER FADING MULTIPA1'1 CHANNELS

Since this study is by no means concerned with the modeling of the mobile communication channel, the contemplation of the mobile communication channel to follow can be considered strictly introductory, with only physical observations being discussed. However, for a more expanded and mathematical approach to the digital mobile radio channel [7], [8] and [9] can be approached.

RF channels having a randomly time-variant impulse response is named *lading mul/ipa/h channels* [10]. This characterization serves as a model for signal transmission over many radio channels, such as shortwave ionospheric radio communication in the 3 to 30 MHz frequency band (IIF), tropospheric scatter (beyond-the-horizon) radio communications in the 300 to 3000 Mllz frequency band (UHF) and 3 to 30 QBz frequency band (SHF) and ionospheric forward scatter in the 30 to 300 Mllz frequency band (VHF). The time-variant impulse response of these channels arc a consequence of the constantly changing physical characteristics of the media.

Radio waves may be propagated around the globe differently. The two main routes which they may travel from the transmitting antenna to the receiving antenna are either by the ionosphere (*sky wave*) or by hugging the ground (*ground wave*). The ground wave may itself he divided into two types; the surface wave and the space wave. For the space wave, three paths may be used to traverse the distance between the transmitted antenna and the receiving antenna. These arc the direct wave, the ground reflected wave and the tropospherically reflected wave.

In the space wave mode of propagation, which is used by the systcrn considered in this

study, the wave travels in the troposphere, which extends to 16 kilometers above the Earth's surface. The wave energy travels from the transmitting to receiving antenna either in a straight line (tine of sight) or is reflected at the ground or from the troposphere. The space wave is the one which is of importance in VHF, UHF and SHF communications.

Several things happen to n radio wave transmitted to or from a moving vehicle [1], [5] and [11]:

2.2.1) FREE SPACE LOSS

First, the *free-space basic transmission loss* (A_0) is the transmission that would occur if the antennas were replaced by isotropic antennas located in a perfect dielectric, homogeneous, isotropic and unlimited environment, the distance between the antennas being retained. If the distance l between the antennas is much greater than the wavelength λ, the free-space transmission loss can be written as:

$$A_0 = 20 \log((4\pi l/\lambda).(E_0/E)) \text{ dB} \tag{2.1}$$

As the signal is radiated in all directions, the power of the signal at any given point steadily diminishes as the inverse of the square of the distance between the receiver and transmitter; thc famous *inverse square law.* Since mobile systems are not set up in outer space, the path loss is more severe than the inverse square law would predict. Path loss for mobile systcms *may* be evaluated as the inverse of the *cube* of the distance, because of the additional doppler shift, sec section 2.2.3.3, induced on the signal.

2.2.2) ATTENUATION

111C second obstacle facing the radio wave in the transmission path is the possibility that it may be partially blocked, or absorbed by some feature of the environment. Propagation is therefore mainly by means of scattering and multiple reflections from the sunounding obstacles, as shown in figure 2.2. The degree of attenuation and the specific factors that may cause attenuation depend chiefly upon the frequcncy.

For example, frequencies below I Gllz are essentially unaffected by rain or atmospheric moisture. In general the lower frequencies have much greater penetrating power and will propagate farther. The higher the frequency, the greater the attenuation, the more power needed at the trnnsmiuer and the shorter the radius of effective transmission. At typical mobile-radio frequencies (150·900 MHz), the most important environmental attenuation

effect is *shadowing*, where buildings or hills create radio shadows. As can be expected,

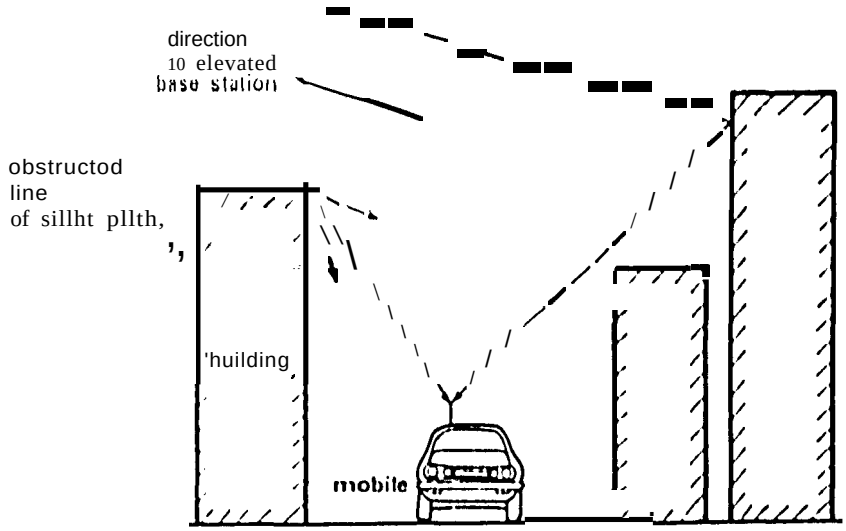


FIGURE 2.2
ILLUSTRATION OF RADIO PROPAGATION IN URBAN **AREAS**

shadowing arc most severe in heavily built-up urban centers. Shadows as deep as 20 dB over very short distances can be found, often literally from one street to the next. depending upon orientation to the transmitter and local building patterns. The fading effects produced by shadowing arc often referred to as *slow fading,* because. from the perspective of a moving automobile the entrance and exit to and from such a shadow takes a fair amount of time, since the area of the fade is large.

2.2.3) MULTIPATJI

In all except very simple transmission conditions, radio communication links are subject to conditions in which energy travel from the trnnsmitter to the receiver via more than one path. TIle radio may be reflected; from a hill. a building, a truck, a discontinuity in the atmosphere, etc. '11e effect is to produce not one but many different paths between the transmitter and receiver. This is known as *mliltil'fllIt propagation*; it is the two-edged sworn of mobile radio, Multipath propagation creates some of the most difficult problems associated with the mobile environment. The three most important rnultipath issues for the digital designer arc:

- The delay spread of the signal:
- Random phase shift which creates rapid fluctuations in signal strength known as

phase with the direct-path signal, If two signals are exactly 180° out of phase, they will cancel each other out at the receiver. TIle signal effectively disappears. Other partia! out-of-phase relationships among multiple received signals produce lesser reductions in measured signal strength.

Assuming the transmitter is stntionary, at any given spot occupied by the receiver the sum of all direct and reflected paths from a transmltter to a receiver produces an alteration in signal strength related to the degree to which the multipath signal are in phase or out of phase. This signal strength may be somewhat more, or considerably less, than the expected signal strength, which can be defined as that which would be expected on the basis of the direct path alone, based solely on Iree-space loss and environmental attenuation. If the actual measured signal is significantly weaker, say 20 dB or 100 times weaker, than the expected signal level, we mny conceive of that spot as a 20-dB fade, *for that frequency and for that precise transmitter locatlon and precisse configuratton of reflectors.* As long as we hold these factors constant, if we place our antenna in this spot we will lose 20 dB of signal strength.

What can we sny about the number, the spacing and the depth of these fades? There has developed a body of statistlcal knowledge which can be used with some success to characterize the incidence of fades in the environment.

The fades are said to fall within a statistical distribution known as Rayleigh distribution (after Lord Rayleigh, the great tum-of-the-century English physicist) and for this reason the phenomenon is often referred to as Rayleigh fading [lO]. The mobile environment is often called, from this perspective, the Rayleigh environment.

The Rayleigh environment is peppered with fades of varying depths, These fades are very deep; the signal strength is reduced by 10 000 to 100 000 times down from its expected value. In between are thousands of shallower fades. Now imagine an automobile antenna moving through this strange Swiss-cheese radio world at 100 kmlhr. The antenna passes through hundreds of holes of varying depths every second, causing the signnl strength to fluctuate very rapidly between normal levels and fades ranging up 10 40 dB or more. An amplitude monitor on a mobile receiver will draw a graph like figure 2.4. This is the way Rayleigh fading is usually experienced and por1rnyed.

These signnl amplitude Iluctuations constitute by fnr the most difficult challenge of Ihe mobile environment. Rayleigh fading is the dominant design challenge for any digital

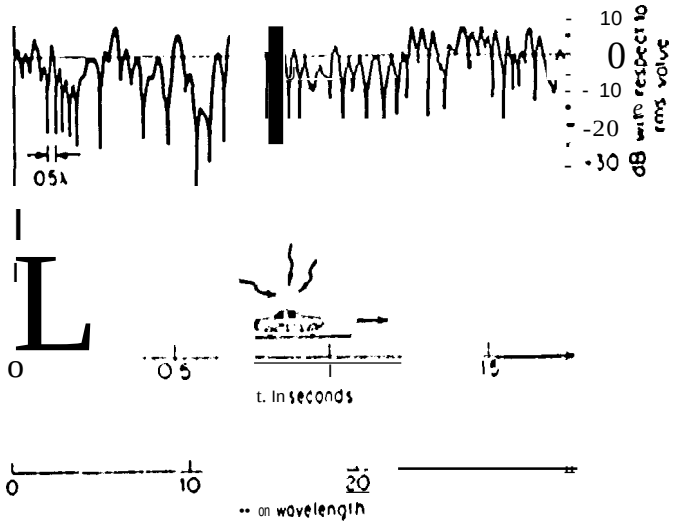


FIGURE 2.4
FADING SIGNAL RECEIVED WHILE 11IE MOBILE UNIT IS MOVING

mobile-radio proposal. It is a characteristic of the environment and cannot be altered by the mobile systems engineer.

Cellular radio today is the final flower of *frequency modulated* (FM) analog radio. Multipath fading is the great destroyer of mobile-radio signals; it overwhelms AM and single sideband systems. The great advantage of FM when it was applied to mobile radio in the 1930's was due to the fact that the FM receiver suppressed (ignored) amplitude modulation. and thus the degradation due to amplitude fading is greatly reduced.

2.2.3.3) DOPPLER SHIFT

Whenever relative motion exists there is a shift in the received signal. this being a manifestation in the frequency domain of the envelope fading in the time domain. This is the varialion in the frequency of the received signal known as the *Doppler shift* [to] (after Chrisrhn Johann Doppler, a nineteenth-century Austrian physicist who first called attention to frequency shifts caused by relative motion.) Much as the sound of a hom on a moving car appears to the stationary observer to be slighlly higher in pitch when the car is approaching rapidly and slightly lower when the car is receding, so radio transmissions arc frequency shifted due to the relative motion of the vehicle. This frequency shift varies considerably as the mobile unit changes direction, speed and it introduces considerably random frequency ITlodulation in the mobile signal. Moreover, the Doppler shift affects all

multiple propagation paths, some of which may exhibit 8 positive shift and some a negative shift, at the same instant.

2.4) Tne CELLULAR CONCEPT

It is important to recognize that today's analog cellular radio is not so much a new technology as a new idea for organizing existing technology on a larger scale. The critical innovation was the "cellular idea". Cellular represented a very different approach to structuring a radio-telephone network. It was an idea that held out the fantastic promise of virtually unlimited system capacity, breaking through the barriers that had restricted the growth of mobile telephony and it did so *without* any fundamental technological leap forward; simply through working smarter with the same resources. Cellular architecture was a system-level concept, essentially independent of radio technology. It appealed to mobile system engineers, because it kept them on relative familiar hardware ground. It appealed to businessmen and entrepreneurs, because it seemed to open the path to a really large market: by the application of the cellular idea, mobile communication could become another first-class growth industry, like television or radio, or telephone itself. The cellular idea also appealed to the authorities, because it seemed to break out of the spectrum shortage that had created terrific political difficulties.

111e cellular idea is elusively simple; one writer concluded that it seemed *"to have materialized from nowhere"* [I]. It began to appear in Bell System proposals during the late 1940's. It had occurred to people that the problem of spectrum congestion might be alleviated by restructuring the coverage areas of mobile radio systems. The traditional approach to mobile radio viewed the problem in terms similar to radio or television broadcasting; it involved selling up a high-power transmitter on top of the highest point in the area and blasting out the signal to the horizon (as much as 70 to 80 kilometers away.) It meant that the few available channels were locked up over a large area by a small number of calls.

111e cellular idea approached the coverage problem quite differently. It abandoned the broadcasting model. Cellular called instead for *low-power* transmitters, lots of them, each specifically designed to serve only a small area, perhaps only a few kilometers across.

By reducing the coverage areas and creating a large number of small cells, it became possible to *re-use* the some frequencies in different small coverage areas, called *cetls.* For a graphic description of a "traditional" vs cellular transmitting system refer to figure 2.5. To

understand how this changes total picture. imagine that all the available frequencies could be reused in every cell. If this can be done, then instead of 12 simultaneous telephone circuits for the entire city there would be 12 circuits for every cell. If there are 100 cells (each about 16 kilometers across), there would be 1200 circuits for the city, instead of only 12.

nut. it is not quite as neat. Early calculations indicated that, because of interference between mobiles operating on the same channel in adjacent cells, the same frequency could not be used in every cell. It would be necessary to skip severnI cells before reusing the

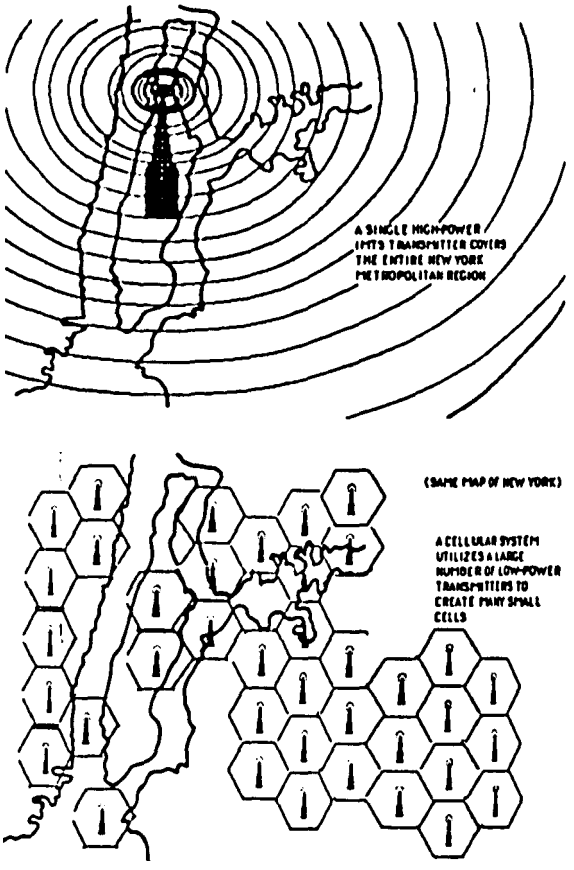


FIGURE 2.5
CELLULAR ARCHITECTURE

same frequencies, sec figure 2.6. nut the basic idea of reuse appears to be valid.

Moreover -and here lay the real power of the cellular idea- it appears that the effects of interference arc not related to absolute distance between cells, but to the *ratio of the*

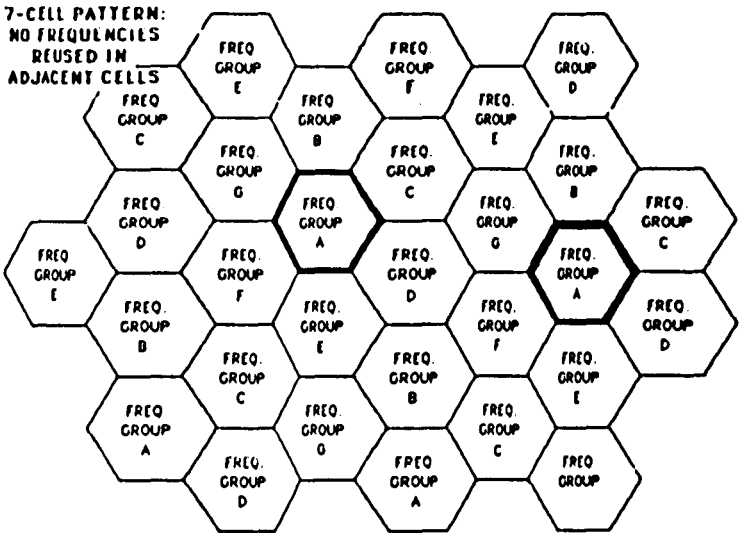


FIGURE 2.6
FREQUENCY REUSE

distance between cells to the radius 01 the cells. The cell radius is determined by the transmitter power; in other words, it is under the system engineer's control. If. for example. a grid of 16 kilometer radius cells allowed reuse of the frequencies in cell A at a distance of 48 kilometers, thcn a grid of 8 kilometer radius cells would allow reuse at 24 kilometers and 2 kilometer radius cells would allow reuse at 6 kilometers. Because of the fabulous cellular geometry (based on the πr^2 rule for cell coverage area), however, each reduction in cell radius by 50% led to a quadrupling of the number of circuits per megahertz per square kilometer. A system based on 1 kilometer radius ceJls would generate one hundred times as many circuits as a system based on 10 kilometer radius cells.

Of course. it would have been enonnously expensive to build thousand-cell systems right from the beginning. It appears, however, that large-radius cells could evolve gracefully into small-radius cells over a period of time through a technique called *cell-splitting.* When the traffic reaches the point in a particular cell stich that the existing allocation of channels in that cell could no longer support a good grade of service, that cell would be subdivided into a number of new cells, with even lower transmitter powers, fitting within the area of the former cell. 111C reuse pattern of capacity multiplied for that area by a factor equal to the number of new cells, When. in time, the smaller cells are saturated, still smaller cells could be created.

Cell-splitting offers many advantages. It allows the financial investment to be spread out

as the system grow. New cells would only be added as the number of revenue-generating customers increases. Moreover. cell-splitting can be applied in a geographically selective manner: the expense of smaller cells would only be necessary in the high density traffic centers. In the minds of cellular architects. it is the ideal surgical technique for boosting capacity precisely where and when it is needed,

Since the mobile user is liable to wander out of one cell and into another as the call progresses. the system must reroute the call to the new base station and switch the call to a new radio channel without interruption. This procedure, known as *"and-off,* is one of the distinguishing features of cellular systems.

2.5) PRESENT CODING TEellNIQUES

In real-thne speech requirements. the much-used protocol for data packets known as ARQ [5]. or *automatic retransmission request.* cannot be applied. An ARQ system needs only to utilize a sufficient amount of coding to enable the receiver to detect errors in a packet and initiate a retransmission request if errors are found. Real-time speech. however. does not allow for this procedure.

Two types of coding arc widely used for structuring the code words. The most commonly used is the linear error correcting block code and in particular the *Bosc-Chaudhuri-Hocqucnghem* (BClI) variant. An alternative technique is offered by convolutional coding. of which the best known type is the *Reed-Solomon* code (the Reed-Solomon code can also be implemented as a block code). Insofar as generalization are possible. Reed-Solomon codes perform well in narrowband channels where burst errors predominate and nCIl codes work well where uniformly distributed statistically-independent errors are encountered [I].

In some commercial applications the baseband data streams in both directions arc encoded such that each non-rerum-to-zero (NRZ) binary I becomes a O-to-I transition and each NRZ zero becomes a one-to-zero trnnsition. This example of a modulation code has been considered on multipath (ading channels and is known as the $(d, k, C) = (0. I, I)$ Manchester code; a strategy which assists the receiving end in recovering the basic data clock. See chapter 8. p257 in Parsons [5] (or a full description.

'111C choice of modulation/coding techniques (or a mobile radio-telephone system is driven by several technical considerations. chief among which are:

- The scarcity of bandwidth, leading to a need for spectral efficiency;
- 111e problem of adjacent-channel interference, lending to the requirement for narrow-power spectra;
- 111e problem of intersymbol interference, which imposes hard limits on the transmission rate in a mobile environment;
- The problem of clock extraction, synchronization is lost if there are too many successive symbols without transitions.

II should be kept in mind that all four parameters are interrelated; in the following discussion, for the purpose of clarification of the underlying issues, some conceptual liberties are taken in treating each of the four constraints separately.

2.5.1) SPECTRAL EFFICIENCY

Spectral efficiency refers to the number of bits that are transmltted in a given period of time, usually one second, over a radio channel with a defined bandwidth. Since the channel bandwidth is measured in kilohertz (kl'lz) or megahertz (Mllz), it is possible to define spectral efficiency as the number of bits per second per hertz (Hz), sometimes loosely referred to as bits per hertz (b/IIz). ntis is also commonly called *information density;* how many bits can be pumped through a given channel in one second.

l11C concept of multi-ary modulation is very powerful. (It is assumed that the reader is familiar with the principles of multi-ary coding.) It is certainly possible to conceive of an a-ary system, which would transmit three bits per symbol, achieving 3 b/hz. A 16-ary modulator, encoding four bits per symbol, would achieve 4 b/Hz. Multi-ary coding is a logical concept that may be applied to any of the familiar modulation schemes.

Whilc spectral efficiency increases arithmetically, the number of levels nnd the precision required at the demodulator increase exponentially. If n is the number of bits per symbol, then the number of levels equals 2^n, which also correlates with the degree of precision required in the demodulator, The diffcrence in signal level between 4-ary PSK nnd 16-ary PSK is about 13 dB; the signal-to-nolse ratio must be about 200 times bener for 16-nry PSK to equal the performance or 4-ary PSK. This trnnslatcs into higher power requirements, reduced range and at some point into absolute limits on the nhilily or higher-nry modulotion schemes to function. 111C mobile environment is particularly severe nnd mnny observers today (11) doubt whether modulators much above 16 levels or so will ever he made to work well for mobile radio. Most field work has favored either 2-ary,

-t-ary or *B-ary,* for robustness. A quantative confinnation of this statement is presented in chapter 9.

2.5.2) NARROW POWER SPECTRUM

When a modulated radio carrier wave is transmitted, the energy it contains is distributed in a characteristic fashion about the center frequency (See figure 2.7). The distribution is known as the *power sipcctrum.* TIle father away from the center frequency in either direction, the less strong the signal. Typically, the energy is concentrated in a *main band.* Some forms of modulation and coding, however, produce significant *sidebands.* In fact, the particular "signature" of the power spectrum, especially the size of the sidclobes, is one of the most important factors for distinguishing among different modulation and coding proposals.

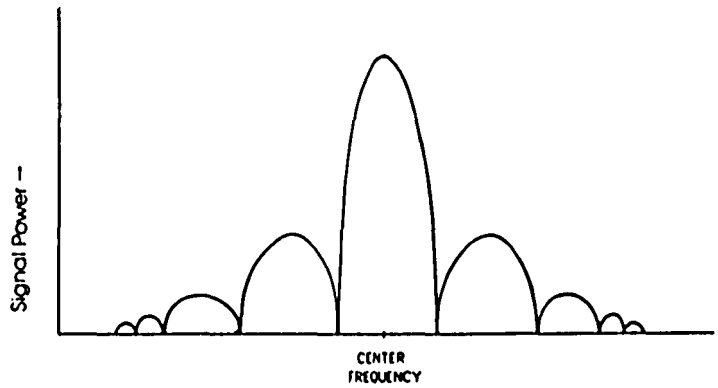


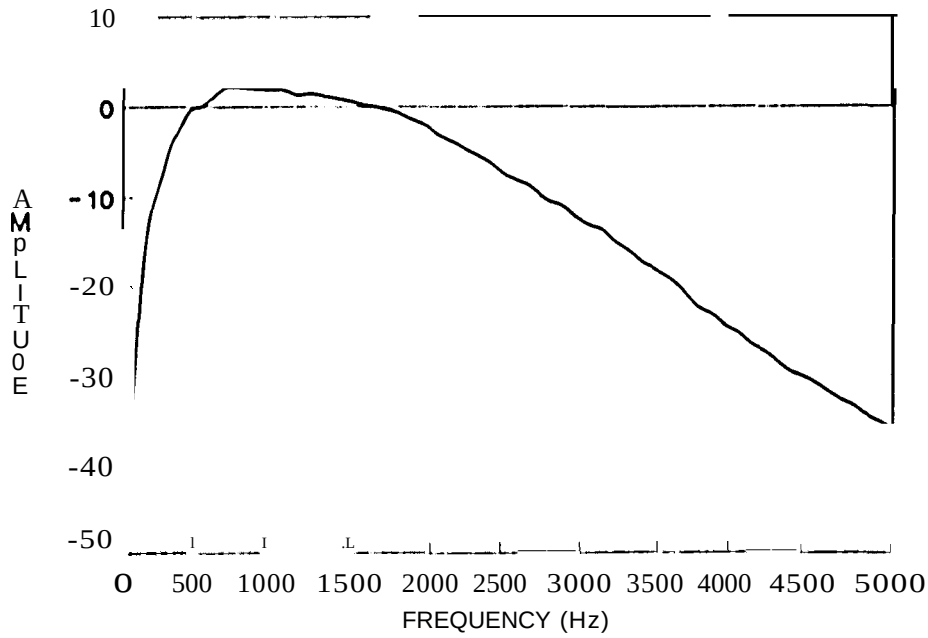
FIGURE 2.7
TYPICAL POWER SPECTRUM

The power spectrum is a determinant of adjacent-channel interference. A modulator with a very broad power spectrum, like conventional FM, will overlap significantly with adjacent transmissions. A broad power spectrum is therefore not desirable. It can be filtered to fit the mask, but such filtering can add considerable expense to the mobile unit.

Another method to ensure a power spectra of desirable shape, is to look at the transfer function of the given channel and shape the data spectrum in such a way as to fit the channel response in the best possible way. The measured transfer function of a mobile communication channel is presented in figure 2.8. It is clear the channel is more lenient on the low frequency side. A code an(Vor modulation scheme which could shape the power spectra in such a fashion as to fit the transfer function of the channel would he a better choice over one that have for instance a peak at the high frequency side of the spectra.

2.5.3) INTERSYMBOL INTERFERENCE

As discussed earlier, one of the effects produced by the mobile environment is the *delay spread.* Depending upon the nature of the environmental reflectors that create multipath transmission. the speed of the mobile unit, and other factors, a sharp transmitted pulse of. say. a fifth of a microsecond duration will be detected by the receiver as a smeared and flattened bulge of considerably greater duration. sometimes up to several microseconds. If



AMPLITUDE IN dB

FIGURE 2.8

MEASURED TRANSFER F'UNcnON OF MOBILE COMMUNICATION CHANNEL

it is severe enough -that is. if the transmission-induced delay spread is large relative to the average symbol time, intersymbol interference will result as the individual symbols begin to overlap one another.

Delay spread is produced by the environment; for a given frequency and a given environment the delay spread should be the same (or all radio signals propagating in that environment. To some extent this can be controlled by adaptive equalization. Another method to reduce intersymbol interference is to make usc of nanlength-Iimited coding on thc digital sequence. In doing this thcre will always be a minimum of *d* (Sec chapter 4) symbols without transitions; restraining the effects of intersyrnbol interference to a certain degree. depending on *d.* Naturally, thc most effective way to reduce intersymbol intcrference will be to utilize both these techniques.

2.5.4) CLOCK EXTRACTION

It must be noted that *coding* in this study is by no means the same as used in present mobile communication papers and handbooks. Under coding is understood shaping of the power spectra and inserting some kind of unique characteristic in the digital channel bits, such as rigidity against intersymbol interference and clock recovery properties.

Mobile communication literature view coding as error correction coding, where channel injected errors arc corrected, or digitizing the analog speech signal using various techniques to reduce the number of bits that need to be transmitted, while maintaining "telephone voice quality." Some of the techniques used are:

- Continuously variable slope delta modulation (CVSD);
- Adaptive subband coding (SBC);
  Residual-excited linear predictive coding (RELP);
  Vector quantization.

For a detailed discussion on these techniques [1] and [11] can be consulted.

When digital data is transmitted over a channel the receiver must be able to recover the clock from the received data. If the channel bits do not have enough transitions, the receiver can, in worst case conditions, loose synchronization and extra errors can occur because of this. By the effective use of coding the data stream can be altered in such a way as to ensure regular transitions.

A further advantage of regular transitions can be appreciated when the signal enters a deep fade and clock extraction is impossible. When the signal recovers from the fade, clock extraction can be regained from the next 10 to 20 bits in a coded scheme, while, in an uncoded scheme, clock extraction may only be regained from the next 100 to 200 bits.

111is so called "modulation coding" or "line coding" will be the focus of this investigation; trying to reduce the aforementioned intersymbol interference and ensuring regular transitions in thc digital data stream.

2.6) CELLULAR RADIO IN THE Rsrunuc OF South AFRICA

The first generation of mobile telephone systems, such as the system installed in November

1981 by the SAPT in the Pretoria-Witwatersrand area, were not suitable for the expected growth in mobile communications: it did not utilize the cellular concept. In May 1986 a cellular mobile telephone network was introduced by the SAPT, using the C450 public land-mobile system from Siemens [t 2].

The C450 system operates in the 450 MJ Iz band. Jt is designed for the reuse of frequencies in every seventh cell, thereby fonning a seven-cell "cluster". A unique feature of the C450 system is that hand-offs are not just determined by field strength distribution and signal quality as in most other systems, but by actual determination of the stations in a cluster. This is accomplished by synchronizing all the base stations and therefore evaluating the relative delay times of the signals from the control channels of the surrounding bose stations in the mobile set itself. 'I'hc mobile set also retransmits the distance data with a fixed delay. By using various algorithms the position of the mobile user can be detennined to a resolution of 400 meters. 111is allows accurate detennination and flexibility of cell boundaries.

A unique feature of the mobile sets is the use of a personal identification card for each subscriber, similar in size to a credit card. There is no identity allocated to a mobile set (MS). Each MS has a card reader that will read the magnetic code on the subscriber's personal identification card and the MS will then assume the identity of a mobile set with the specified dircctory number. This allows the driver of a rented car to make/receive calls in his rented car using his own directory number.

There arc seven base stations, comprising one seven-cell cluster. Base stations are located at Strydom Tower (Johannesburg), Langerand (Vereenigingl, Springs, Benoni. Doornkloof, Lewisham (Krugersdorp) and John Vorster Tower (Pretoria).

The ultimate theoretical capacity with the existing equipment is approximately 6500 subscribers. If there is further growth the existing cell boundaries would have to redesigned for the addition of new cells. Ilowever due to frequency limitations the theoretical capacity is presently only about 2500 subscribers, At present there are approximately 600 mobile telephone users connected to the C450 system nnd these are still growing steadily.

Although the present system is adequate, the restricted radio spectrum in the 450 Mllz band and lower where this system and other existing mobile radio systems work has to be overcome. Future systems will usc digital technology and generally usc the 900 MHz band where adequate frequencies are still available.

Since the capital investment involved is considerable (in excess of RIO million for the C450 system installed in the PWV area in 1986), the South African mobile telephone system will only be expanded to the other major urban centers such as Cape Town and Durban, once the PWV system has proven to be commercially viable.

CHAPTER 3

THEORY OF FINITE-STATE MACHINES

TIle study of finite-state machines (FSM) in this chapter is concerned with describing their structure, analyzing their capabilities and limitations and investigating various fonns in which they can be realized physically. The significance of such machines is that these models are not confined to any particular scientific area, but arc directly applicable to problems in practically every field of investigation - from psychology to business administration, from communication to computer science. Since the human brain operates in a sequential manner, even the thinking process and the analysis of the English syntax can be employed with a FSM.

FSM's arc also used extensively in communication engineering, hence a solid background on this topic is essential. The discussion to follow will give a general background on FSM behavior, following [13], [14], and implementations on communication systems will be considered in chapters to follow.

3.1) THE FINITE-STATE Monm.

Most problems encountered in engineering investigations can be classified as *analysis* problems or *synthesis* problems. Analysis problems usually involve prediction of the

behavior of a system. and synthesis problems where one wishes to construct a system with a specified behavior. In this chapter the analysis part of finite-state machines will be considered, where as in chapters to follow. some synthesis problems will be encountered. In both analysis lind synthesis problems three groups of variables are encountered which characterize a system. namely *excitation variables,* which represents the stimuli generated by systems othcr than the one under investigntion, and which Influence the behavior of the system under investigation; *response variabtcs,* representing those aspects of system behavior which arc of intcrest to the invcstlgator: *tntermcdkue variables,* whose Importance does not lie in their individual behaviors, but rather in their combined effcct on the relationship between the input and output variables.

A finite-state machine is an abstract model consisting of a finite set of *input symbols* representing the excitation variables, a finite set of *output symbol:* representing the response variables. a finite set of *states* representing the intermediate variahies, a *next-state function* and an *output function,* '111e intermediate variables, which arc of no direct interest. are assumed to be embedded inside the system. 111e sets of input and output symbols arc usually referred to as the input and output *alphabets.*

Input, output and state variables nrc defined only for integral values of time. Every system representable by the basic finite-state model is assumed to be controlled by an independent *synchronizing source,* in the following fashion: TIle system variables are not measured continuously. but only at the discrete instants of time nt which a certain specified event, called a *synchronizing signal.* is exhibited by the source. These instants of time are called *sampling limes,* the 11h sampling time being denoted by t_v (v = l, 2....). An additional assumption is that the behavior of the system at nny sampling time t_v is independent of the interval between t_v and the previous sampling time t_{v-1}. Thus, the true independent quantity, against which every system variable is measured, is not time, but the ordinal number associated with the sampling times. As will be seen. synchronization is most important in communication systems,

It should be emphasized that the foregoing assumptions do not imply that the time intervals between two successive synchroniaing signals arc uniform, neither does it imply that a system variable, within such an interval. exhibits some specific mode of behavior. The only implication is that, whntever the interval is and whatever the system variations within the interval are, the values of the variables at the rth sampling time depend on the number v and not on the value of t_v. Systems which conform with the time-discreteness assumption llrc said to be *synchronous.* Systems in which this assumption is not valid arc

called *asynchronous* systems. Such systems will not be discussed.

3.1.1) THE BASIC MODEL

An exact definition for the class of systems which we shall call finite-state machines can now be provided.

DEFINITION 3.1

A finite-state machine M is a synchronous system with a finite input alphabet $X = \{\xi_1, \xi_2, \ldots, \xi_p\}$, a finite output alphabet $Z = \{\zeta_1, \zeta_2, \ldots, \zeta_q\}$, a finite state set $S = \{\sigma_1, \sigma_2, \ldots, \sigma_n\}$, and a pair of characterizing functions f_z and f_s given by

$$z_v = f_z(x_v, s_v) \tag{3.1}$$

$$s_{v+1} = f_s(x_v, s_v) \tag{3.2}$$

where x_v, z_v and s_v arc, respectively the input symbol, output symbol and state of M at time t_v ($v = 1, 2, \ldots$). Throughout, the assumption will be that M, as postulated in definition 3.1, is *deterministic*, i.e., its characterizing functions are not subject to any uncertainty and that M is *nonrestricted*, i.e., any input symbol can be applied to M at any time t_v.

A special case of finite-state machine arises when

$$f_z(x_v, s_v) = f_z(x_v) \tag{3.3}$$

Such a machine is called a *trivial* machine. The intermediate variables in a trivial machine have no effect on its input-output relationship and hence the concept of state in this case is redundant. A *nontrivial* machine is one in which

$$f_z(x_v, s_v) \neq f_z(x_v) \tag{3.4}$$

When a machine is considered that has just one input symbol, there is never really any choice as to what symbol to apply to such a machine at a given time instant. The machine operates without any influence from the outside world. A machine of this type is called an *autonomous* machine. Since there is only one input symbol, we do not ordinarily bother to give it a name. (Some people like to think of an autonomous machine as having no inputs

at all. since its transitions are not under external control. TIle viewpoint taken is largely academlc.)

Machines whose characterizing functions arc the same except for possible differences in state labeling are said to be *isomorphic* to each other. Given a finite-state machine M representing a certain system. any machine which is isomorphic to M may serve as a representation of the same system. The representation of a system by a flnite-state machine is. by no means unique. A good example of two isomorphic machines is the Mealy and Moore representation of a given machine [13]. A machine whose characterizing function,!.. is solely a function of the state that is entered. is called a *Moore*-type machine. In a *Mealy* machine however.j'. is also a function of the input alphabet. Conversion from the one to the other will be discussed in chapter 5. together with some more synthesis problems concerning finite-state machines. A program listing in Turbo C can be found in Appendix E to convert between the two types of machines.

It is important to remember that the finite-state model is an abstract model and. as such, says nothing about a physical realization of the process it describes. The states and symbols need not be thought of as having physical representations in terms of voltages or currents. level or pulses etc. All the physical phenomena of a sequential or iterative realization have been replaced by the next-state and output functions. The value of such an abstraction is that it enables us to strip away the unimportant physical details and study the common properties of a variety of different processes.

3.2) PREDlcnNG MACHINE BEHAVIOR

A succession of input symbols ie. ξ_{i1}, followed by ξ_{i2}..... followed by ξ_{il}, is called an *input sequence* and written as $\xi_{i1}\xi_{i2}\cdot\ldots\cdot\xi_{il}$. A succession of output symbols ie. ζ_{j1}, followed by ζ_{j2}, followed ζ_{jl}, is called an *output sequence* and is written as $\zeta_{j1}\zeta_{j2}\ldots\cdot\zeta_{jl}$. The number of symbols in a sequence is referred to as the *length* of the sequence. As is evident from the time-discreteness assumption. excitations applied to finite-state machines are always in the fonn of input sequences and responses are always in the form of output sequences; an input sequence of length *l* always results in an output sequence of length *l*.

The state of machine M at time t_l is called the *initial* state of M. Since t_l is arbitrary, the initial state of M is commonly token as the state in which M is found whcn first presented to the invcstigator. The state of machine M at timc t_v is called the final state of machine M.

THEOREM 3••

Let M be a nontrivial machine with characterizing functions $!_I$ and f_S. Then the response of M. at any initial state $(1_I$, to any input sequence $\xi_{i1}\xi_{i2} \cdots \xi_{iu}$:

- is not predictable if only $!_I$ and f_S are known;
- is predictable if I_I, I_S and σ_i are known.

The proof of this theorem can be found in [13]. The functions $!_t$ and f_S of a FSM arc analogous to the equilibrium equations characterizing a linear device. with the initial state of the machine analogous to the initial energy distribution in the linear device. The response of the device to any given excitation can be predicted whcn both the equilibrium equation and the initial energy distribution nrc known. but it is unpredictable when the initial energy distribution is not specified.

3.3) TRANSITION TAIlLES. DIAGRAMS AND MATRICES

Once all the variables of a system arc established. the system can be formalized by means of a table, a diagram or a matrix. The table, diagram or matrix are alternative fonns of displaying the characterizing functions of the FSM. Such a display is indispensable to any precise analysis or synthesis of a FSM and it will be used extensively.

3.3.1) Tne TRANSmON TABLES

The characterizing functions I_t and I_S can be displayed in a tabular form referred to as the *transition table*. This table lists the values of the two functions for all possible arguments, i.e.• for all possible ordered pairs (x_v, s_v). where x_v ranges over the input alphabet X and s_v over the state set S. The format of the transition table for a machine whose input alphabet is $\{\xi_1, \xi_2, \ldots, \xi_p\}$, output alphabet is $\{\zeta_1, \zeta_2, \ldots, \zeta_q\}$, and state set is $\{a., 02' \cdots \sigma_n\}$, is shown in table 3.1. TIle table is composed of two adjacent subtables, the 1_v *subtable* and the s_{v+1} subtable, which display I_t and I_S, respectively. The two subtahles have a common stub which lists all possible present states s_v: the column headings in both subtables arc the same and consist of all possible present input symbols xv' 111e rows, then, arc labelled $0., 2, \ldots, all'$ and the columns $\xi_1, \xi_2, \ldots, \xi_p$. The entry common to the row (1, and column ξ_j is $f_t(\xi_j, \sigma_i)$ in the z_v suhtahle and $is]_S(\xi_j, a_I)$ in the s_{v+J} subtable. 111C z_V and s_{v+J} entries arc seen to range over the output alphabet Z and the stale set S, respectively, or over any subset thereof. See table 3.1.

Under the assumption that f_t and I_S are the characterizing function of a deterministic.

nonrestricted machine, these functions must be uniquely defined for every ordered pair (x_v, s_v), where x_v ranges over X and s_v over S. Consequently, the z_v subtable must contain exactly one element of Z and the s_{v+1} subtable exactly one element of S at the intersection of every row and column.

	z_v	s_{v+1}
x_v / s_v	ξ_1 ξ_2 ... ξ_p	ξ_1 ξ_2 ... ξ_p
σ_1 σ_2 . . . σ_n	Entries selected from $\zeta_1, \zeta_2, \ldots, \zeta_q$	Entries selected from $\sigma_1, \sigma_2, \ldots, \sigma_n$

TABLE 3.1

GENERAL TRANSITION TABLE

EXAMPLE 3.1

To illustrate the variety of situations which lend themselves to representation by the basic finite-state model, an example is presented. The pertinent input alphabet X, output alphabet Z and an appropriate state set S is listed. The names of the states will be so chosen as to convey the system conditions which the states imply.

Given: An English text, composed of the 26 letters of the alphabet and spaces has to be scanned with the purpose of counting the number of words starting with *"un"* and ending with *"d"* (i.e, "understand", "united", etc.), For simplicity, a space will be designated by π and letters other than *d*, *n* and *u* by λ. For this machine:

X = $\{d, \pi, u, x, \lambda\}$;
Z = (Count, No count);
S = (New word, Wait for new word, Mark *u*, Mark *u-n*, Mark *u-n-di* relabeled as (1, 2, 3,4,5).

To illustrate the construction of a transition table, table 3.2 shows the transition table for the system described. The system is referred to as "machine At" and the states "New word". "Wnit for new word", "Mark *u"*, "Mark *u-n"*, "Mark *u-n-d"* arc relabeled as 1, 2, 3. 4 and 5 respectively. TIle table entries constitute the numerical counterpart to the verbal arguments justifying the choice of state set.

When the input is "π", the next state is "New Word", regardless of the present state. If the present state "Mark *u-n-d'* and the input is "π", the output is "Count"; under all other conditions the output is "No count". If the present state is "New word" and the input is *"u".* the next state is "Mark *u";* if the input is *"d",* lin" or "λ", the next state is "Wait for new word". If the present state is "Mark *u"* and the input is *"n".* the next state is "Mark *u-n";* if the input is *"d", "u"* or "λ", the next state is "Wait for new word". If the present state is "Mark *u·n"* or "Mark *u-n-d"* and the input is *"d",* the next state is "Mark *u-n-d";* if the input is *"n",* "UN or "λ... the next state is "Murk *ll-n".* If the state is "Wait for new word" and the input is other than "x", the state remains unchanged.

	z_v					s_{v+1}				
x_v / s_v	*d*	11	*U*	π	λ	*d*	*n*	*u*	π	λ
1	0	0	0	0	0	2	2	3	1	2
2	0	0	0	0	0	2	2	2	1	2
3	0	0	0	0	0	2	4	2	1	2
4	0	0	0	0	0	5	4	4	1	4
5	0	0	0	1	0	5	4	4	1	4

TABLE 3.2

MACHINE AI

3.3.2) Tns TRANSmON DIAGRAM

The transition diagram can he considered as a directed graph and is a structure composed of *vertices,* drawn as smnll circles and of edges or *oriented branches,* drawn as lines between pairs of vertices, with arrow signs pointing from one vertex to the other. A trnnsition diagram describing an n-state machine contains *n* vertices, each vertex representing a different state; the state represented by a vertex is identified hy the label attached to this

vertex. The oriented branches arc drown and labelled according to the following nile:

Let $X_J = \{\xi_1, \xi_2, \ldots, \xi_r\}$ be thc set of x_v values for which $f(x_v, \sigma_i) = \sigma_j$ and $f(\xi_h, \sigma_i) = \zeta_h$ for $h = 1, 2, \ldots, r$. If X_j is nonempty a branch is drawn from the vertex labeled σ_i to the vertex labeled σ_j; the arrow sign of this branch is pointed

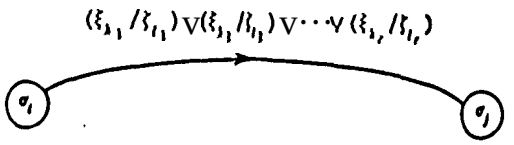


FIGURE 3.1
nRANCII LABELING.

from vertex a_i to vertex σ_j and thc branch labeled is written as $(\xi_1/\zeta_1) \vee (\xi_2/\zeta_2) \vee \ldots \vee (\xi_r/\zeta_r)$. The V is thc standard symbol for the logical "or". Each term (ξ_h/ζ_h) contained in a branch label is called an *input-output pair.* The above nile for constructing the transition diagram of a given machine is illustrated in figure 3.1.

This nile implies a one-to-one corrcspondcnce between a transition diagram and a transition table which represents the same machine. Thus, given the one representation. the other representation can always be constructed. As an example, figure 3.2 shows the transition diagram of machine Al specified by table 3.2.

By construction, a branch pointing from vertex σ_i to vertex a_J places in evidence the input symbols which cause the machine to pass from state σ_i into state a_J and the output symbols which accompany the passage. Since the machine is detcrministic and nonrestricted, every input symbol causes evcry state to pass into exactly one other state; consequently the branches originating from any given vertex nrc labeled with the total number of p input-output pairs. wherc p is the size of the input alphabet.

The immediate obvious advantage of the transition diagram over the transition table. is that it facllitates the determination of machine responses to input sequences of arbitrary lengths. Given the initial state σ_i of machine M and an input sequcnce $\xi_1, \xi_2, \ldots, \xi_l$, the response of M to $\xi_1, \xi_2, \ldots, \xi_l$ can be readily detcnnincd by tracing (in the arrow direction) the continuous sequence of l branches which originate at the vertex labelled σ_i and whose kth branch ($k = 1, 2, \ldots, l$) exhibits the input-output pair (ξ_k/ζ_k). The output sequence yielded by M when $\xi_1, \xi_2, \ldots, \xi_l$ is applied is then simply $\zeta_1, \zeta_2, \ldots, \zeta_l$; the state into which

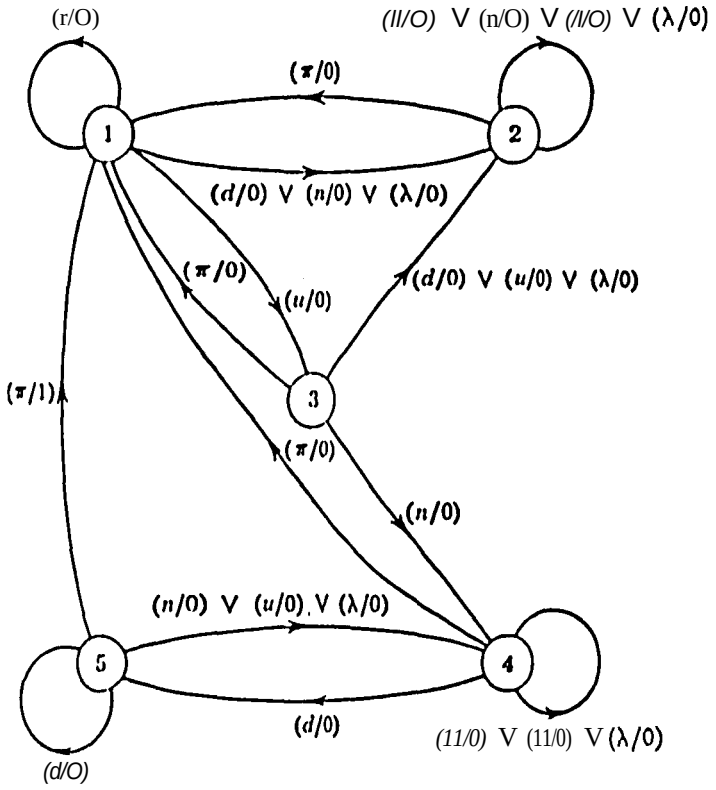


FIGURE 3.2

MAcmNEAI

M passes when $\xi_1, \xi_2, \ldots \xi_l$ is applied. is given by the label of the vertex at which the traced sequence of l branches terminates. For example. the response of machine At to the input sequence $\pi u n \lambda \lambda d \pi$ when the initial state is 3 is readily detennincd from figure 3.2 to be 0000001. The states traversed by machine Al when the above input sequence is applied are 1. 3. 4, 4. 4. 5 and 1. in that order. The role played by the transition diagram in the theory of finite-state machines is similar to that played by the circuit diagram in the theory of electric networks. TIle diagram transforms an abstract model into a physical picture which enhances the investigator's intuition and enables him to visualize various processes and properties which would otherwise remain a series of dry mathematical facts. As is the case in electric network theory, it is convenient to regard the diagram as the model itself and the symbols which appear in the diagram as the abstract components of which the model is composed.

3.3.3) Tne TRANsmON MATRIX

Tlic transition matrix is the mathematical counterpart of thc transition diagram: it enables one to carry out mechanically a number of operations which. in the transition diagram. can

be carried out visually. The transition matrix is, therefore. advantageous wherever the operations cannot be carried out by a human investigator. and hence cannot be carried out visually, or wherever the transition diagram is complex to the extend that visual approach is futile.

For each input-word ξ_v e X, we define:

- the *output matrix* Γ_v of dimension I x N;
- the *state transition matrix* E_v, which is an I x I binary matrix with the (i,j)th entry

$$E_v(i,j) = \begin{cases} 1. \text{ if } f_s(O_i \cdot \xi_v) = \sigma_i \\ 0, \text{ otherwise} \end{cases} \tag{3.5}$$

with I, N and K respectively the number of states, the number of output words and the number of input words.

The output and state transition matrices have the following characteristics:

- the matrices Γ_v. $v = 1 \ldots K$. specify and are completely specified by the output function f_i;
- the matrices E_v • $v = 1 \ldots K$, specify and are completely specified by the state transition function f_s.

From (3.5). $E_v(i.j) = 1$ if and only if the input-word ξ_v forces the FSM to pass from O_i to a_j. In particular. this implies that each row of E_v has one and only one non-zero entry.

For an application of the transition matrix approach. see chapter 5.

3.4) CLASSIFICATION OF STATES

A branch connected to any given state, say oj' may be a *converging branch* of $O1'$ if it points toward σ_i from another state. or a *diverging branch* of σ_i• if it points from O_f toward another state, or a *reflecting branch* of σ_i, if it loops around σ_i. Figure 3.3 illustrates these three types of branches.

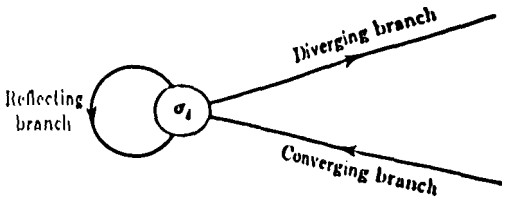


FIGURE 3.3
TYPES OF nRANClmS

A stale which lacks converging and/or diverging branches may be one of the following:

- A *transient state*; a state that has no converging branches but at least one diverging branch. Such a state can lead into at least one other state, but cannot be reached once it is abandoned;
- A *persistent state;* a state that has no diverging branches, but at least one converging branch. Such a state can be reached from at least one other state, but cannot be abandoned once it is reached;
- An *isolated state;* a state that has neither converging nor diverging branches. Such a state cannot lead into any other state and cannot be reached from any other state.

3.5) EQUIVALENCE AND MINIMIZATION

It was emphasized that the states of a finite-state machine need not be observable or even physical quantities and that their only function is to assist in the formulation of the input-output relationships of the machine. Consequently, any state set which fulfills this function is a satisfactory set, regardless of whether the states convey any intuitive meaning or not. This freedom inherent in the choice of a state set is quite advantageous. since it permits the replacement of one set with another set which may be considered more convenient for various purposes. More specifically, it permits one to carry out operations using a state set which is optimal or minimal in one sense or the other. It will become apparent that this concept not only paves the way for more precise and more concise formulation of finite-state machines, but sheds new light on the entire prohlem of machine analysis as well as synthesis.

3.5.1) STATE EQUIVALENCE

The notation M/σ will he used as an abbreviation for the phrase "machine M in state σ".

All proofs of theorems and lemmas can be found in [13].

DEFINITION 3.2

State σ_i of machine M_1 and state σ_j of machine M_2 are said to be *equivalent*, if M_1/σ_i and M_2/σ_j, when excited by any input sequence of possible infinite length, yield identical output sequences. If σ_i and σ_j are not equivalent, they are said to be *distinguishable*, M_1 and M_2 may refer to the same machine.

Illustrations, σ_i and σ_j are equivalent *iff* there is no way of distinguishing, by observing the external terminals, between machine M_1 at the initial state σ_i and machine M_2 at the initial state σ_j. Equivalence between σ_i and σ_j is indicated by $\sigma_i = \sigma_j$ and distinguishability between σ_i and σ_j is indicated by $\sigma_i \neq \sigma_j$. From definition 3.2 it can be readily verified that state equivalence obeys the reflexive law ($\sigma_i = \sigma_i$, $\sigma_j = \sigma_j$), the symmetric law (if $\sigma_i = \sigma_j$, then $\sigma_j = \sigma_i$) and the transitive law (if $\sigma_i = \sigma_j$ and $\sigma_j = \sigma_k$, then $\sigma_i = \sigma_k$). Consequently, state equivalence can be treated as an ordinary equivalence relation and applied directly to sets of states of any size. State distinguishability, on the other hand, does not obey the reflexive and transitive laws and, hence, can be applied only on pairs of states.

In some cases equivalence or distinguishability of a pair of states belonging to the same machine can be established by inspection of the transition table of this machine. Some of these cases are described by means of the three lemmas.

LEMMA 3.1

Let σ_i and σ_j be states of machine M_1. If rows σ_i and σ_j in the z_v subtable of M_1 are distinct then $\sigma_i \neq \sigma_j$.

LEMMA 3.2

Let σ_i and σ_j be states of machine M_1. If rows σ_i and σ_j spanning the entire transition table of M_1 are identical, then $\sigma_i = \sigma_j$.

LEMMA 3.3

Let σ_i and σ_j be states of machine M_1. If rows σ_i and σ_j spanning the entire transition table of M_1 become identical when every σ_i is replaced by σ_j (or every σ_j is replaced by σ_i), then $\sigma_i = \sigma_j$.

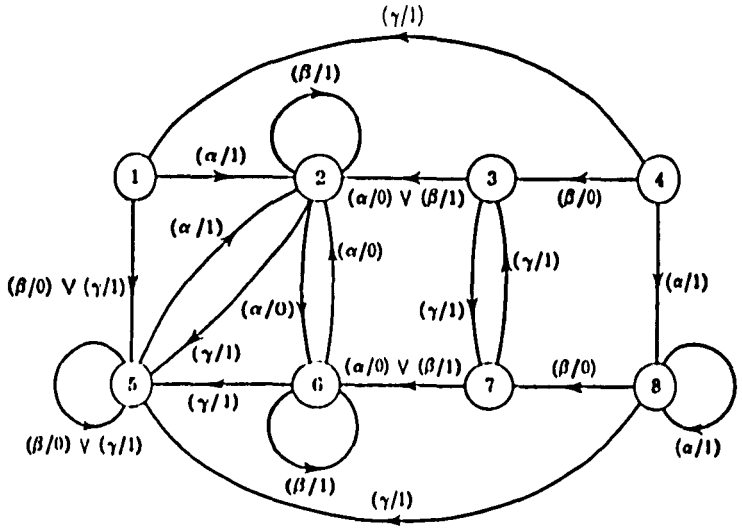


FIGURE 3.4
MACHINE A2

Pairs of rows which exhibit the property cited in lemma 3.1 are said to be *simply distlnguishable.* Pairs of rows which exhibit the properties cited in lemma 3.2 or lemma 3.3 are said to be *simply equivalent.*
We thus have:

THEOREM 3.2
If o_i and σ_j are simply distinguishable, then $o_i \neq q_j$: If σ_i and o_j are simply equivalent, then $\sigma_i =_{(J_j}$
It should be pointed out that the converse of theorem 3.2 is not true: Not every distinguishable pair of states is simply distinguishable and not every equivalent pair of states is simply equivalent.

To illustrate lemmas 3.1 to 3.3, consider machine A2, specified by figure 3.4 and table 3.3. It can be noted that rows 1 and 5 in the transition table arc identical and that rows 2 and 6 become identical when every 2 is replaced by 6 (or every 6 is replaced by 2). Consequently, each of the state pairs (1, 5) and (2, 6) is equivalent, A glance at the z_y subtable of machine 2 reveals that no state in the set (1, 4, 5, 8) can be equivalent to any state in the set (2, 3, 6, 7).

	z_v			s_{v+1}		
x_v / s_v	a	β	r	a	β	r
1	1	0	1	2	5	5
2	0	1	1	6	2	5
3	0	1	1	2	2	7
4	1	0	1	8	3	1
5	1	0	1	2	5	5
6	0	1	1	2	6	5
7	0	1	1	6	6	3
8	1	0	1	8	7	5

TABLE 3.3

MACHINE A2

3.5.1.1) k-EQUIVALENCE

A useful notation for future discussions is that of k-equivalence:

DEFINITION 3.3

State a_i of machine M. and state σ_j of machine M_2 arc said to be *k-equivalcnt,* if M_1/σ_i and M_2/σ_j, when excited by an input sequence of length k, yield identical output sequences. If σ_i and σ_j are not k-equivalentthey are said to be *k-distinguishable.* M_1 and M_2 may refer to the same machine.

Thus σ_i and σ_j are k·cquivalent *iff* there is no way of distinguishing, by using input sequences of length k and by observing the external terminals, between machine M_1 at state σ_i and machine M_2 at state σ_j From definition 3.2 it can be readily verified that k'equivalence obeys the reflex, symmetric and transitive laws. Consequently, k-cquivnlence can be treated as ordinary equivalence relation and applied directly to sets of states of any size.

LEMMA 3.4

- If two states nrc k·cquivnlent, then they nrc I-equivalent for every $I \leq k$;
- If two states arc k-distinguishable, then they arc l-distinguishable for every $l \geq k$.

The state into which state 0_i passes when an input sequence of length k is applied is called the *kth successor* of 0_i with respect to this sequence, 111e zeroth successor of D state is the state itself.

THEORIEM 3.3

If states o_i and o_j are k-equivnlcnt and if their kth successors wilh respect to any input sequence of length k are equivalent. then $0_i = 0_j$

THEORIEM 3.4

If stlltcs σ_i and σ_j are equivalent, then their kth successors, with respect to any input sequence of length k and for nny k, are equivalent.

The preceding results can he used. in many cases, to estuhlish equivalence of states when the equivalence of other states is already established. Suppose, for example that the pairs of states (1, 5} and (3. 7} in mnchine A2 of figure 3.4 are known to be equivalent. Consequently, the pair (", 8} must be equivalent. since 4 and 8 lire l-equivalcnt, with the pairs (1,5) and (3,7) being their first successors, If the pair (4. 8} is known to be equivalent , then the pairs (I, 5}, (2.61 and (3. 7) must also be equivalent, since they constitute pairs of corresponding states in paths originating in states 4 and 8.

3.5.1.2) k-EQUIVALENCE PARTITIONS

For the purpose of state minimization, it is of interest to divide, or partition. the states of a machine into classes according to the following criteria:

- All states which belong to the same class must be k-equivalent;
- nil states which belong to different classes must be k-distinguishable.

111is partition is called the *k-cquivalcnce partitioning* of the machine and is denoted by P_k. 111e classes of P_k are called *k-equivalcncc classes* nnd are denoted by Σ_{k1}, Σ_{k2}, Σ_{k3}, etc. States belonging to the same class arc called *m/joint states;* states belonging to different classes are called *disjoint states.*

LEMMA 1.5

The k-equivalence partition of n machine is unique,

LEMMA 3.6

States which arc disjoint in P_k must also be disjoint in P_{k+1}.

LEMMA 3.7

If machine M contains two distinguishable states which are r-cqulvalent, then it must also contain two states which are k-equivnlent but (k + t)-distinguishable.

TIIEORF.M 3.5

P_{k+1} must be a proper refinement of P_k, unless the adjoint states in every class of P_k are equivalent, in which case P_k and P_{k+1} arc identical.

In any but the simplest cases, the process of determining the equivalence partition of a given machine by inspection of the transition table or diagram is virtually impossible. A method for describing the partitioning can be carried out systematically, by constructing a series of so-called P_k *tables*.

The P_k table of a given machine is essentially the same as the *sv+}* subtable for that machine, with the following modifications:

> If {a. $_{r}$, a,'2' ... $_{r}$} is a class in p_k, rows a_i}. σ_{i2} .... σ_{ir} are grouped together. each group from the adjacent ones by a rule. The order of the groups in the table and the order of the rows within each group are arbitrary. Rows which belong to the same group and hence represent a k-equivalence class. will be called *adjoint rows;* rows which belong to different groups will be called *disjoint* rows;
>
> a "Σ" column is added. which labels each group of rows in the P_k table. The labels are arbitrary and may be chosen independently in each new P_k table;
>
> a subscript is attached to every *s"+1* entry, which identifies the group in the P_k table to which the entry belongs. Thus, if row a_i is in the group labelled "*a*", then every s_{v+}} entry "σ_i" is assigned the subscript "*a*".

Tables 3.4 to 3.7 are the PI' P2' P_3 and P_4 tables for machine A3 of figure 3.5.

3.5.1.3) CONSTRUCTION OF P_k TABLES

3.5.1.3.1) CONSTRucnON OF THE P_1 TABLE

Reorder the rows of the transition tahie so that rows which are identical in the z_v subtable become adjacent. Each group of such rows corresponds to a l-equivalence class and hence to a group of adjoint rows in the PI table. The PI table can now be constructed by

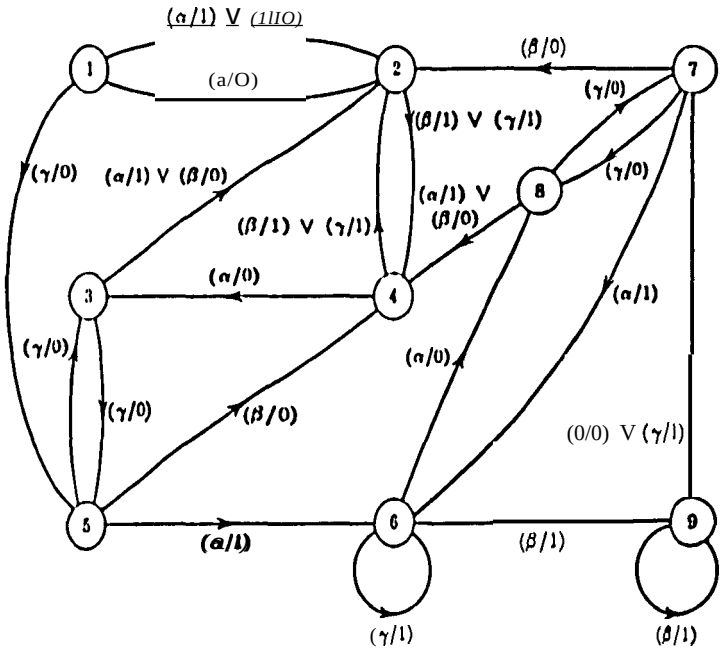


FIGURE 3.5

MACHINE AJ

deleting the z_v subtable, separating the row groups by rules, adding a "Σ" column and subscripting the s_{v+J} entries as described above.

3.5.1.3.2) CONSTRUCTION OF THE P_{k+1} TABLE (k ≥ 1)

A pair of adjoint rows in the P_k table which, in every column, exhibit identical subscripts are adjoint rows in the P_{k+1} table. A pair of adjoint rows in the P_k table which, in some column, exhibit different subscripts arc disjoint rows in the P_{k+1} table. Disjoint rows in the P_k table are also disjoint in the P_{k+1} table. A group in the P_k table consisting of a single row remains a single-row group in the P_{k+1} table. Thus, the groups of the P_{k+1} table can be established by inspection of the subscripts in the P_k table. Once the groups arc established, the table itself can be constructed according to the format stipulated above. The justification for the foregoing rules follows directly from the manner in which the subscripts are assigned and from the criteria for determining P_{k+1} from P_k.

As an example, consider the P3 table for machine AJ, shown in table 3.6. In group -a-, rows I. 3 and 8 have identical subscripts in every column and so do rows 5 and 7 (whose subscripts differ from those of I. 3 and 8). Consequently, rows I. 3 and 8 and rows 5 and 7

		s_{v+1}		
Σ	x_v s_v	a	β	γ
a	t	2_b	2_b	5_a
	3	2_b	2_b	5_a
	5	6_b	4_b	3_a
	7	6_b	2_b	8_a
	8	4_b	4_b	7_a
b	2	1_a	4_b	4_b
	4	3_a	2_b	2_b
	6	8_a	9_b	6_b
	9	7_a	9_b	7_a

TABLE 3.4

P1 TABLE FOR A3

		s_{v+1}		
Σ	x_v s_v	a	β	γ
a	1	2_b	2_b	5_a
	3	2_b	2_b	5_a
	5	6_b	4_b	3_a
	7	6_b	2_b	8_a
	8	4_b	4_b	7_a
b	2	1_a	4_b	4_b
	4	3_a	2_b	2_b
	6	8_a	9_e	6_b
c	9	7_a	9_e	7_a

TABLE 3.5

P_2 TABLE FOR A3

constitute two groups of rows in the P4 table. In group *"b"* all rows exhibit identical subscripts in every column and hence the group remains intact in the P4 table. Groups *"e"* and *"d"*, consisting of one row each, can be transferred intact to the P_4 table.

Given a procedure for constructing the PI table and the P_{k+1} table from the P_k table ($k \geq 1$), one can construct the P_k table for successive values of k, until a table is obtained in which all adjoint rows exhibit identical subscripts in every column. TIle stub entries of these adjoint rows represent equivalent states and hence the groups of stub entries in this table represent the desired equivalence classes. For machine A3 the condition is exhibited by the P4 table in table 3.7. TIle equivalence partition for mnchine A3 is, therefore, given by:

P: {I, 3, 8}, (2, 4), (5, 7), (6), (9)

Σ	x_ν / s_ν	$s_{\nu+1}$: a	β	t
a	t	2_b	2_b	5_a
	3	2_b	2_b	5_a
	5	6_c	4_b	3_a
	7	6_c	2_b	8_a
	8	4_b	4_b	7_a
b	2	1_a	4_b	4_b
	4	3_a	2_b	2_b
c	6	8_a	9_d	6_c
d	9	7_a	9_d	7_a

TABLE 3.6

P_3 TABLE FOR A3

Σ	x_ν / s_ν	$s_{\nu+1}$: a	β	γ
a	t	2_b	2_b	5_c
	3	2_b	2_b	5_c
	8	4_b	4_b	7_c
b	2	1_a	4_b	4_b
	4	3_a	2_b	2_b
c	5	6_d	4_b	3_a
	7	6_d	2_b	8_a
d	6	8_a	9_e	6_d
e	9	7_c	9_e	7_c

TABLE 3.7

P4 TABLE FOR A3

3.5.2) MACHINE EQUIVALENCE

The concept of equivalence can be extended to entire machines, through the following definition:

DEFINITION 3.4

Machine M_1 and machine M_2 are said to be *equivalent* if to each state σ_i of M_1 there corresponds at least one state of M_2 which is equivalent to σ_i and to each state σ_j of M_2 there corresponds at least one state of M_1 which is equivalent to σ_j. If M_1 and M_2 are not equivalent, they are said to be *distinguishable*.

111l1S, M. and M_2 are equivalent *iff* there is no way of distinguishing. by observing the external terminals, between machine M. at any of its states and machine M_2 and between machine M_2 at any of its states and machine MI' M1 and M_2 are distinguishable *iff* there is at least one state in M1 which is not equivalent to any state in M_2. or at least one state

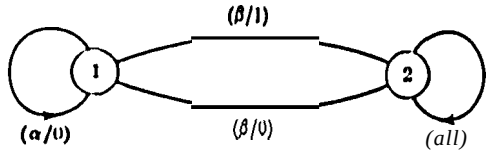


FIGURE 3.6

MACIIINE A4

in M_2 which is not equivalent to any state M_1. Equivalence between M1 and M_2 is indicated hy $M_t = M_2$ and distinguishablity is indicated by M. ≠ M2' From definition 3.4 it can be readily verified that machine equivalence obeys the reflexive law (M, $= M_j$), the symmetric law (if M. $= M_J$., then M. $= M_J$.) and the transitive law ($M_i = M_j$ and $M_j = M_k$' then M, $= M_k$). Consequently, machine equivalence can be treated as an ordinary equivalence relation and applied directly to sets of machines of any size.

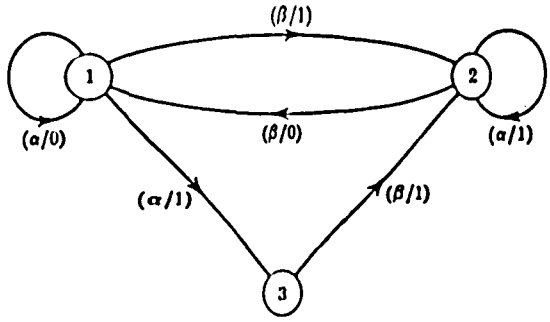


FIGURE 3.7

MACIIINE A5

Machines A4 and AS of figures 3.6 nnd 3.7, respectively, represent two equivalent machines. ntis can be verified by noticing that machine AS becomes identical to machine A4 when sme 3 of machine A4 is ignored; consequently, states) and 2 of machine A4 are equivalent to states I and 2. respectively, of machine AS. Also states 1 and J of machine A4 are simply equivalent and hence equivalent; consequently, state 3 of machine A4 is equivalent to stllte) of machine AS.

CHAPTER 4

MODULATION CODES FOR CLOCK EXTRACTION

Perhaps some future historian will classify this as the digital age, when everyday processes increasingly came to be performed using discrete numbers. The principles of numerical coding can be found in this chapter, accentuating modulation codes. Modulation codes, ie. codes based on runlcngth-limited (RLL) sequences or (d, k) constrained codes are, at present, used on disk recorders whether their nature is magnetic or optical. By far the most frequently reported coding schemes applied in recording practice have been those constituted by RLL sequences. Although this study is not concerned with recorders, it is with this type of coding scheme that this chapter is almost exclusively concerned. The reason for this is to providc a thorough background on RLL sequences for reference in chapters to follow, since we shall investigate modulation codes for usc on channels other than recording channels.

4.1) INTRODUCI10N

Many popular recording codes for peak detection channels fnll into the class of RLL codes. A digital magnetic recording channel is an example of a peak detection channel. For an in depth description of such a channel see [3]. These (d, k) codes, in their general form, were

pioneered by Freiman and Wyner[t5]. Kautz(16], Gabor[t7J. Tang and 8ahl[18] and notably by P. Franaszek [19J in the late 1960's. Since then. a considerable body of engineering and mathematical literature have been written on the subject. The length of time usually expressed in channel bits between transitions is known as the *nmlcngth.* RLL sequences are characterized by two parameters, $(d + 1)$ and $(k + 1)$ which stipulate. respectively. the minimum and maximum runlength that may occur in the sequence. The parameter d controls the highest transition frequency, and thus has a bearing on intersymbol interference when the sequence is conveyed over a bandwidth-limited channel. In the transmission of binary data It is generally desirable that thc received signal is self-synchronizing or self-clocking. Timing is commonly recovered with a phase-locked loop which adjusts the phase of the detection instant according to observed transitions of the received waveform. The parameter k ensures adequate frequency of transitions for synchronization of the received data. This quality is of interest for usc on mobile radlo channels,

DEFINmON 4.1

A elk-Iimitcd binary sequence, in short, $td, k)$ sequence, satisfies simultaneously the following two conditions:

I) d constraint: Two logical "ones" are separated by a nan of consecutive "zeros" of length at least d.

2) k constraint: Any run of consecutive "zeros" is of length at most k,

If only condition I is satisfied, the sequence is said to be d-Iimited (with $k = \infty$) and will be termed *d-sequence.*

In general, a $id,$ k)-sequcnce is employed in optical recording. magnetic recording and mobile radio channels with a simple preceding step. A $td,$ k)-sequence is converted to a runlength-lirnited channel sequence in the following way. Let the channel signals be represented by a sequence $\{y_i\}, y_i \in \{-I. I\}$. The logical "ones" in the $(d,$ k)-sequence indicate the positions of a transition $1 \to -1$ or $-1 \to 1$ of the corresponding runtength-limited sequences.

EXAMPLE 4.1

Consider the binary $(d,$ k)·sequcnce given by:

Coder output: 0 1 0 0 0 I 0 0 1 0 0 0 1 1 0 I ...

The transformed RLL channel sequence would be converted to:

Channel waveform: 1 -1 -1 -I ·11 1 1 -1 -1 -1 -I I -1·1 1

The mapping of the wavefonn by the precoding step is known as *non-retum-to-zero-inversion* (NRZJ), or *change of state* encoding. Waveforms which are transmitted without such an intermediate coding step are referred to as *non-return-to-zero* (NRZ). These names stcm (rom telegraphy and has no meaning in relation to recording channels; these nebulous tenns are in common usc and will be used throughout. It can readily be verified that the minimum and maximum distance between consecutive transitions of the RLL sequence derived from a *(d,* k)-scquence , respectively, is $d + 1$ and $k + I$ symbols.

For example, most flexible and low-end rigid disk files, as well as some high-end drives, today incorporate a code known as Modified Frequency Modulation (MFM). It also goes by other names, such as Delay Modulation or the Miller code. MFM is an RLL code with *id, k)* =(1, 3). The usc of this code on the flexible $5^{1}/_{4}$ disk uscd on IBM PC's, made it possible to store twice the previous amount of data on a disk made of the same material. The reason for this will become more apparent later.

The grounds on which *d* and *k* values are chosen, in tum, depend on various factors such as the channel response, the desired data rate (or infonnation density) and the jitter and noise characteristics. TIle problem faced by the coding theorist is the construction of a simple, efficient correspondence, or code mapping, between the arbitrary binary strings that a user might want to store on a magnetic disk or transmit over a channel, and the *id, k)* constrained code strings which the peak detector can more easily recover correctly. The tenn "efficient" arc now given quantitative meaning, by introducing the third important parameter, the code rate.

TIle conversion of arbitrary strings to constrained *(d, k)* strings could be accomplished as follows. Pick a codeword of length *n*. List all the strings of length *n* which satisfy the *(d, k)* constraint. If there arc at least 2^m such strings, assign a unique codeword to each of 2^m possible binary input words of length *m*. This kind of code mapping is commonly referred to as a *block code.* The ratio, *mln,* of input word length *m* to codeword length *n* is called the *code rate,* designated hy R. Since there arc only t' unconstrnincd binary strings of length *n,* there will be less than this numher of constrained codewords, Therefore, the rate must satisfy *mln* < I. In fact there is a maximum achievable rate, called the *Shannon capacity* C. In 1948, Shannon proved that, as the codeword length grows, the number of constrained codewords approaches 2^{Cn} from below, for some constant C which depends on the code constraints. This

result implies that the rate mln of any code mapping for that constraint must satisfy $mln \leq C$. Roughly speaking. a code is called *efficient* if the rate mln is close to C. It is possible to derive the capacity $C(d, k)$ of (d, k)-sequences. Sequences that meet prescribed (d, k) constrains may be thought of to be composed of phrases of lengthj + 1. $d \leq j \leq k$, denoted by T_{j+1}, of the fonn $(1\text{V}, 10^{d}+/, \ldots 10^{j} \ldots 10^{k})$, where 0^{j} stands for a sequence of } consecutive "zeros". The characteristic equation of (d, k)-sequences ([2] and [3]) is (for finite k):

$$z^{-(k+1)} + z^{-k} + \ldots + z^{-(d+1)} - 1 = 0. \tag{4.1}$$

or

$$z^{k+2} - z^{k+1} - z^{d+1} + 1 = 0 \tag{4.2}$$

Following Shannons definition. the capacity of a (d, k) sequence is given by:

$$C(d, k) = \lim_{n \to \infty} 1/n \log_2 N_d(n) \tag{4.3}$$

where $N_d(n)$ denotes the number of distinct d sequences of length n [3]. For large n,

$$N_d(n) \propto \lambda_{max}^{n} \tag{4.4}$$

Applying equation (4.3), the capacity of d constrained sequences. denoted by $C(d, 00)$ is:

$$C(d, 00) = \lim_{n \to \infty} 1/n \log_2 Nd(n) = \log_2 \lambda_{max}. \tag{4.5}$$

with λ_{max} the maximum real root (eigen value) of the characteristic equation.

Table 4.1 lists some of the capacities. $C(d, k)$. versus the runlength parameters d and k. A complete list of (d, k) capacities can be found in (20]. As can be seen in table 4.1. the quantity $C(d, 00)$ supplies the maximum rate possible of any implemented code for a given d constraint. TIle capacity decreases with increasing d which may be a property of much concern on some channels, especially on mobile communication channels where there is such a bandwidth shortage. On a recording medium an increasing d parameter, as will be seen in the following paragraph. suggests a higher density ratio.

Some further results of computations. which arc obtained by numerical methods [3), are collected in table 4.2. The quantity DR. called *density ratio,* sometimes called *packing (Icnsity* which express the minimum physical distance between consecutive nansitions of an RLL sequence, is defined as:

$$DR = (d + 1)R \tag{4.6}$$

where R is the rate of the RLL code. It can he seen in table 4.2 that an increase of the density ratio can be obtained at the expense of decreased code rate. It can even be shown that the density ratio DR can be made arbitrurily large by choosing the minimum runlength d sufficiently large. This follows from:

$$DR = (d + 1)\log_2 \lambda_{max} \tag{4.7}$$

	d=0	d=1	d=2	d=3	d=4
k = 1	0.694242	0.000000	0.000000	0.000000	0.000000
2	0.879146	0.405685	0.000000	0.000000	0.000000
3	0.946777	0.551463	0.287761	0.000000	0.000000
4	0.975225	0.617447	0.405685	0.223180	0.000000
5	0.988109	0.650900	0.464958	0.321757	0.182342
6	0.994192	0.669032	0.497906	0.374585	0.266896
7	0.997134	0.679286	0.517370	0.405685	0.314230
8	0.998578	0.685252	0.529340	0.425068	0.343229
9	0.999292	0.688789	0.5369JJ	0.437620	0.361992
10	0.999647	0.690915	0.541797	0.445971	0.374585
11	0.999824	0.692203	0.544997	0.451640	0.383262
12	0.999912	0.692989	0.547114	0.455546	0.389359
13	0.999956	0.693471	0.548527	0.458268	0.393709
14	0.999978	0.693767	0.549475	0.460182	0.396847
15	0.999989	0.693949	0.550114	0.461539	0.399133
oo	1.000000	0.694242	0.551463	0.464958	0.405685

TABLE 4.1

CAPAcrrY $C(d, k)$ VERSUS RUNLENGTH PARAMETERS d AND k

The root of equation 4.2 satisfies

$$\left(\frac{(1-\varepsilon)d}{\log_2 d}\right)^{1/d} \le \lambda \le \left(\frac{(1+\varepsilon)d}{\log_2 d}\right)^{1/d} \tag{4.8}$$

d	$C(d, -)$	$(d+1)C(d, \infty)$
1	0.694	1.388
2	0.551	1.654
3	0.465	1.860
4	0.406	2.028

TABLE 4.2

C AND DR VERSUS MINIMUM RUNLENGTH d

for large d [3]. Thus DR grows like a constant times log d, see table 4.2. Another important parameter in recording systems, is the width of the detection window, during which the presence or absence of a transition has to be detected. This detection window is of width m/n data bit intervals and a large detection window is preferred due to possible bit synchronization imperfections during the read or receive process.

It should be appreciated that codes with a larger value of d, and thus a lower rate, provide an increasingly difficult trade-off between the detection window and the density ratio in applications with very high information density and data rates. However, it was experimentally demonstrated that the detection window has no apparent influence on a mobile radio channel. This experimental results can be found in chapter 9.

4.2) Finite-State Transition Diagrams

In this section the background developed in chapter 3 are put to use in communication systems employing runlength-limited codes. Most (d, k) codes can be implemented as FSM's and a brief description of the procedure taken to develop a FSM for a specified (d, k) sequence will be shown.

One can employ *a finite-state transition diagram* (FSTD), also known as a *Markov model*, in order to conveniently represent the infinitude of binary strings satisfying the (d, k) constraint.

This graph representation for constrained channel strings dates back to Shannon's seminal paper [21] and it was exploited by Franaszek in his work on RLL codes. figure 4.2 shows a FSTD for the (1, 3) constraint, It consists of a graph with 4 vertices, called states and oriented branches between them, called edges, represented by arrows, The edges are labelled with

channel bits. Paths through the graph correspond precisely to the binary strings satisfying the (I, 3) constraint. A similar FSTD having k + 1 states can be used to describe any (d, k) constraint. While table 4.1 shows (d, k) sequences with their theoretical capacities C, table 4.3 shows the practical achievable rates, R = *mln im* and *n* small integers), and the efficiencies" = *RIC* for a few constraints.

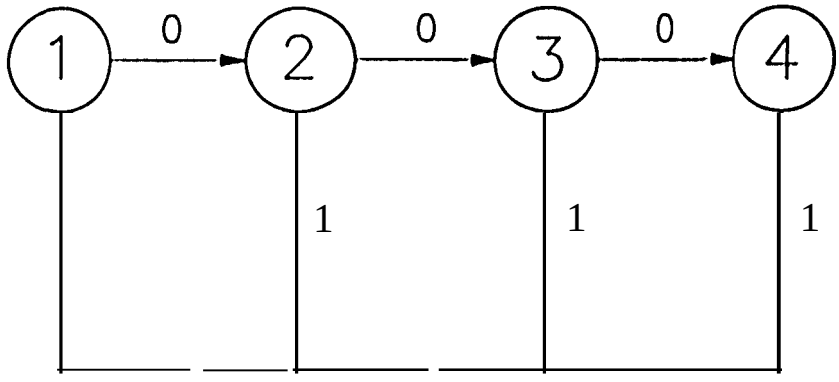


FIGURE 4.1

FJNTm STATE DIAGRAM FOR (d, k) =(1,3)

The conclusion drawn from table 4.3 is that it is possible to construct codes in such a way that the rate will be near the theoretical limit for the code. The rate R $=1/2$, (d, k) =(2, 7) code is currently applied in the IBM 3380 rigid disk drive.

(d, k)	CAPACITY C	RATE R	EFFICIENCY η
(0, 1)	0.6942	$2/3$	0.96
(1,3)	0.5515	$1/2$	0.91
(1,7)	0.679	$2/3$	0.98
(2,7)	0.5172	$1/2$	0.97

TABLE 4.3

COMPARISON OF CODES

The capacity C of the RLL (d, k) constrained channel is directly related to the structure of the FSTD. TI,e state-transition matrix $\bar{T} = I_{ij}$ is defined and associated to the FSTD with states I, ..., k + 1 as follows:

- t_{ij} = x, if there are x edges from state ; to statej;
  = 0. otherwise.

For example, for the (d, k) = (J. 3) case:

$$T = \begin{bmatrix} 0100 \\ 1010 \\ 1001 \\ 1000 \end{bmatrix}$$

Note that λ_{max} can also be obtained by solving $T . \lambda I = 0$ [2J. It is found that λ_{max} = 1.465. in which case C = 0.5515. In practice, one chooses for the rate a rational number $m/n \leq C$. To help keep the codeword length small, the integers m and n are often selected to be small. Thus. for the (1 , 3) constraint, it would be natural to look for a code mapping at rate R = 1/2. which uses codcwords of length 2 bits. Sec table 4.3.

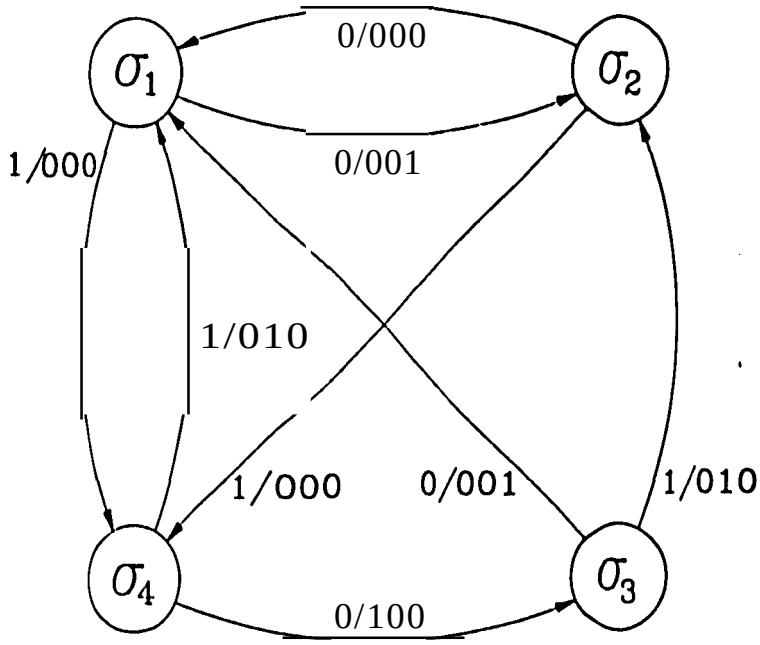
4.2.1) SYNTImSIS ALGORITHM FOR MODULATION CODES

The algorithm presented were used to obtain a rate, R = 1/3, (d, k) = (3. 7) code. This code was considered for usc on mobile radio channels. but was not used because of its inefficient usc of bandwidth. In a later chapter another algorithm will be presented for constructing (d, k) = (0, k) constrained codes.

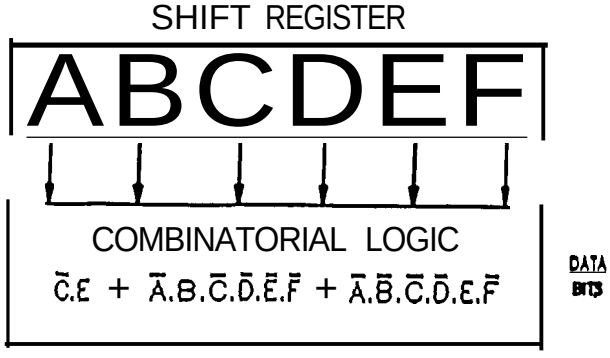
The code wcre synthesized using the algorithms for sliding block codes [22], of which a tutorial exposition mny be found in [2]. Briefly, it comprises the following steps when synthesizing a R = m/n code. Starting with the Markov model, G, for thc (d, k) constrained channel. develop the graph G^n and state-transition matrix $\bar{T}^n$. Next find the simplest approximate eigenvector V which satisfies

$$\bar{T}^n \bar{V} \geq 2^m \bar{V} \tag{4.9}$$

As shown in [22J and (21. the approximate eigenvector directs the state splitting of G^n which is the next synthesis step, 111e Shannon capacity from [20J and efficiency for the (d, k) = (3, 7) constrained code investigated, are C = 0.405685 and η = 0.822. Finite-state transition diagrams for the encoder and sliding-block decoder are shown in figure 4.2.



a)



b)

FIOURP. 4.2

R $=1/3$, (d, k) = (3,7) SYSTEMATIC CODE

a) ENCODER b) DECODF.R

4.3) SPECTRUM OF RLL SEQUENCES

If it is assumed that a iransmltter emits the phrases T_J independently with probability $Pr(T_J)$, then the power spectral density function of the corresponding RLL sequence is given by [3]:

$$\Pi(\omega) = \frac{1}{T} \frac{1}{\sin^2 \omega/2} \frac{|1 - \Theta(\omega)|^2}{|1 + \Theta(\omega)|^2} \tag{4.9}$$

whcre

$$G(w) = \sum_{I=d+1}^{k+1} Pr(T_l) e^{i\omega l} \tag{4.10}$$

and

$$\bar{T} = \sum_{l=d+1}^{k+1} l P_d T_l \tag{4.11}$$

An elegant proof. which is based on gcnernting functions. is available in [3]. The runlengihs of a *maxcntropic sequence'* follow a truncated geometric distribution with parameter λ:

$$Pr(T_l) = \lambda^{-l} \cdot l = d + 1, d + 2 \cdot \ldots \cdot k + 1 \tag{4.12}$$

whence

$$\bar{T} = \sum_{l=d+1}^{kt1} l Pr(T_l) = \sum_{l=d+1}^{kt1} l \lambda^{-l}. \tag{4.13}$$

Substitution of the distribution provides a straightforward method of determining the spectrum of maxentropic RLL sequences, Figure 4.3 depicts the spectra of the maxentropic MFM code. as wcll as the implemented spectra. Figure 4.4 shows the spectra H(w) of some maxentropic *d* sequences for selected values of the minimum runlength *d.*

The following characteristics may be observed: From figure 4.3, a surprisingly good conformity (a few dB diffcrence) with the spectra of the maxentropic counterpart in the low frequency range can be seen, while. at higher frequencies. there is a significant difference. From figure 4.4, a maxima occur at non-zero frequency. and the spectra exhibit a more pronounced peak with increasing *d.* The energy in the low-frequcncy range

I A sequence is termed maxentropic if Ihe thcoretlcal caJlacily equals Ihe practical achievable dala nle. lhus If Rile.

diminishes with decreasing minimum runlength *d*. THE effects on the spectra of a reduction of the maximum runlength can be seen in figure 4.5. The figure depicts the spectrum of maxentropic ($d = 2$) sequences with the maximum runlength *k* as parameter.

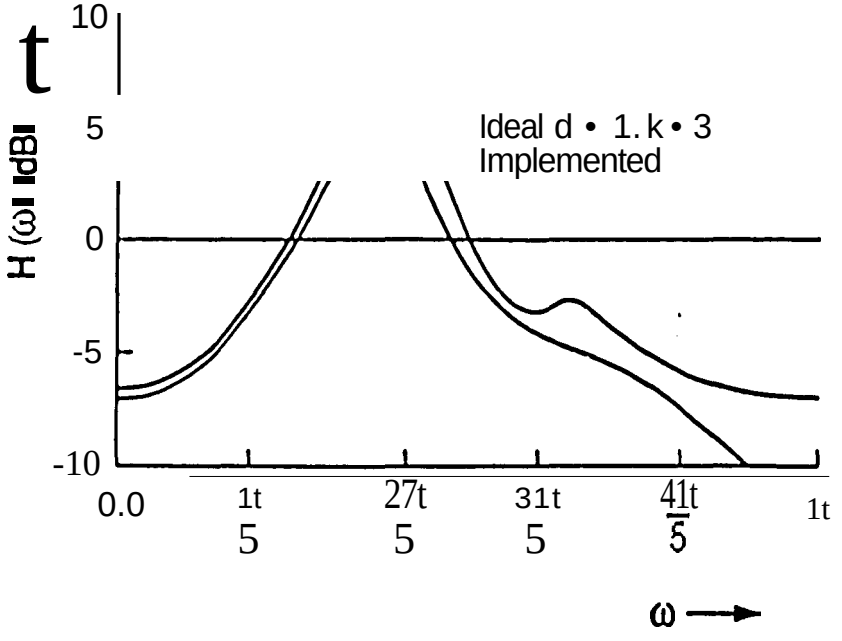


FIGURE 4.3
SPECTRUM OF MFM CODE

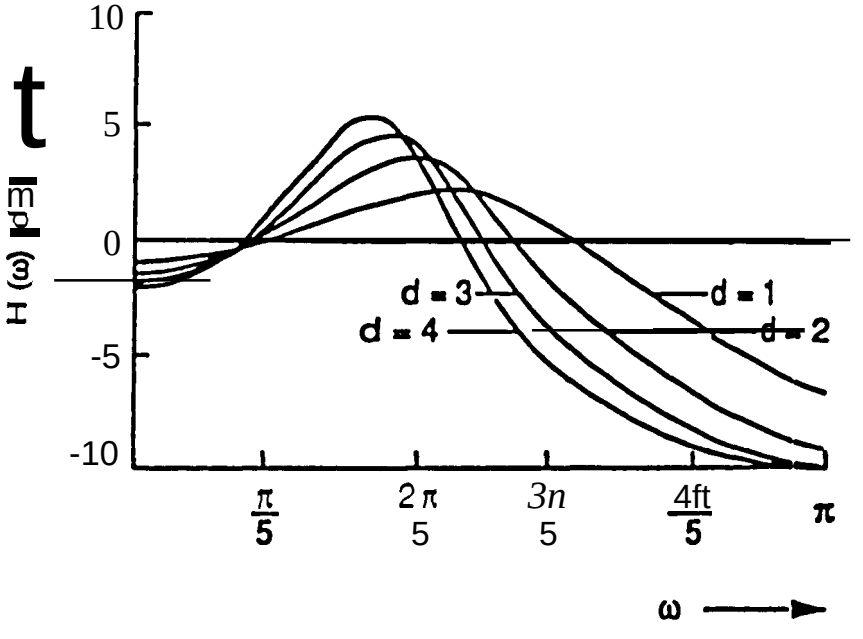


FIGURE 4.4
SPECTRUM OF MAXENTROPIC RLL SEQUENCES, *k* CONSTANT

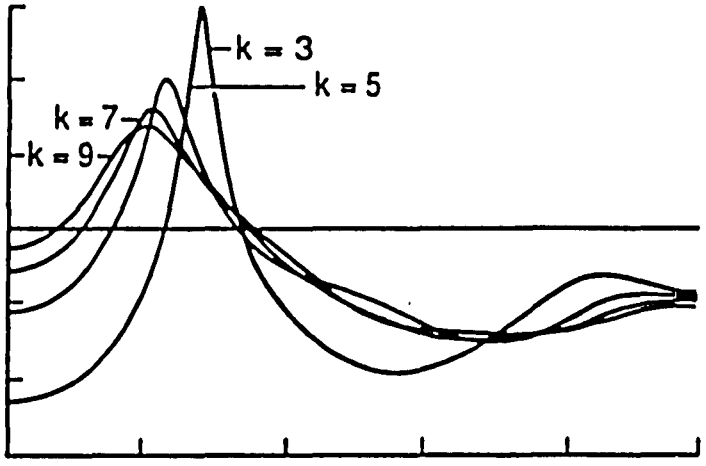


FIGURE 4.5
SPECTRUM OF MAXENTROPIC RLL SEQUENCES, *d* CONSTANT

4.4) FIXED-LENGTH BINARY RLL CODES

One approach that has proved very successful for the conversion of arbitrary source information into constrained sequences is the one constituted by *block codes*. The source sequence is partitioned into blocks of length *m*, and under the code rules such blocks are mapped onto words of *n* channel symbols. A code may be state-dependent, in which case the choice of the codeword used to represent a given binary source block is a function of the channel or encoder state, or the code may be state independent. State independence implies that codewords can be freely concatenated without violating the sequence constraints. The additional restriction leads, in general, to codes that are longer than state-dependent codes for a given bit-per-symbol value. In some instances, state independence may yield advantages in error propagation limitation. The EFM code, used on CD players (3], is a proper representative of a code with limited error propagation. State-independent decoding may be achieved for any fixed-length (d, k) code.

EXAMPLE 4.2

Consider the MFM code, $R = 1/2$, $(d, k) = (1, 3)$ and $\eta \simeq 0.91$. It has proven very popular from the viewpoint of simplicity and ease of implementation. MFM is essentially a block

code of length $n = 2$ with a simple merging rule when the NRZI notation is employed. The MFM encoding table is shown in table 4.4. The symbol indicated with 'x' is set to 'zero' if the preceding symbol is 'one', else it is set to 'one'. It can be verified that this construction yields a maximum runlength $k = 3$.

SOURCR	OUTPUT
0	x0
1	01

TABLE 4.4

CODING RULES FOR MFM CODE

A graphical representation of the FSM underlying the MFM code, using NRZI notation rules, is pictured in figure 4.6. '11e labelled edges emanating from a state define the encoding rule, and the state in which an edge terminates indicates the state, or coding rule to usc next. '11c state A represents the condition that the previous channel bit was 'zero',

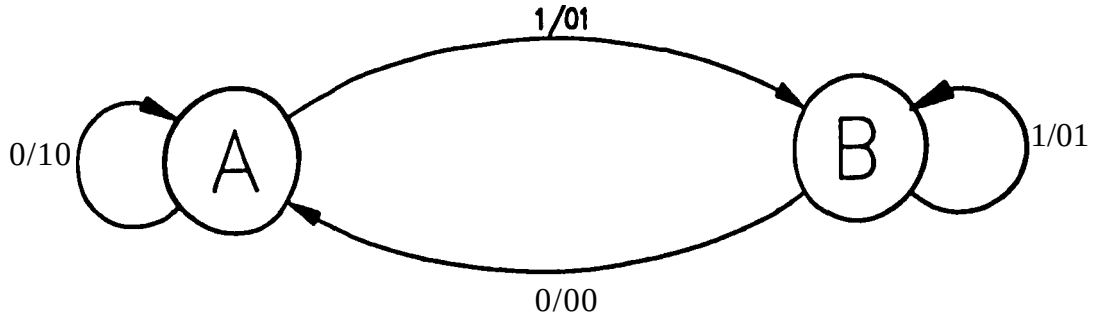


FIGURE 4.6
MFM CODE (NRZI)

while thc state B indicates that the previous channel bit was a 'one'. It may be noticed that there are only two states needed to model the MFM code whereas the FSM based on NRZ notation of the MFM code needs four states: see figure 4.7. The explanation is that thc change-ol-state encoder, which is used to translate a *td, k)* sequence into a RLL sequence, accounts for one memory clement, or a douhling of the numher of states. Decoding of the MFM code is simply accomplished by discarding the redundant first hit in each received 2·bit block.

'I1,C MFM code wns also recommended by the CCIR to be transmitted over a mobile VHF communicntion channel. Again, the experimental results can be found in chapter 9.

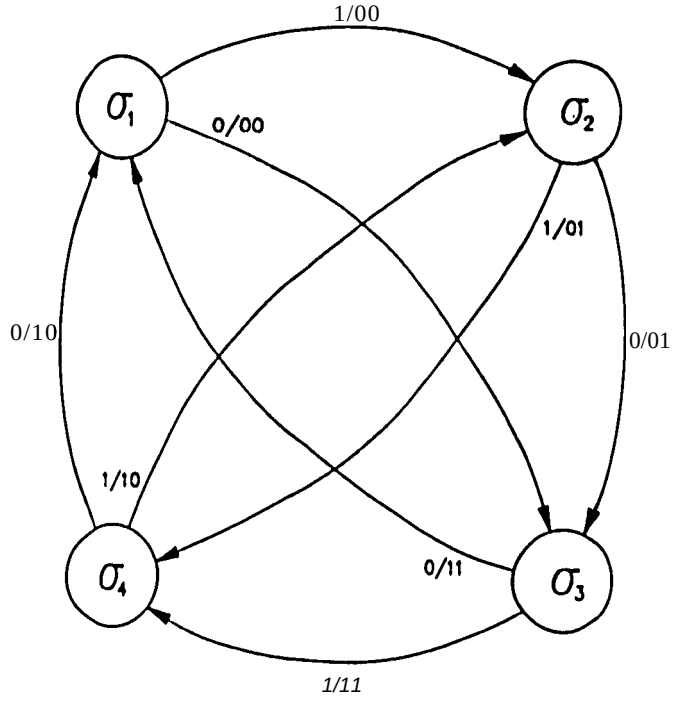


FIGURE 4.7
MFM CODE (NRZ)

4.5) VARIABLE-LENGTII BINARY RLL CODES

In an attempt to increase fixed-length state-dependant code efficiency, the codeword length is increased and thus result in a rapid increase of coder and decoder complexity [3]. Variable-length codes, which may combine the advantages of short nnd long word lengths, are frequently profitable in terms of hardware complexity. The basis of variable-length synchronous codes was laid by Frnnaszek with his pioneering work [19]. Variable-length codes offer the possibility of using short words more often thnn those of longer lengths. This often permits a marked reduction in coder and decoder complexity relative to a fixed-length code of like rnte and sequence properties.

The structure of variable-length codes required to comply with sequence properties is quite similnr to that of fixed-length codes. Various special features, however, arise from the presence of words of different lengths. The requirement of synchronous transmission, coupled with the assumption that ench word carries an integer number of infonnotion bits, implies that the codeword lengths are integer multiples of a basic word length n, where n is

the smallest integer for which the bit per symbol ratio *mln* is that of two integers. The example to follow is as a whole due to the work of Franaszek.

EXAMPLE 4.3

Table 4.5 discloses the code table of the R = $^1/_2$, (d, k) = (2, 7) and η =0.97 code, which, as mentioned before, forms the bed-rock of the IDM 3380 high-performance rigid disk files.

DATA	CODE
10	0100
11	1000
011	001000
010	100100
000	000100
0011	00001000
0010	00100100

TABLE 4.5

VARIABLE-LENGTH SYNCHRONOUS (2, 7) CODE

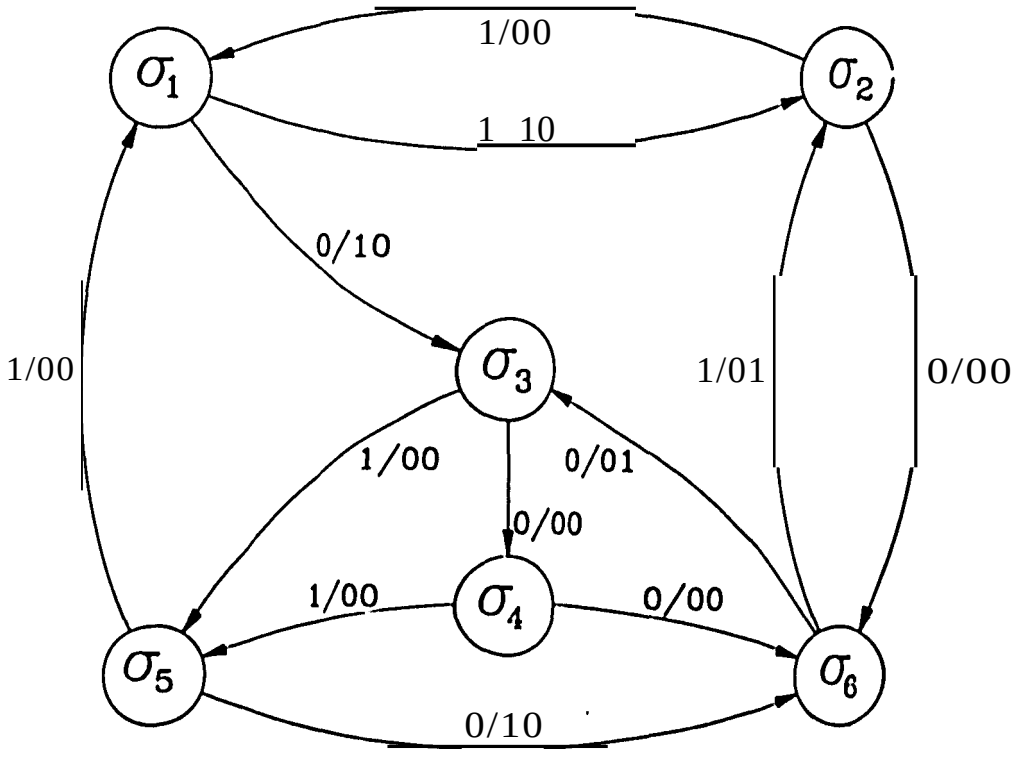


FIGURE 4.8

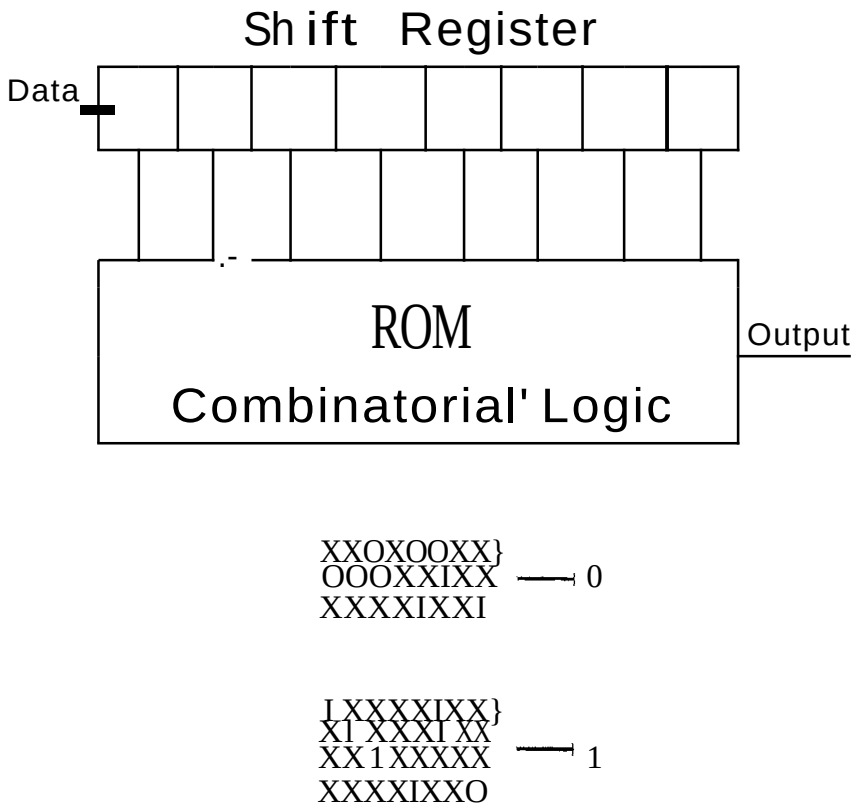
FSM ENCODER FOR (d, k) = (2, 7) CODR

The encoding of the incoming data is accomplished by dividing the source sequence into two-, three- and four-bit partitions to match the entries in the code table and then mapping them into the corresponding channel representations. Consider a source sequence 010111010. then after the appropriate parsing obtain:

in: 010 1110 10 ...,

which, by using table 4.5, is transformed into the corresponding output sequence:

out: 100100 1000 0100 0100 ...



FJGURR4.9

DRCODER FOR $(d, k) = (2, 7)$ CODE

Figure 4.8 and 4.9 shows, respectively, an encoder as FSM and decoder for this example. Decoding of the received message is achieved with a shift register of length eight. The incoming message is shifted into the register; every two channel clock cycles, because of $R = 1/2$.

This code was also transmitted over a mobile communication channel, with surprising results (chapter 9).

4.6) DC-BALANCFD Cones

In this section a brief overview on de-balanced, or ac-coupled, codes are given, together with a de-free code spectrum, For an in depth study on de-free codes see [3] and [23].

In digital transmission it is sometimes desirable for the channel stream to have low power near zero frequency. Suppression of the low-frequency components is usually achieved by restricting the unbalance of the transmitted positive and negative pulses. The de-balanced requirement further imposes a charge constraint. TIle waveform should have neither lengthy runlengths nor high-magnitude dc components. Expanding on Tang's notation. consider *td, k, C*) codes, where C is an upper bound on the accumulated charge of the waveform. The (*d*, *k*, *C*) codes have two primary constraints;

$$d \leq \text{run of zeros in code} \leq k, \text{ and} \qquad \text{(a)}$$

$$|\Sigma f_i| \leq C. \qquad \text{(b)}$$

with f; the channel bits in NRZ notation.

The essential principle of operation of a channel encoder that translates arbitrary source data into a de-free channel sequence is remarkably simple. The approaches which have actually been used for de-balanced code design arc basically three in number:

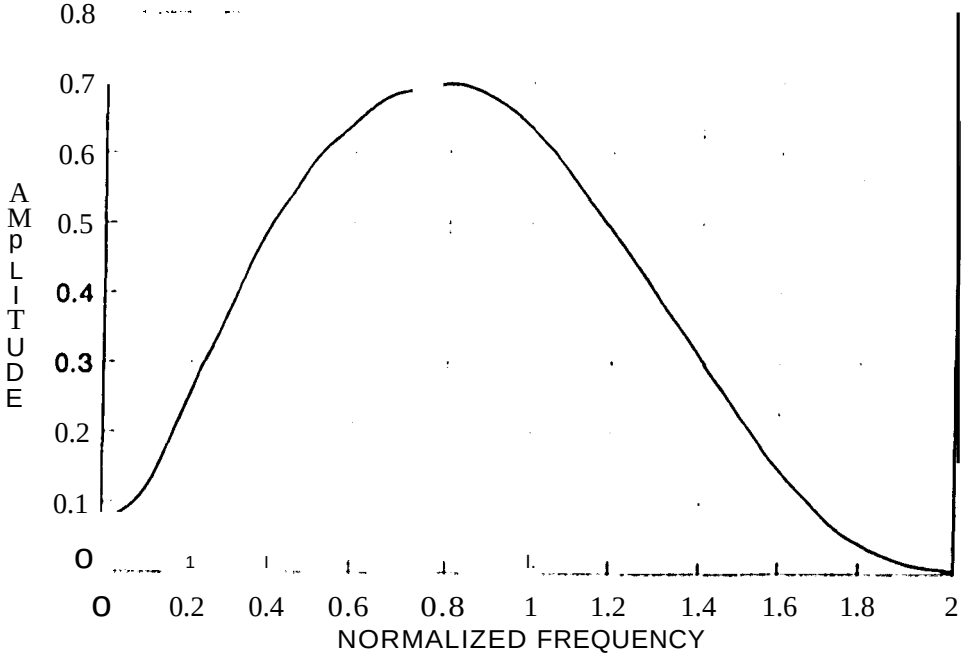
- Zero-disparity code;
- Low-disparity code;
- Polarity bit code.

TIle *disparity* of a codeword is defined as the excess of the number of 'ones' over the number of 'zeros' in the codeword: thus the codewords 000110 and 100111 have disparity -2 and +2, respectively. An important special case is zero-disparity codewords, which contain equal numbers of 'ones' and 'zeros', The obvious method for the construction of de-balanced codes is to employ codewords that contain an equal number of 'ones' and 'zeros', or stated alternatively, to employ zero-disparity codewonJs which have a one-to-one correspondence with the source words.

A good step will be to extend this mechanism to the *iow-dlsparity* code, where the translation are not one-to-one. The source words operate with two alternative modes which. being of equal or opposite disparity. each of them is interpreted by the decoder in the same way. The zero-disparity words are uniquely allocated to the source words. Other codewords are allocated in pairs of equal and opposite disparity. During transmission, the choice of a specific translation is made in such a way that the accumulated disparity, or the *running digital* Slim, of the encoded sequence, after transmission of the new codeword, is as close to zero as possible. The running digital sum (RDS) is defined for a binary stream as the accumulated sum of 'ones' and 'zeros' (a 'zero is counted as -1) counted from the start of the transmission. Doth of the basic approaches to de-balanced coding are due to Cattermole [24] and Griffiths [25].

Figure 4.10 shows the spectrum of a rate R =1/2, *id, k,* C) =(0, 1. J) code. In comparison with the MFM. rate R =1/2, *(d, k)* =O. 3) code, the spectrum of the de-free code. as expected, begins at zero, where the spectra of the MFM code begins a few dB higher.

Two de-free codes were also transmitted over mobile communication channels.



AMPLITUDE IN VOLT

FIGURE 4.10
SPECTRUM OF A RATE R=1/2, *(d, k,* C) = (0, 1,1)

CHAPTER 5

NEW FSM REPRESENTATIONS FOR MODULATION CODES

Recalling from chapter 3, two finite-state machines arc considered equivalent if:

- they have the same input set and the same output set, *and*
- for each state of one there is at least one state of the other, such that, assuming these states as initial ones, equal input sequences give rise to equal output sequences,

Roughly speaking, two FSM's are equivalent if they nre indistinguishable by regarding their input-output behavior, in the sense that the same output sequence can be equally well originated by both FSMs as a response to a given input sequence. FSM's can thus be divided into equivalence classes and any non-trivial FSM has infinitely many equivalent FSM's. Moreover, in each equivalence class there is a minimal FSM fl.e, a FSM with a minimal number of states), which can be obtained from each FSM of the class through a simple standard procedure (chapter 3). Finally, the minimal FSM of the class is unique, apart from a possible relabelling of the states.

111is fact has some relevance to the problems encountered in communication systems, since

this makes it possible to translate coder niles into different equivalent FSM's. Even if the analysis can be correctly perfonned on the basis of any FSM of the equivalence class. it may be "easier" with a particular FSM than with another; "easy" referring to computational complexity in terms of the number of elementary operations involved, in this case the minimal FSM is usually more convenient. or to a greater tractability of formulae in terms of algebraic manipulations. in which case the *Moore equivalent machine* or *state assigned machine* may be of interest. For example. when one wishes to determine the theoretical spectra of a specific code. the Moore equivalence machine may be used, since the algebraic manipulations will often be considerably minimized.

The minimal fonn FSM model derived in chapter 3 is also known as a *Mealy machine* or *transition assigned machine.* which can be particularized to give a Moore machine.

This chapter will attend to the conversion between this two types of isomorphic machines. together with a ncw algorithm developed to map fixed length coding rules on finite·state machines.

5.1) CONVERTING BE1WEEN IsOMORPIIIC MACHINES

There exist various algorithms for the conversion between Mealy and Moore machines; [14]. [13] and [26]. The method presented for conversion from Mealy to Moore is one with a more analytic nature; in other words a method which can be implemented with a digital computer. In fact. this algorithm was used in a program which convert between the two types of machines (see appendix E).

TI1C conversion from Moore back to Mealy is done using the minimization rules as outlined in chapter 3.

A Mealy machine can be converted to a Moore machine if the output function f_z is only state dependent, i.e. if (3.1) assigning the code-word is given by

$$z_v = f_z(s_v) \qquad \text{e.n}$$

and is independent of the source-word ξ. ntis structure implies some simplification in the characterizing function representation. In particular. in the matrix representation, the output matrices are equal:

$$r_v = r.\ v=1.\ldots. K \qquad (5.2)$$

Now it is straightforward to find a Moore machine equivalent to a given Mealy machine. Indeed, it suffices to define, as a new state, the pair $\tilde{s}_\nu = (s_\nu, \xi_\nu)$ and the new input $\tilde{\xi}_\nu = \xi_{\nu+1}$ and one gets the equivalent Moore machine $\tilde{M} = (X, Z, \tilde{S}, J_S', J_z)$, where the new space set $\tilde{S} \bullet S \times X$ and the new characterizing functions are given by:

$$\tilde{s}_{\nu+1} = J_s(\tilde{s}_\nu, \tilde{\xi}) = (f_s(s_\nu, \xi_\nu), \xi_{\nu+1}) \tag{5.3}$$

$$z_\nu = J_z(\tilde{s}_\nu) \equiv f_z(s_\nu, \xi_\nu) \tag{5.4}$$

Such a representation is achieved at the cost of a larger state set. $\tilde{S}$ has $K \times I$ states in comparison with the I states of the original state set S. On the other hand, it will be apparent later that this model can be more amenable for theoretical developments.

It can be verified [27] that the matrix representation of the Moore equivalent machine is straightforward related to the previous one.

The new state transition matrices $\tilde{E}_u$ {of order $K \times n$ become:

$$\tilde{E}_1 = \begin{bmatrix} E_1 & 0 & .. & 0 \\ E_2 & 0 & & \vdots \\ \vdots & & \ddots & \vdots \\ E_K & 0 & .. & 0 \end{bmatrix}, \ldots, \tilde{E}_K = \begin{bmatrix} 0 & \ldots & 0 & E_1 \\ \vdots & & 0 & E_2 \\ \vdots & \iddots & \vdots & \vdots \\ 0 & \ldots & 0 & E_K \end{bmatrix} \tag{5.5}$$

whereas the common output matrix $\tilde{\Gamma}_u$, of dimension $KI \times N$ (N the length of the code-words) is:

$$\tilde{\Gamma} = \begin{bmatrix} \Gamma_1 \\ \Gamma_2 \\ \vdots \end{bmatrix} \tag{5.6}$$

EXAMPLE 5.1

Considering the well-known Miller code, described in chapter 4. We wish to convert the two state Mealy machine to a $KI = 4$ state Moore machine. Figure 5.1 and table 5.1 describe the Mealy machine representation.

The Mealy machine is described by the following matrices (see chapter 3 for state transition matrix description):

$$E_1 = \begin{bmatrix} 0 & 1 \\ 0 & 1 \end{bmatrix}, E_2 = \begin{bmatrix} 1 & 0 \\ 1 & 0 \end{bmatrix}, \Gamma_1 = \begin{bmatrix} 0 & 0 \\ 1 & 0 \end{bmatrix}, \Gamma_2 = \begin{bmatrix} 0 & 1 \\ 0 & 1 \end{bmatrix}$$

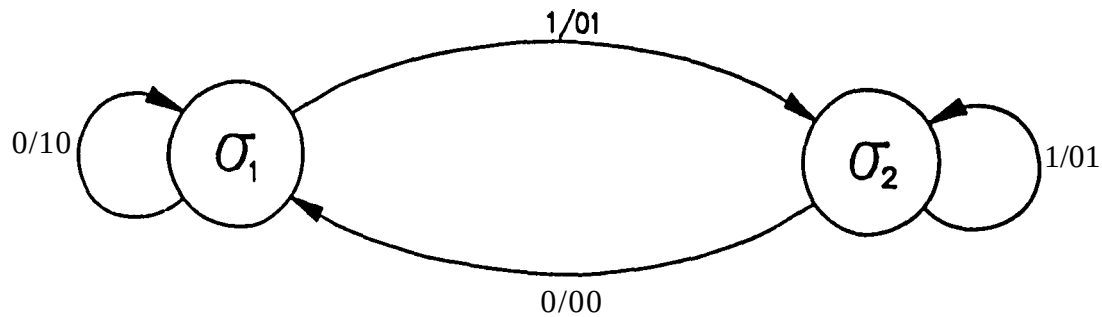


FIGURE 5.1

STATE TRANSITION DIAGRAM OF THE MILLER CODE

	0	1	0	1
σ_1	10	01	σ_1	σ_2
σ_2	00	01	σ_1	σ_2

TABLE 5.1

STATE TRANSITION TABLE OF THE MILLER CODE

We have: Source alphabet X= {0, 1}, code-alphabet Z = (00, 01, 10), state set $S = \{\sigma_1, \sigma_2\}$, source-word number $K=2$, code-word number $J=3$, state number $I=2$ and code-word length $N=2$.

The equivalent Moore FSM has $K/=4$ states (σ_1, ξ_1), (σ_1, ξ_2), (σ_1, ξ_1), (σ_2, ξ_2) and, following (5.5) and (5.6), the corresponding matrix representation is given by:

$$\hat{E}_1 = \begin{bmatrix} E_1 & 0 \\ E_2 & 0 \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix}, \quad \hat{E}_2 = \begin{bmatrix} 0 & E_1 \\ 0 & E_2 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

$$\Gamma = \begin{bmatrix} \Gamma_1 \\ \Gamma_2 \end{bmatrix} = \begin{bmatrix} 0 & 0 \\ 1 & 0 \\ 0 & 1 \\ 0 & 1 \end{bmatrix}$$

Figure 5.2 denote the state transition diagram of the new Moore representation.

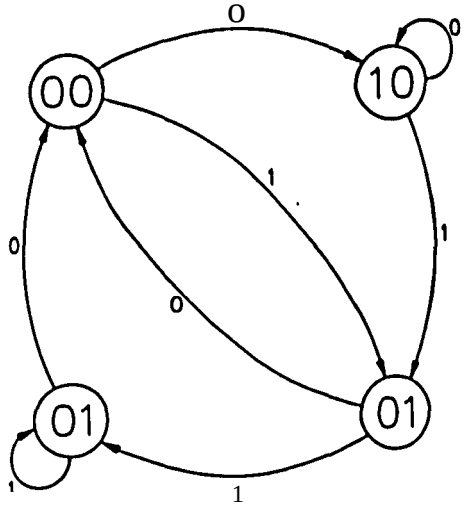


FIGURE 5.2

STATE TRANSITION DIAGRAM OF MOORE MACHINE

At this point it would be interesting to know if there is a minimum *Moore* machine representation. Since the normal procedure of minimization would lead back to a Mealy machine, that method cannot be used. However, a new algorithm was developed. by the author, for a minimum Moore machine. This method was applied extensively to verify its validity.

The first step would be to consider the states that would produce the same outputs when *entered* (rom another state. In the fonner example it will be states σ_J and σ_4. The next step is to compare the next states, thus the outputs of these states for equivalence. Returning to the example, we know that any input leading to states σ_3 and σ_4 will have the output 01. Considering the next states, it is evident that the two states is equivalent. (see table 5.2).

Figure 5.3 shows the new minimized state diagram (3 states) for the Miller code as Moore machine. To verify the conversion from Mealy to Moore machine representation and the minimized Moore machine, consider the following input sequence:

	0	1	0	1
σ_3	00	01	σ_1	σ_3
σ_4	00	01	σ_1	σ_3

TABLE 5.2

CHECK FOR MOORE EQUIVALENT STATES

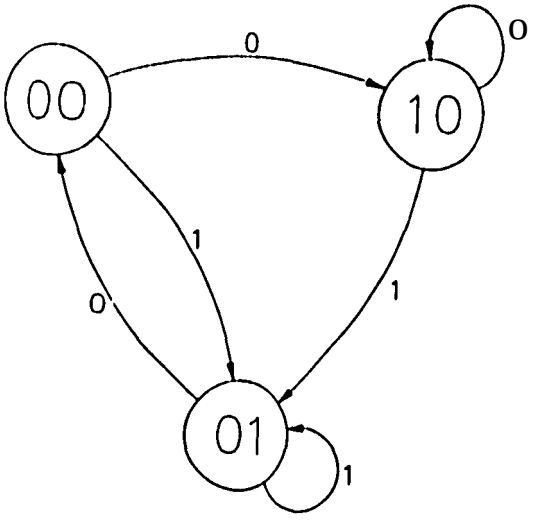


FIGURE 5.3

MINIMIZED MOORE MACHINE

0101001

For the Miller code as Mealy machine (figure 5.1), starting in state σ_1, the input sequence will be converted to:

00 01 00 01 00 10 01

Remembering thnt the output, when going from the one state to the other, will be thc same as the *neu* state, the Moore machine representation (figure 5.2), also starting in σ_1 will convert the input sequence to:

10 01 0001 00 10 01

Figure 5.3. also starting in σ_1 will convert the input sequence to:

10 01 0001 00 10 01

By a proper choice of starting state in the Mealy machine. the first code-words will also correspond. It must be notcd that any state can be chosen as starting state. The first few code words will be different, depending on the amount of memory in the system and of course the starting state.

In the following example a de-free, rate 1/2. $(d, k, C) = (0.2.1)$ code, is converted from a minimal Mealy machine to a Moore mnchine. This specific code was also transmitted over a mobile communication channel.

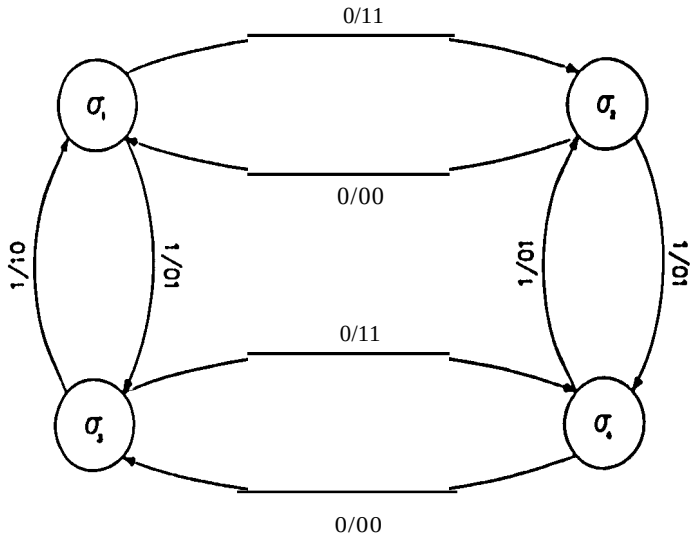


FIGURE 5.4

STATE TRANSmON DIAGRAM OF THE $(d, k, C) = (0.2.1)$

Following figure 5.4. we have: source alphabet $X = (0.1)$. code-alphabet $Z = \{00.01.10.11\}$, state sct $S = \{\sigma_1, \sigma_2, \sigma_3, \sigma_4\}$. source-wont number $K = 2$. code-word number $J = 4$, state numbcr $I = 4$ and code-word length $N = 2$.

The Mealy machine is described by the following matrices:

$$E_1 = \begin{bmatrix} 0&1&0&0 \\ 1&0&0&0 \\ 0&0&0&1 \\ 0&0&1&0 \end{bmatrix}, \quad E_2 = \begin{bmatrix} 0&0&1&0 \\ 0&0&0&1 \\ 1&0&0&0 \\ 0&1&0&0 \end{bmatrix}, \quad \Gamma_1 = \begin{bmatrix} 1&1 \\ 0&0 \\ 1&1 \\ 0&0 \end{bmatrix}, \quad \Gamma_2 = \begin{bmatrix} 0&1 \\ 0&1 \\ 1&0 \\ 1&0 \end{bmatrix}$$

The Moore equivalence FSM thus hove $KI = 8$ states and, following (5.5) and (5.6), the corresponding matrix representations is given by:

$$\tilde{E}_1 = \begin{bmatrix} E_1 0 \\ E_2 0 \end{bmatrix} = \begin{vmatrix} 01000000 \\ 10000000 \\ 00010000 \\ 00100000 \\ 00100000 \\ 00010000 \\ 10000000 \\ 01000000 \end{vmatrix}, \quad \tilde{E}_2 = \begin{bmatrix} 0 & E_1 \\ 0 & E_2 \end{bmatrix} = \begin{vmatrix} 00000100 \\ 00001000 \\ 00000001 \\ 00000010 \\ 00000010 \\ 00000001 \\ 00001000 \\ 00000100 \end{vmatrix}$$

$$\tilde{\Gamma} = \begin{bmatrix} \Gamma_1 \\ \Gamma_2 \end{bmatrix} = \begin{vmatrix} 11 \\ 00 \\ 11 \\ 00 \\ 01 \\ 01 \\ 10 \\ 10 \end{vmatrix}$$

The new Moore machine is presented in figure 5.5. By inspection it is evident that the Moore machine cannot be minimized.

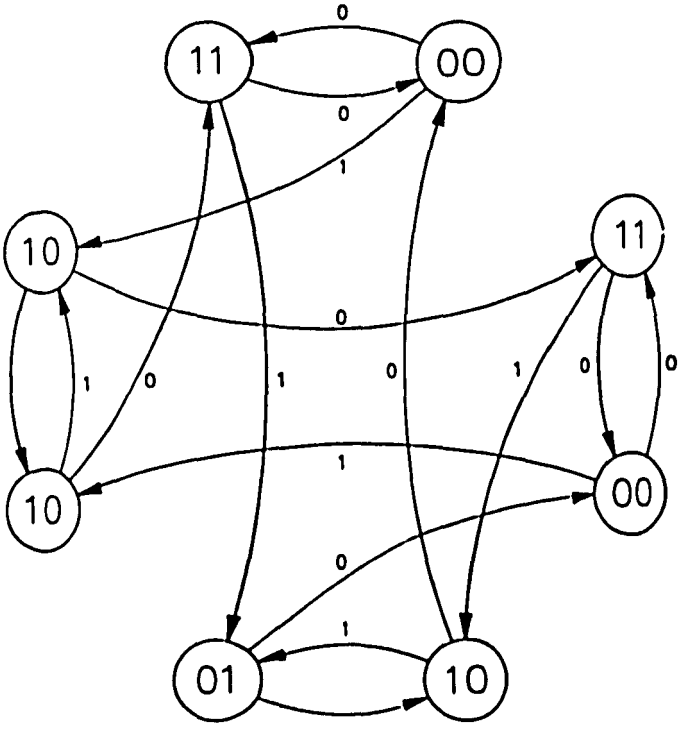


FIGURE 5.5

STATE TRANSmON DIAGRAM OF MOORE MACJIINE

5.2) MAPPING FIXED-LENGTH CODING RULES ON FSM'S

This algorithm is based on an engineering approach to sequential design [4]. Figure 5.6 shows the basic block diagram on which this algorithm is based. The algorithm is called the BW algorithm since we are working backwards; from the circuit diagram to the state diagram.

When figure 5.6 is considered, $D_1, ..., D_m$ is the source bits, $G_1, ..., G_n$ the memory elements and $O_1, ..., O_p$ the output bits ($m, n, p = 1, 2, ..., \infty$). The blocks labelled Next State Decoder and Output Decoder enables respectively correct state transitions and correct system outputs.

By using memory elements with feedback together with the present inputs, a decision can thus be made about the present output. There are five basic steps required in this algorithm:

- First, there must be decided what to remember in the memory elements;
- Set up a block diagram with the desired inputs, i.e. present data bits and inputs from the memory elements;
- A truth table with the present inputs and feedback values as inputs to the system and the desired outputs;
- Next a present state - next state table must be derived from the truth table;
- Finally, if necessary, the minimization rules, as outlined in chapter 3, must be applied for a minimum state diagram.

To show that this algorithm is not only valid for the more credulous binary codes, the following example addresses a multi-level code.

EXAMPLE 5.2

The mapping rule for a RLL, multilevel code is described as follows [28]:

> Every group of two data bits must be mapped onto one five-level symbol, using $2^2 = 4$ unique symbols. Since the sequence must be run-length limited, the remaining five-level symbol must be reserved when the same group of data bits are repeated. When a group is repeated, the symbol 0 is transmitted. If the group is repeated again, the original symbol allocated to the data is transmitted, etc.

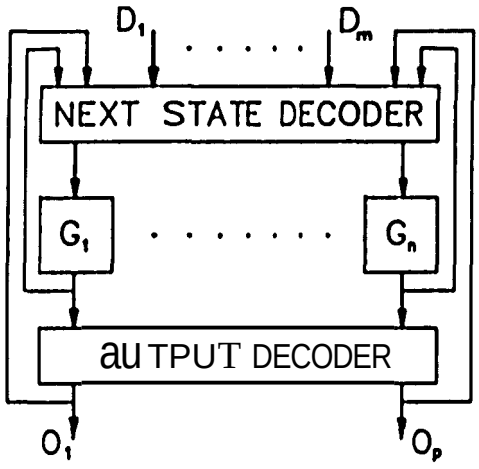


FIGURE 5.6
BASIC BI.OCK DIAGRAM 01' BW AI.GORInIM

THC code mapping with an example of a data stream and corresponding five-level sequence is presented.

CODE MAPPING:	00	=>	-2	=>	a
	01	=>	-1	=>	b
	10	=>	+1	=>	c
	11	=>	+2	=>	d
Whcn the two input bits are repeated:		=>	0	=>	e

DATA:	01	00	10	00	00	01	01	01
CODE SEQUENCE:	b	a	e	a	e	b	e	b

Using the previously outlined steps:

What must be stored in the memory e/cmcnL'i?
The previous code-word, because of the repetition rule in the code description,

Draw a block diagram:
The hlock diagram, figure 5.7, shows the present data bits *(01' D_2)* and previous code wont *($O_{I'V}$, v* being time dependent) as input to the system, with ll five-level code word *(OJ)* as output.

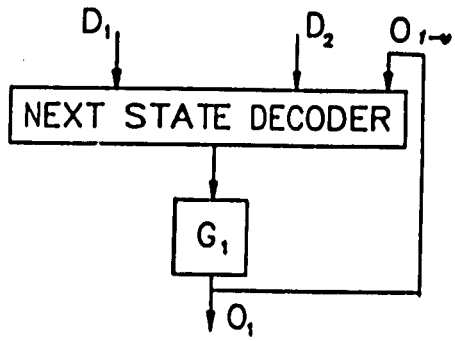


FIGURE 5.7
BLOCK DIAGRAM FOR EXAMPLE 5.2

SCI/ing up the truth table:

Thc truth table is composed of the previously mentioned inputs and nn output, thc code wont. Consider table 5.3, the entries can be explained as follows:

For thc first entry: the present data bits (D_1 D_2) arc 00, the previous code word (O_{1-1}) was a, thus the present output must be e (following the coding rules);
second entry: the present data bits are 00, the previous code word was b, thus the present output must be a;
third entry: the present data bits are 00, the previous code word was c, thus the present output must be a, etc.

It is clear that a e as output will only be present when the previous code word would have been the same as the present code word.

Derive a present state-next state diagram from the truth table:

Consider table 5.4, the present states arc taken from the column designated by C_1 and the next states and outputs nrc taken as the $C_{J,V}$ column. By re-assigning the emeries in C_1 and $C_{J,V}$ so that a becomes state A, b becomes state B ete., we have everything that is necessary for a state diagram: inputs, present states, next states and outputs.

If necessary, minimize the state diagram:

For this example thnt was not even necessary. Table 5.5 shows the state transition table and figure 5.8 shows the state transition diagram.

D_1	D_2	$C_{1\cdot\nu}$	C_1
0	0	a	e
0	0	b	n
0	0	c	a
0	0	d	a
0	0	e	a
0	1	a	b
0	1	b	e
0	1	c	b
0	1	d	b
0	1	c	h
1	0	n	c
1	0	b	c
1	0	c	e
1	0	d	c
1	0	e	c
1	1	n	d
1	1	b	d
1	1	c	d
1	1	d	e
1	1	e	d

TABLE 5.3

rnrrn TABLE FOR EXAMPLE 52

	00	01	10	11	00	01	10	11
A	e	b	c	d	E	B	C	0
n	n	e	c	d	A	E	C	0
C	a	b	e	d	A	B	E	0
0	n	b	c	c	A	B	C	E
E	n	b	c	d	A	B	C	0

TABLE 5.4

STATE TRANSmON TABLE

Input	P.S.	N.S.	Output
00	A	E	e
01	A	n	b
10	A	C	c
11	A	D	d
00	B	A	n
01	B	E	e
10	B	C	c
11	B	D	d
00	C	A	a
01	C	B	b
10	C	E	c
11	C	D	d
00	D	A	n
01	D	n	b
10	D	C	c
11	D	E	e
00	E	A	a
01	E	n	b
10	E	C	c
11	E	D	d

TAlLE5.5

PRESENT STATE - NEXT STATE TABLE

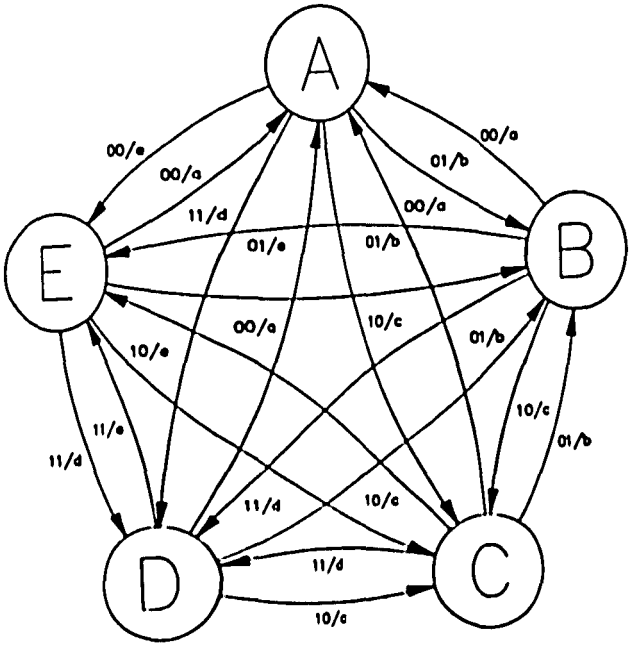


FIGURE 5.8

STArn TRANSmON DIAGRAM

A test string from the state diagram verifies the algorithms validity. staning in state A:

DATA:	01	00	10	00	00	01	01	01
CODE SEQUENCE:	b	a	c	a	c	b	e	b

CHAPTER 6

NEW RESULTS ON (0,k) MODULATION CODES

With the necessary background on RLL codes developed in chapter 4, we have now acquired an infonnation-theoretical knowledge on the key aspects of these sequences, enabling us to venture deeper into this interesting field. As mentioned in chapter 4, the larger the *d* parameter, the lower the code rate R (table 4.1) and the larger the bandwidth when a constant *data* rate is to be achieved over a given channel. When mobile radio channels are considered, the available bandwidth must be used as efficiently as possible, see chapter 2. Hence, to gain clock extraction via modulation codes on mobile radio channels, a small or ultimately no *d* parameter must be compelled on the code stream, Thus, this chapter will look at a special class of *d* and *k* parameters; the event where $d = 0$.

An interesting property of this class of (d, k) codes are that the channel capacities (e) of these codes asymptotical approach I for $k \to \infty$. Further, they can be consnucted with *practical* coding rates of R = *n-iln;* for a relatively large *n,* clock extraction can be gained with a marginal loss in bandwidth. It is thus evident that this class of codes had to be investigated for use on mobile radio channels. At present the well known Manchester, or rate R = *]12, (tl, k,* C) =(0, I, 1) code is used on mobile radio channels, thus further strengthening the idea of investigating this class of codes.

At the moment only two (0. k) codes are widely used, the Group-Coded Recording (GCR), rate R = 4/5, (0, 2) code (3) and the rate R = 8/9. (0, 3) code [29]. Doth these codes are used in a large variety of magnetic tape products. In the OCR code, 4 user bits are uniquely represented by 5 channel bits, while 8 user bits are mapped on 9 channel bits for the (0, 3) code. According to table 4.1, the capacity of a sequence with no runs of more than two "zeros" Is C(O. 2) ≃ 0.879, while the channel capacity of a sequence with no runs of more than three "zeros" is C(O. 3) ≃ 0.946. Jlence, the efficiencies of the aforementioned two codes are respectively η = 91 % and η = 94%.

As mentioned earlier, for a larger codeword length. n, an inclination to a smaller bandwidth can be achieved. With this in mind, a rate R = 11/12, (d, k) = (0.3) code was developed. using a newly developed time saving algorithm, the TS algorithm, The TS algorithm can also be used in future when other $(0, k)$ codes need to be developed. The efficiency of the aforementioned code is $\eta = {}^{11}/_{12}/C(0, 3) \simeq 0.968$, but more important. the bandwidth is almost the same as that of the uncoded data. Since the codeword length is so large, 2^{11} bits. it was decided to present the code table for the encoder on a floppy disk, in the cover of this thesis. with filename 0_3.TXT. The code table for the decoder can be found on the same disk, with filename DECO_3.TXT.

As Immink [3] points out. the dk constraints define a number of channel states; the states of a Markov model for a given dk constraint. The crucial problem for the creation of fixed-length $(d. k)$ codes of minimum length. of which $(0. k)$ codes is a subset. is to find a subset of states. referred to as *principal states,* of which there exist a sufficient number of sequences of length n terminating at other principle states. The existence of a set of principal states can also be used to verify the existence of a code with a specified rote and codeword length. Franaszek [19] developed a recursive search technique for determining the existence of a set of principal states through operations on the connection matrix.

The advantage of the TS algorithm over the method developed by Franaszek is that a code book can be obtained while searching for the code existence. with a negligible loss in speed. A timesaving with the TS algorithm for large n is also anticipated, since the Franaszek algorithm involves plenty of multiplications (raising matrices to powers).

Since the developed TS algorithm was so powerful and easy to implement on a digital computer, it was decided to do a complete search for the .class of $(0, k)$ codes in the range $1 \leq k \leq 20$ and $2 \leq n \leq 20$.

When something new is developed, it is wise and imperative, to confinn its validity. It was thus decided to first check the algorithm by duplicating the known results for the GCR and rate 8/9, *(ti. k)* = (0.3) code. For the OCR code; from the 32 possible unconstrained combinations of 5 bits, 15 were eliminated, leaving 17 valid codewords, From the 512 possible unconstrained combinations of 9 bits, 219 can be eliminated leaving 293 valid codewords in the (0, 3) code. This Information, which can also be obtained from Immink [3], serves as confirmation of the TS algorithm. since the exact same number of valid codewords was obtained using the TS algorithm (see tables 6.3 and 6.4).

6.1) Tim TS ALGORITIIM

Pill is section will be devoted to a description of the TS algorithm. Consider the following representation of a candidate codeword. consisting of *n* bits; numbered from the least significant bit (LSB). 0, to the most significant bit (MSB), *n-l:*

	II-J	*n-2*	, ...	*k-1*		0
Potential codewords	0	1	, ...,	0	, ...,	0

With the above representation in mind, the algorithm can be described step by step in the following way:

When searching for valid codewords, the first 2^{n-2} n·bit binary words (in normal counting or hexicographical order) arc ignored, since we do not include codewords ending with more than one zero. In other words, if bit *n-I* is zero (the MSm. bit *n-2* is not allowed to be zero: this was done since experimentation showed that the largest number of valid codewords can be obtained in the least time with this restriction;

Next. bits 0 to *k-1* of the remaining candidate codewords are searched for a violation of *k* consecutive zeros. simultaneously with zeros that violate the *k* constraint from bit *n·2* to bit *k*;

- Finally, the number of valid codewords obtained in the previous search must be at least *1'./* for the (0. *k*) code to be valid. since we arc dealing with a code rate of R = *n-i!«*.

The example to follow will give the reader a feeling how this algorithm can be implemented.

	2	1	0
0	0	0	0
1	0	0	1
2	0	1	0
3	0	1	1
4	1	0	0
5	1	0	1
6	1	1	0
7	1	1	1

TABLE 6.1

CANDIDATE CODEWORDS FOR EXAMPLE 6.1

EXAMPLE 6.1

In this example we shall construct a rate R =2/3, *(d, k)* = (0, 2) code. Admittedly this example docs not utilize the full power of the TS algorithm, but will serve a tutorial purpose.

Using the previously outlined steps:

Ihe first 2^{n-2} potentia! codewords must be discarded:

With n =3, the first 2^{3-2} =2 words are ignored, since bits 2 and 1 are both zeros.

Bits 0 to k-I = 1 must be searched for a violation of k zeros; simultaneously with a search for a violation of k zeros from bit n-2 to bit k:

Only word 4, in table 6.1, violate k zeros at bit positions 0 and 1, with no violation of k zeros between bits n-t =2 and k-1 = 1.

For a rate R =n-ltn; (d, k) = (0, 2) code to be valid, there must be at least 2^{n-1} =4 valid code words:

Since words 0, 1 and 4 arc invalid, the remaining words 2, 3, 5, 6 and 7 are valid, hence, there arc 5 valid and 3 unvalid codewords, and since 2^{3-1} =4 < 5, a rate 2/3, (d, k) = (0, 2) code can be constructed with 5 potential codcwords. One of the codewords may be discarded, leaving 4 unique codewonJs to be mapped on by the 2^{n-1} = 2^2 =4 data bits.

rille search with the TS algorithm was performed on an 10M AT compatible with a 16 MIIz clock. Figures 6.1 to 6.2 compare the time in seconds versus n for the 1'8 algorithm and an exhaustive search for valid codewords (i,e. exhaustively checking binary words 0 to 2^n-1 (or violation of the k constraint), with $k = 4$, 5 and $14 \leq n \leq 20$. Figures 6.3 and 6.4 show 3D graphs for k versus n and the time in seconds for, respectively, the exhaustive search and the TS algorithm.

The graphs show a constant improvement in speed for a fixed value of k. The TS algorithm achieves more than twice the speed of the exhaustive search. The gain of the TS algorithm is directly proportional to n; the larger n, the better the improvement on an exhaustive search.

From the 2^n possible unconstrained combinations of n bits, the valid codewords in the first 2^n-1 unconstrained words, designated by No, were recorded, together with the valid codewords in the second 2^n-1 unconstrained words, designated by N_1. Figures 6.5 to 6.8 show these valid words (and the sum $No+N_1$) for different values of n, The reason why No and N_1 were recorded, is to confinn the anticipation of more valid codewords in the second 2^n-1 unconstrained words, NJ' Figure 6.9 shows a comprehensive 3D graph with $No+N_J$, $1 \leq k \leq 20$ and $2 \leq n \leq 20$.

Table 6.2 to 6.21 show the efficiencies and number of valid codewords for $1 \leq k \leq 20$ and $2 \leq n \leq 20$. When, for a specific k and n value, a code does not exist, no entries are listed in the efficiency (η) column of tables 6.2 to 6.21. There are two possible reasons for this; firstly, the search yielded fewer than 1'.) valid codewords, and secondly, since these are block codes, the k constraint could not be accomplished. This happens when $k > n$.

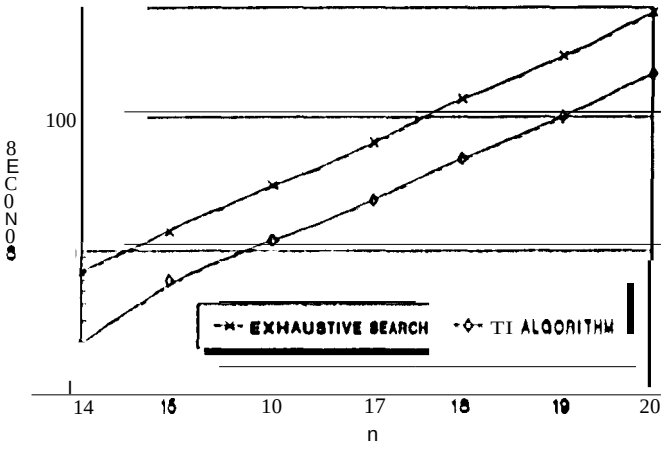


FIGURE 6.1

TIME COMPARISON OF EXHAUSTIVE SF.ARCH AND TS ALGORITHM FOR $k = 4$

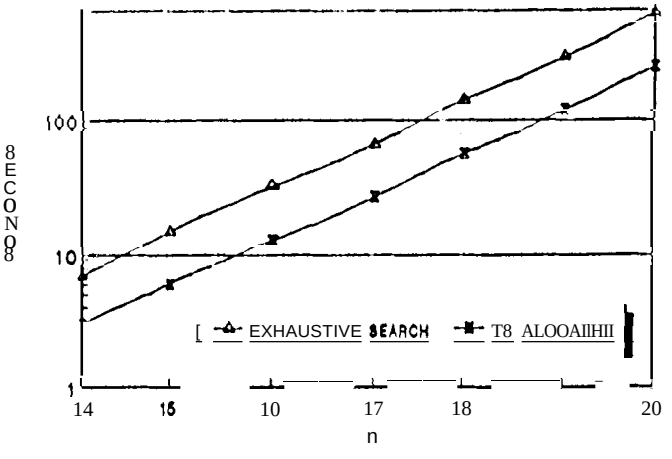


FIGURE 6.2

TIME COMPARISON OF EXHAUSTIVE SEARCH AND TS ALGORITHM FOR $k = 5$

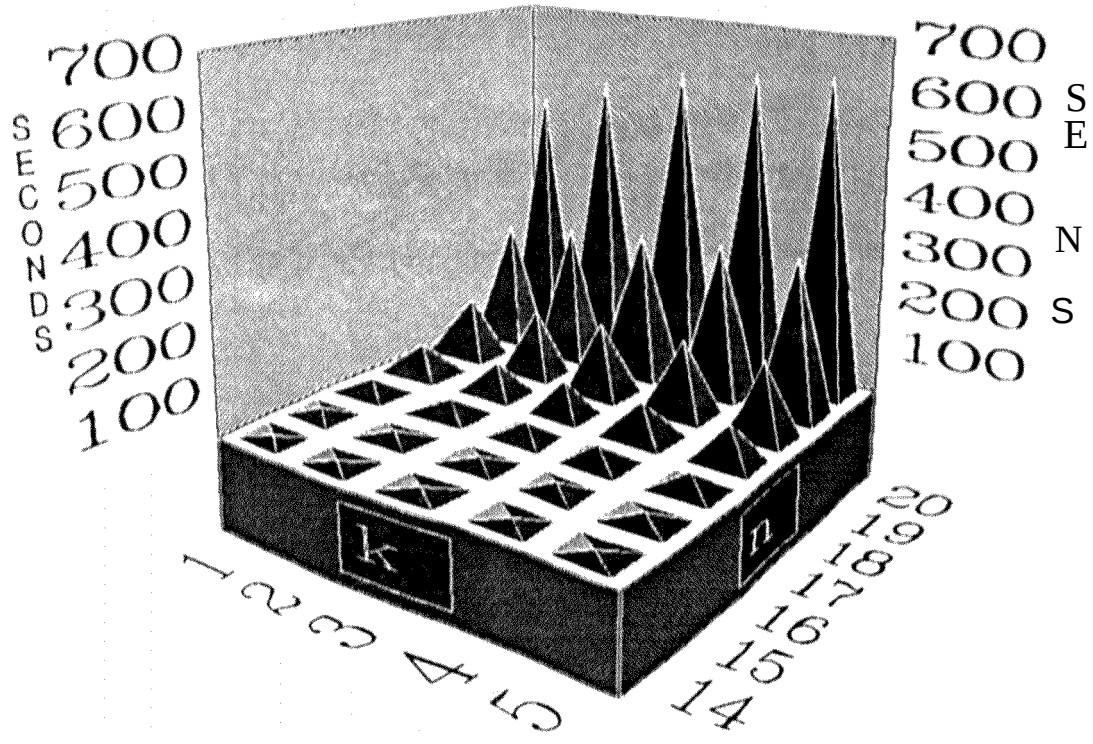


FIGURE 6.3

FOR EXHAUSTIVE SEARCH wml 1 k 5, 14 n

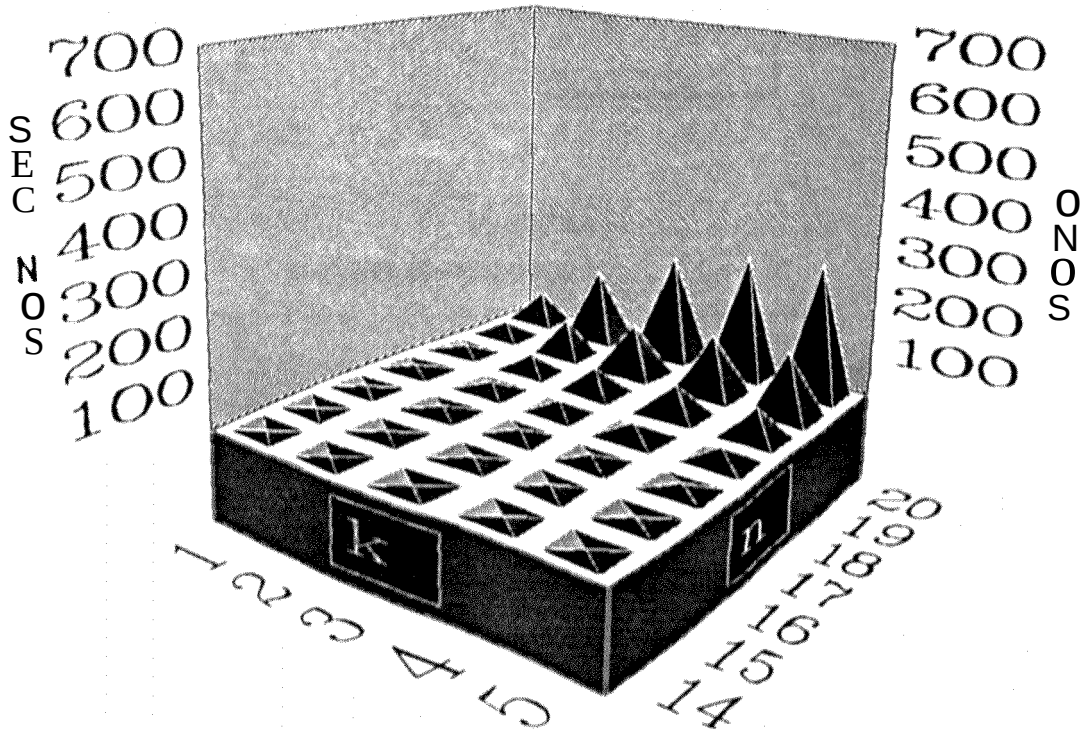


FIGURE 6.4

TIME FOR TS ALGORITHM WITH 1 $k \le 5$, $14 \le n \le 20$

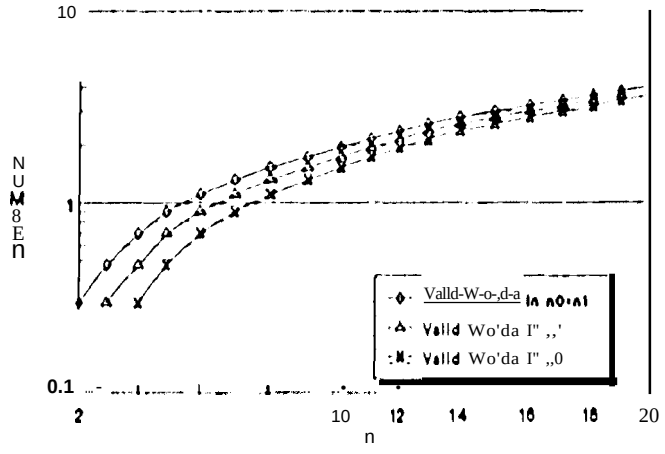


FIGURE 6.5

VALID CODEWORDS FOR $k = 1$, $2 \le n \le 20$

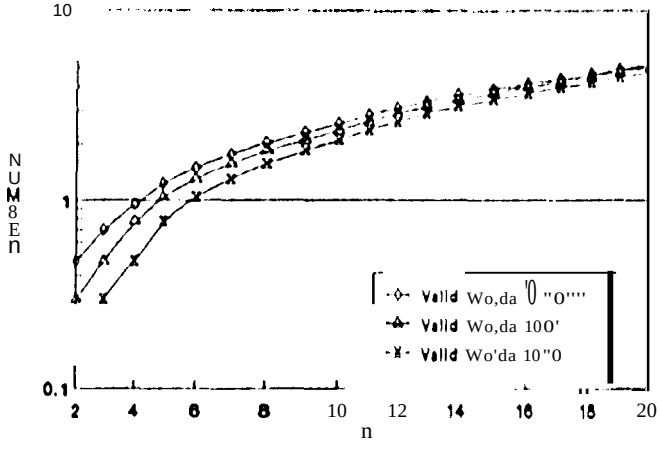


FIGURE 6.6

VALID CODEWORDS FOR $k = 2$, $2 \le n \le 20$

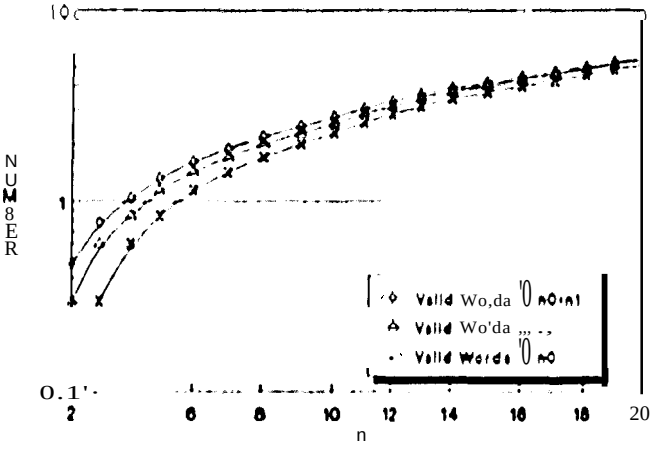


FIGURE 6.7

VALID CODEWORDS FOR $k = 3$, $2 \le n \le 20$

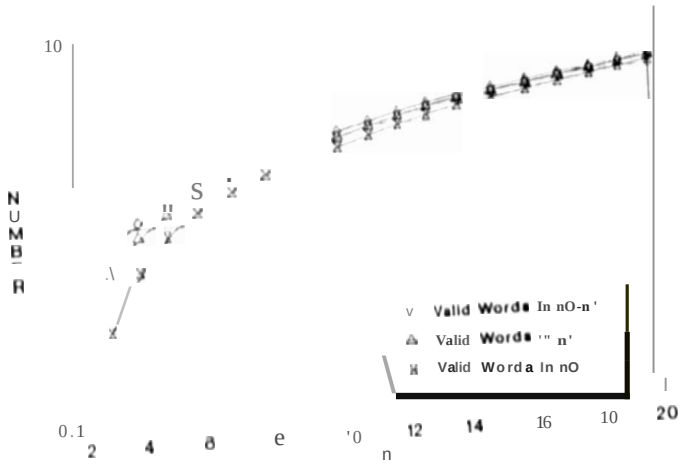


FIGURE 6.8
VALID CODEWORDS FOR k = iI, 2 S 1\ 20

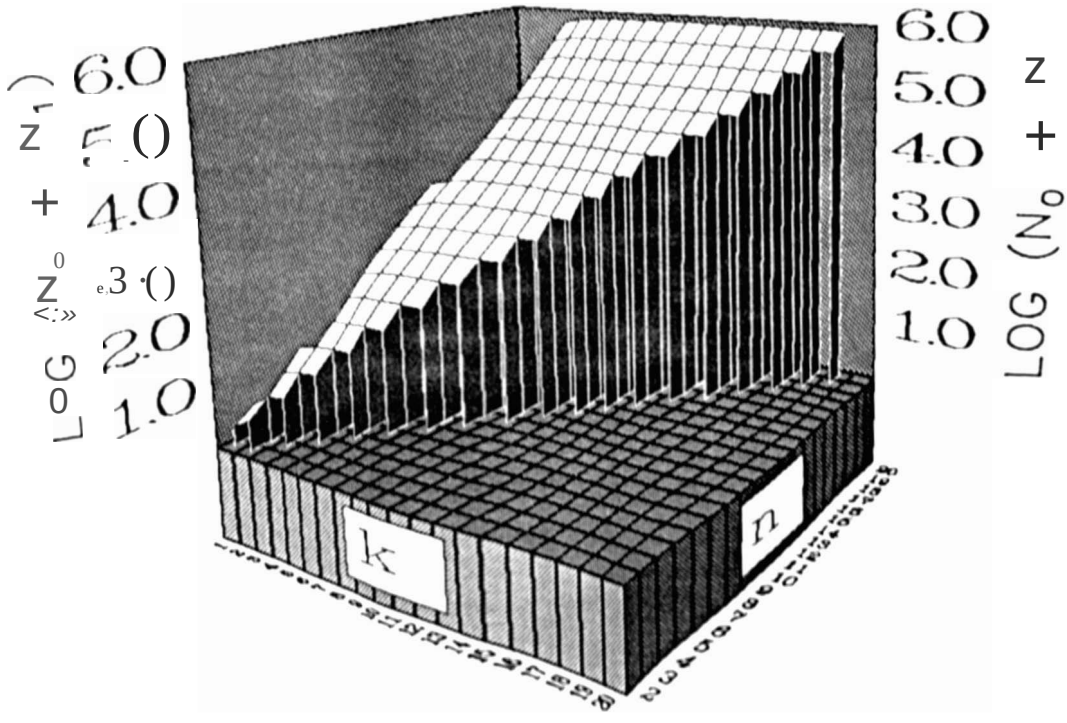


FIGURE 6.9
YALID CODEWORDS ur AI ED w rrn TS ALGORITHM \' R 1 ≤ k ≤ 20, n O

n	η	No	HI	No+NI
2	0.720	1	1	2
3	0.961	1	2	3
4	.	2	3	5
5	.	3	5	8
6	.	5	8	13
7	.	8	13	21
8	.	13	21	34
9	.	21	34	55
10	.	34	55	89
11	.	55	89	144
12	.	89	144	233
13	.	144	233	377
14	.	233	3n	610
15	.	377	610	987
16	.	610	987	1597
17	.	987	1597	2584
18	.	1597	2584	4181
19	.	2584	4181	6765
20	.	4181	6765	10946

TABLE 6.2

$k = 1$, C(O, 1) = 0.694242

n	η	No	HI	No+NI
2	0.569	1	2	3
3	0.758	2	3	5
4	0.853	3	6	9
5	0.910	6	11	17
6	.	11	20	31
7	.	20	37	57
8	.	37	68	105
9	.	68	125	193
10	.	125	230	355
11	.	230	423	653
12	.	423	n8	1201
13	.	778	1431	2209
14	.	1431	2632	4063
15	.	2632	4841	7473
16	.	4841	8904	13745
17	.	8904	163n	25281
18	.	163n	30122	46499
19	.	)0122	55403	85525
20	.	55403	101902	157305

TABLE 6.3

$k = 2$, C(O, 2) = 0.879146

n	η	No	Nl	$No+Nl$
2	-	1	2	3
3	0.704	2	4	6
4	0.792	4	7	11
5	0.844	7	14	21
6	0.880	14	27	41
7	0.905	27	52	79
8	0.924	52	100	152
9	0.938	100	193	293
10	0.950	193	372	565
11	0.960	372	717	1089
12	0.968	717	1382	2099
13	-	1382	2664	4046
14	-	2664	5135	7799
15	.	5135	9898	15033
16	.	9898	19079	28977
17	-	19079	36776	55855
18	-	36776	70888	107664
19	-	70888	136641	207529
20	-	136641	263384	400025

TABLE 6.4

$k = 3$, C(O, 3) = 0.946777

n	η	No	Nl	$No+Nl$
2	-	1	2	3
3	.	2	4	6
4	0.769	4	8	12
5	0.820	8	15	23
6	0.854	15	30	45
7	0.878	30	59	89
8	0.897	59	116	175
9	0.911	116	228	344
10	0.922	228	448	676
11	0.932	448	881	1329
12	0.939	881	1732	2613
13	0.946	1732	3405	5137
14	0.952	3405	6694	10099
15	0.957	6694	13160	19854
16	0.961	13160	25872	39032
17	0.965	25872	508(13	76735
18	0.968	50863	99994	150857
19	0.971	99994	196583	*2965n*
20	0.974	196583	386472	583055

TABLE 6.5

$k = 4$, C(O. 4) = 0.975225

n	"	No	NI	No+Nl
2	.	1	2	3
3	.	2	4	6
4	.	4	8	12
5	0.809	8	16	24
6	0.843	16	31	47
7	0.867	31	62	93
8	0.885	62	123	185
9	0.899	123	244	367
10	0.910	244	484	728
11	0.920	484	960	1444
12	0.927	960	1904	2864
13	0.934	1904	srn	5681
14	0.939	3777	7492	11269
15	0.944	7492	14861	22353
16	0.948	14861	29478	44339
17	0.952	29478	58472	87950
18	0.955	58472	115984	174456
19	0.958	115984	230064	346048
20	0.961	230064	456351	686415

TABLE 6.6

$k = 5$, C(O, 5) = 0.988109

n	"	No	Nl	No+Nl
2	-	1	2	3
3	-	2	4	6
4	.	4	8	12
5	.	8	16	24
6	0.838	16	32	48
7	0.862	32	63	95
8	0.880	63	126	189
9	0.894	126	251	377
10	0.905	251	500	751
11	0.914	500	996	1496
12	0.922	996	1984	2980
13	0.928	1984	3952	5936
14	0.933	3952	7872	11824
15	0.938	7872	15681	23553
16	0.942	15681	31236	46917
17	0.946	31236	62221	93457
18	0.949	62221	123942	186163
19	0.952	123942	246888	370830
20	0.955	246888	·191792	738680

TABLE 6.7

$k = 6$, C(O, 6) = 0.994192

n	η	No	NI	No+NI
2	.	1	2	3
3	.	2	4	6
4	.	4	8	12
5	.	8	16	24
6	.	16	32	48
7	0.859	32	64	96
8	0.877	64	127	191
9	0.891	127	254	381
10	0.902	254	507	761
11	0.911	507	1012	1519
12	0.919	1012	2020	3032
13	0.925	2020	4032	6052
14	0.931	4032	8048	12080
15	0.936	8048	16064	24112
16	0.940	16064	32064	48128
17	0.943	32064	64001	96065
18	0.947	64001	127748	191749
19	0.950	127748	254989	382737
20	0.952	254989	508966	763955

TABLE 6.8

k =7, C(O, 7) = 0.997134

n	η	No	N_I	$No+N_J$
2	.	1	2	3
3	.	2	4	6
4	.	4	8	12
5	-	8	16	24
6	.	16	32	48
7	-	32	64	96
8	0.876	64	128	192
9	0.890	128	255	383
10	0.901	255	510	765
11	0.910	510	1019	1529
12	0.917	1019	2036	3055
13	0.924	2036	4068	6104
14	0.929	4068	8128	12196
15	0.934	8128	16240	24368
16	0.9.18	16240	32448	48688
17	0.942	32448	64832	97280
18	0.945	64832	129536	194368
19	0.948	129536	258817	388353
20	0.951	258817	517124	775941

TABLE 6.9

k =8, C(O, 8) = 0.998578

n	η	No	Nt	$No+N_t$
2	.	1	2	3
3	.	2	4	6
4	.	4	8	12
5	.	8	16	24
6	.	16	32	48
7	.	32	64	96
8	.	64	128	192
9	0.889	128	256	384
10	0.900	256	511	767
11	0.909	511	1022	1533
12	0.917	1022	2043	3065
13	0.923	2043	4084	6127
14	0.929	4084	8164	12248
15	0.933	8164	16320	24484
16	0.938	16320	32624	48944
17	0.941	32624	65216	97840
18	0.945	65216	130368	195584
19	0.948	130368	260608	390976
20	0.950	260608	520960	781568

TABLE 6.10

$k = 9$, C(O, 9) = 0.999292

n	η	No	Nt	$No+Nt$
2	.	1	2	3
3	.	2	4	6
4	.	4	8	. 12
5	.	8	16	24
6	.	16	32	48
7	.	32	64	96
8	.	64	128	192
9	.	128	256	384
10	0.900	256	512	768
11	0.909	512	1023	1535
12	0.916	1023	2046	3069
13	0.923	2046	4091	6137
14	0.928	4091	8180	12271
15	0.933	8180	16356	24536
16	0.937	16356	32704	49060
17	0.941	32704	65392	98096
18	0.944	65392	130752	196144
19	0.947	130752	261440	392192
20	0.950	261440	522752	784192

TABLE 6.11

$k = 10$, C(O, 10) = 0.999647

n	η	No	N,	No+N,
2	.	1	2	3
3	.	2	4	6
4	.	4	8	12
5	.	8	16	24
6	.	16	32	48
7	.	32	64	96
8	.	64	128	192
9	.	128	256	384
10	.	256	512	768
11	0.909	512	1024	1536
12	0.916	1024	2047	3071
13	0.923	2047	4094	6141
14	0.928	4094	8187	12281
IS	0.933	8187	16372	24559
16	0.937	16372	32740	49112
17	0.941	32740	65472	98212
18	0.944	65472	130928	196400
19	0.947	130928	261824	392752
20	0.950	261824	523584	785408

TABLE 6.12

$k = 11$, C(O, 11) = 0.999824

n	η	No	NI	No+N_1
2	-	1	2	3
3	.	2	4	6
4	-	4	8	12
5	.	8	16	24
6	.	16	32	48
7	.	32	64	96
8	.	64	128	192
9	.	128	256	384
10	.	256	512	768
11	-	512	1024	1536
12	0.916	1024	2048	3072
13	0.923	2048	4095	6143
14	0.928	4095	8190	12285
15	0.933	8190	16379	24569
16	0.937	16379	32756	49135
17	0.941	32756	65508	98264
18	0.944	65508	131008	196516
19	0.947	131008	262000	393008
20	0.950	262000	523968	785968

TABLE 6.13

$k = 12$, C(O, 12) = 0.999912

n	η	No	HI	No+NI
2	.	1	2	3
3	.	2	4	6
4	.	4	8	12
5	.	8	16	24
6	.	16	32	48
7	.	32	64	96
8	.	64	128	192
9	.	128	256	384
10	.	256	512	768
11	.	512	1024	1536
12	.	1024	2048	3072
13	0.923	2048	4096	6144
14	0.928	4096	8191	12287
15	0.933	8191	16382	24573
16	0.937	16382	32763	49145
17	0.941	32763	65524	98287
18	0.944	65524	131044	196568
19	0.947	131044	262080	393124
20	0.950	262080	524144	786224

TA8LE6.14

$k = 13$, C(O, 13) = 0.999956

n	η	No	HI	No+N$_I$
2	.	1	2	3
3	.	2	4	6
4	.	4	8	12
5	.	8	16	24
6	.	16	32	48
7	.	32	64	96
8	.	64	128	192
9	.	128	256	384
10	.	256	512	768
11	.	512	1024	1536
12	.	1024	2048	3072
13	.	2048	4096	6144
14	0.928	4096	8192	12288
15	0.933	8192	16383	24575
16	0.937	16383	32766	49149
17	0.941	32766	65531	98297
18	0.944	65531	131060	196591
19	0.947	131060	262116	393176
20	0.950	262116	524224	786340

TABLE 6.15

$k = 14$, C(O, 14) = 0.999978

n	η	No	N_I	$No+N_I$
2	-	1	2	3
3	.	2	4	6
4	.	4	8	12
5	.	8	16	24
6	.	16	32	48
7	.	32	64	96
8	.	64	128	192
9	.	128	256	384
10	-	256	512	768
11	-	512	1024	1536
12	.	1024	2048	3072
13	-	2048	4096	6144
14	.	4096	8192	12288
15	0.933	8192	16384	24576
16	0.937	16384	32767	49151
17	0.941	32767	65534	98301
18	0.944	65534	131067	196601
19	0.947	131067	262132	393199
20	0.950	262132	524260	786392

TABLE 6.16

$k = 15$, C(O, 15) =0.999989

n	η	No	N_I	$No+N_I$
2	.	1	2	3
3	-	2	4	6
4	-	4	8	12
5	-	8	16	24
6	-	16	32	48
7	-	32	64	96
8	-	64	128	192
9	-	128	256	384
10	.	256	512	768
11	.	512	1024	1536
12	.	1024	2048	3072
13	-	2048	4096	6144
14	.	4096	8192	12288
15	.	8192	16384	24576
16	0.937	16384	32768	49152
17	0.941	32768	65535	98303
18	0.944	65535	131070	196605
19	0.947	131070	262139	393209
20	0.950	262139	524276	786415

TAnt.B 6.17

k =16, C(O, 16) =0.999994

n	η	No	NI	No+NI
2	.	1	2	3
3	.	2	4	6
4	.	4	8	12
5	.	8	16	24
6	.	16	32	48
7	.	32	64	96
8	.	64	128	192
9	.	128	256	384
10	.	256	512	768
11	.	512	1024	1536
12	.	1024	2048	3072
13	.	2048	4096	6144
14	.	4096	8192	12288
15	.	8192	16384	24576
16	.	16384	32768	49152
17	0.941	32768	65536	98304
18	0.944	65536	131071	196607
19	0.947	131071	262142	393213
20	0.950	262142	524283	786425

TAnLE6.18

$k = 17$, C(O, 17) = 0.999997

n	η	No	NI	No+NI
2	.	1	2	3
3	.	2	4	6
4	.	4	8	12
5	.	8	16	24
6	.	16	32	48
7	.	32	64	96
8	.	64	128	192
9	.	128	256	384
10	.	256	512	768
11	.	512	1024	1536
12	.	1024	2048	3072
13	.	2048	4096	6144
14	.	4096	8192	12288
15	.	8192	16384	24576
16	.	16384	32768	49152
17	.	32768	65536	98304
18	0.944	65536	131072	196608
19	0.947	131072	262143	393215
20	0.950	262143	524286	786429

TABLE 6.19

$k = 18$, C(O, 18) = 0.999999

n	▮▮	No	NI	No+NI
2	.	1	2	3
3	-	2	4	6
4	-	4	8	12
5	.	8	16	24
6	.	16	32	48
7	.	32	64	96
8	.	64	128	192
9	.	128	256	384
10	.	256	512	768
11	.	512	1024	1536
12	.	1024	2048	3072
13	.	2048	4096	6144
14	.	4096	8192	12288
15	-	8192	16384	24576
16	.	16384	32768	49152
17	.	32768	65536	98304
18	-	65536	131072	196608
19	0.947	131072	262144	393216
20	0.950	262144	524287	786431

TABLE 6.20

k =19, C(O, 19) =0.999999

n	▮▮	No	NI	No+NJ
2	-	1	2	3
3	-	2	4	6
4	.	4	8	12
5	.	8	16	24
6	.	16	32	48
7	-	32	64	96
8	.	64	128	192
9	-	128	256	384
10	-	256	512	768
11	-	512	1024	1536
12	.	1024	2048	3072
13	.	2048	4096	6144
14	.	4096	8192	12288
15	.	8192	16384	24576
16	.	16384	32768	49152
17	.	32768	65536	98304
18	.	65536	131072	196608
19	.	131072	262144	393216
20	0.950	262144	524288	786432

TABLE 6.21

k =20, C(O, 20) =0.999999

CHAPTER 7

EXPERIMENTAL SET UP

With the theoretical background and theoretical results concluded, the experimental set up and experimental results obtained can be discussed. One might venture the opinion that the experimental results ratified this study; the results were truly meaningful. This chapter will give a system level (block diagram) discussion of the apparatus designed and developed to assist the experimental observations, while Appendix A comprehend the circuit diagrams and a confinnative discussion why certain components were used in the designs.

Three basic designs are presented, the "flagship" of which is an JnM PC based, programmable DSP finite-state machine generator, which consist of a Texas lnstruments TMS 32010 digital signal processor with a potential of sixteen 16-hit programmable *input-output* (I/O) ports. The PC act as an operating system for the DSP processor board hy transferring pre-compiled program memory to static RAM common to the TMS and PC. This method ensures a quick and easy alternative to burning a ROM for every small change in the program.

TIle second design consists of another PC based system incorporating the 8255 *programmable pcriphera! interface* (PPJ) and 8254 *programmable event counter* (PEC), both from Intel. This design assists the nSf> processor board in some essential timing

functions and serve as baseband decoder for bandpass channel experiments conducted. The concourse of the previously mentioned designs will be made evident at a later stage in this chapter.

Since this project concerned *mobile* experiments. it was impractical to implement the aforementioned PC apparatus in a vehicle to carry out the desired measurements. TIle best solution to this problem was to construct a dedicated piece of apparatus which could be interfaced with existing mobile equipment. The apparatus consisted of a programmable finite-state machine generator, an encoder, which could be programmed to test various modulation codes over a mobile VHF channel. This encoder was designed with existing modems in mind, which was already adopted for usc in a mobile environment (a suitable power supply and connectors were available for existing mobile radio transmitters), and had sockets available for plug in modules, like the encoder. A "plug in" decoder module was also designed for usc with the stationary receiving end modem.

7.1) DESIGN DESCRIPTIONS

As previously mentioned three designs arc presented. Figure 7.1 presents a block diagram of the IBM PC based, programmable DSP finite-state machine generator. Between the PC, TMS and the static memory on thc card there are tri-state buffers. These are needed to prevent any clashes on the bus of either processor; when the PC is transferring program memory to the static RAM, the TMS buffcr is in tri-state mode; whcn the TMS is reading program memory. the PC buffer is tri-stated, The concept of tri-state will be discussed in Appendix A.

This unit was used as encoder for measuring spectra of various modulation codes; the spectra were necessary to decide which modulation codes were to be considered for usc over mobile communication channels. Chapter 8 deals with the results obtained in this investigation.

Since modulation codes have rates $R = m/n < 1$, the data clock had to be divided by nand multiplied by m for correct coding. A programmable divide by n and multiply by m generator had to be developed. This was realized with an Intel 8254 PEC and a 4046 CMOS *phase-locked loop* (PLI.).

111e bandpass experiments required an encoder, the DSP FSM, and a decoder to pcrfonn these experiments. The 8255 PPI enabled the PC to act as such a decoder.

Since spectra were measured at 1200 baud, it was decided to verify the usc of the super-fast TMS 320ClO. In othcr words, to check if the TMS 320CIO was not an overkill for measuring spectra at such low data rates, 111C spectrum of a rate 1/2, $(tl, k) = 0,3)$ code was measured with the TMS and PC, respectively depicted in figure 7.2 (a) and (b).

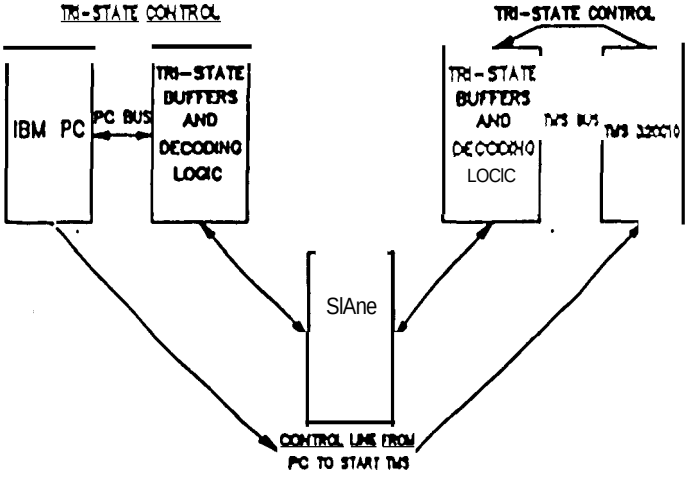


FIGURE 7.1

BLOCK DIAGRAM OF TMS 320C10 DSP CARD

Admittedly, the DSP FSM spectrum looks better; the symbols generated with the DSP FSM arc almost jitter free when leaving the encoder (thc smoother spectra), while the slower PC variation induced pulscwidth variations on the channel bits, not a desired property!

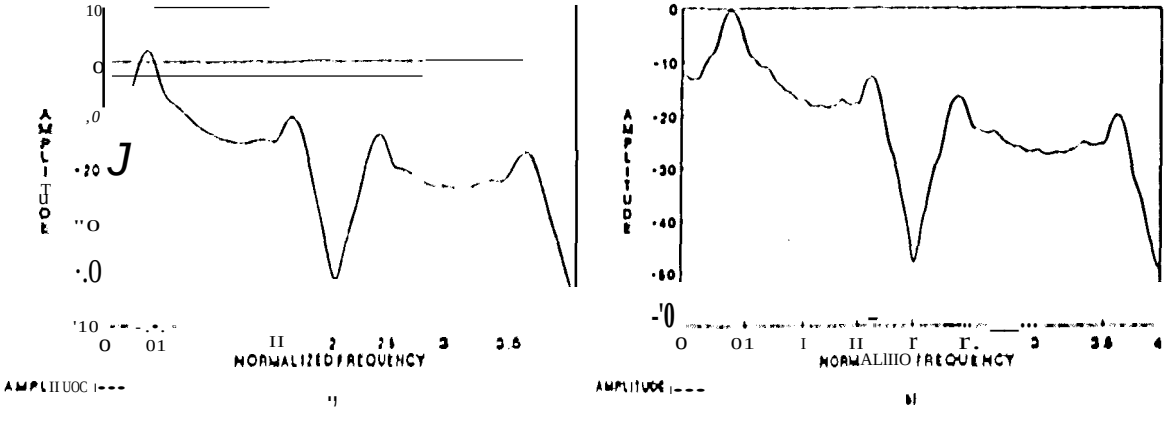


FIGURE 7.2

SPECTRA OF MILLER CODES

a) MEASURED WITH THE DSP FSM b) MEASURED WITH AN 10M PC

'111C block diagram of the PPI-PEC unit is illustrated in figure 7.3.

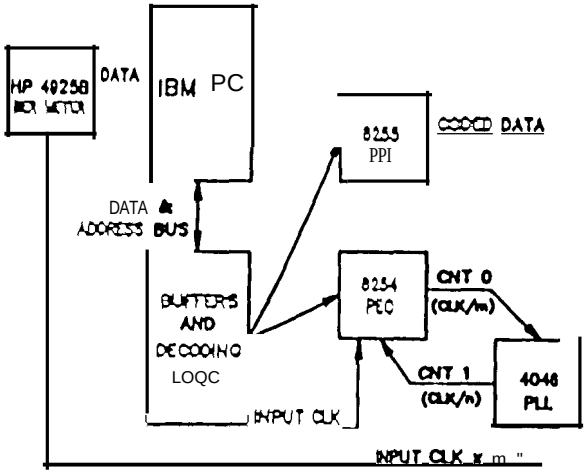


FIGURE 7.3

nl.OCK DIAGRAM FOR TIMER CARD

Since the well-known Intel 8751 microcontrollcr is so versatile, the mobile coders basically consist of this component, a divide-by-a, multiply by *m* circuit and a RS 232 - TTL - RS 232 converter. TIle mobile encoder and decoder arc the same circuit, but programmed differently. lienee, figure 7.4 shows one block diagram for both these units.

TIle equipment developed in this study can be used for at least a few generations of post-graduate studies. For instance, mobile communications, at present in the RSA. operate at]200 to 4800 baud, which is well within reach of the DSP card to do realistic frequency domain investigations. In addition, the mobile coders are fully programmable and can be used in future to implement mobile experiments; if error correcting codes are investigated. this error correcting codes can be programmed in the mobile-coders, and real-time tests cnn be conducted.

7.2) EXPERIMENTAL SET lit)

'This section will he devoted to a block diagrammatic description of the experimental set up for thc bandpass experiments and the mohile experiments.

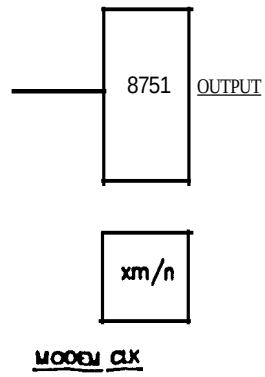


FIGURE 7.4

BLOCK DIAGRAMS FOR MOBILE COOESS

The bandpass experiments were conducted as shown in figure 7.5. An liP 49258 BER meter [30] was used to generate a $2^9\_1 = 511$ PN-scqucnce. 'ntis PN-scqucnce was coded with a pre-programmed modulation code in the DSP FSM generator and transmitted through lowpass, highpass and bandpass filters with various cut off frequencies and slopes. The liP 49258 compares the decoded data, received from the I'C, with the transmitted data until a DER of 10^{-1} is achieved. The same PC can be used for both the encoder and decoder, since the DSP FSM runs, after being started, independent from the PC. TIle results obtained with these experiments arc presented in chapter 8.

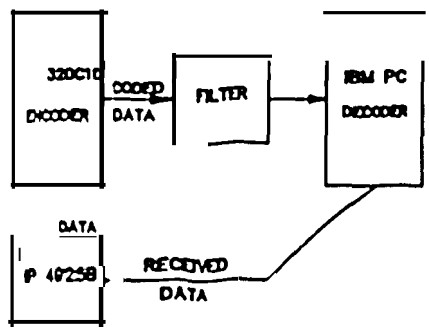


FIGURE 7.5

BLOCK DIAGRAM FOR BANDPASS EXPERIMENTS

Mohile experiments were conducted with an experimental set up as shown in figure 7.6. Again the lIP 49258 nER mcter was used as data source for the mohile encoder. which was installed in a vehicle. Since only an analog radio was available, indirect modulation of the carrier had to be accomplished; the modern converts the digital coded signal to a sinusoidal

analog signal. '111e modem could be programmed to either elght-phase-shlft-keying or four-phase-shift-keying: the output was then frequency translated to the VIIF region and transmitted to the base station, situated at the RAU.

The process was reversed for the received signal. 111e received VIIF data was converted back to a phase-shift-keyed baseband signal and decoded by the decoder, where error-recording equipment [7J recorded the gap recording of the received data. 111c gap recording and related issues, together with the results obtained with this experiments, will be discussed in chapter9.

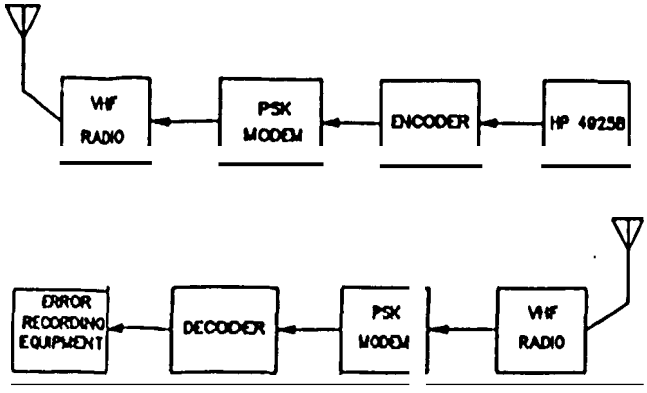


FIGURE 7.6
BLOCK DIAGRAM FOR MOBILE EXPERIMENTS

CHAPTER 8

FREQUENCY DOMAIN INVESTIGATION OF MODULATION CODES

The communication systems engineer is often concerned with the signal location in the *frequency domain* and the signal bandwidth rather than the time transient analysis: this is, for example, useful when the energy of a signal at a specific frequency is needed. This chapter will launch a frequency domain investigation of modulation codes.

8.1) EXPERIMENTAL SPECTRAL ANALYSIS

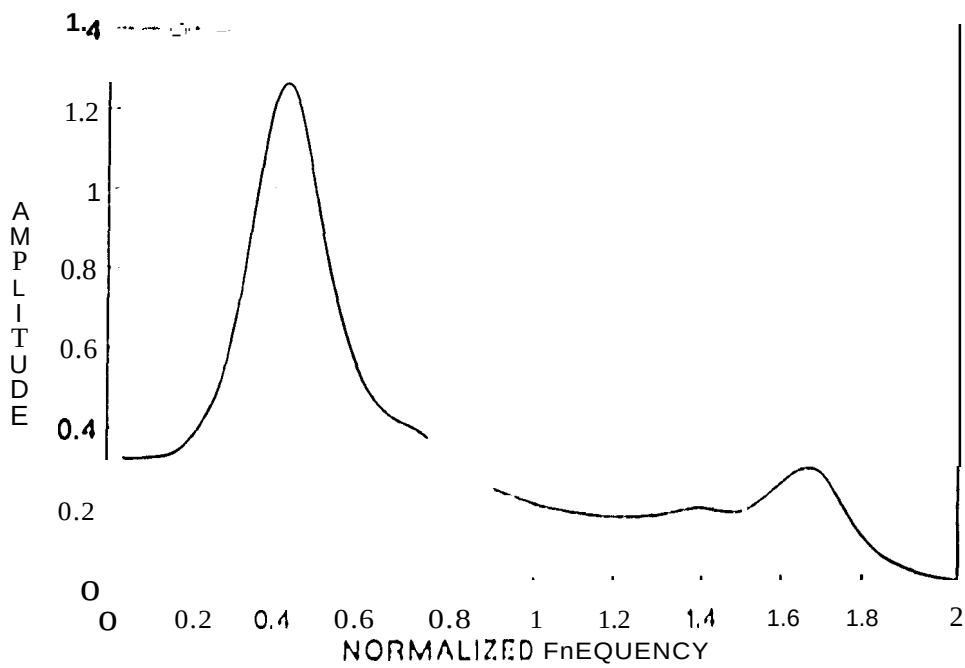
A spectrum analyzer presents a window to the frequency domain, and, since this study concentrate on experimental rather than theoretical results, this apparatus was used to gain information applicable to the frcquency domain.

SpCClnllll analyzers come in two basic varieties: swept-tuned and real-time. '1tC swept spectrum analyzer looks at only one frequency at a lime and generates a complete spectrum by sweeping in time. 'ntis can he n real disndvnntagc, since transient events cannot he measured. In addition, when scanning with narrow bandwidth, the sweep rate must be kept slow. Finally, only a small portion of the input signal is being used at anyone time,

These disadvantages of swept spectrum analyzers are remedied in the real-time spectrum analyzers, which was used in this study. '111is spectrum analyzer is based on digital Fourier analysis, in particular the famous Cooley-Tukey fast Fourier transfonn (FFT). Using a fast analog-to-digital converter. the analog input signal is converted to digital numbers where a special purpose computer implements the FFT, generating a digital frequency spectrum. Since this method looks at all frequencies simultaneously. it has excellent sensitivity and speed.

'The spectrum obtained is thus the Fourier spectrum of the signal and not the power spectral density (PSD). When consulting literature on modulation codes. spectra usually refers to the PSI) and not the Fourier spectra of a given code. llence, to have comparable results. the PSD must he derived from the measured Fourier spectrum by calculating the square of each measured point. Luckily a shortcut exists. By plotting the measured Fourier spectra on a logarithmic scale. the spectra will have the same shape as the PSD. with an offset. If desired. the average power can also be calculated directly from the logarithmic Fourier spectra.

Figures 8.1 and 8.2 depict the baseband spectra of the well-known rate R =1/2. *(d. k)* =0.3) Miller code on a linear and logarithmic scale. 111is spectra were compared to previously published spectra [3] and serves as an affinnation of the techniques and methods used to measure spectra of modulation codes.



AMPLITUDE IN VOLT

FIGURE ft.1

MEASURED SPECTRA OF RATE R =1/2. *(d, k)* = (1. 3) CODE ON LINEAR SCALE

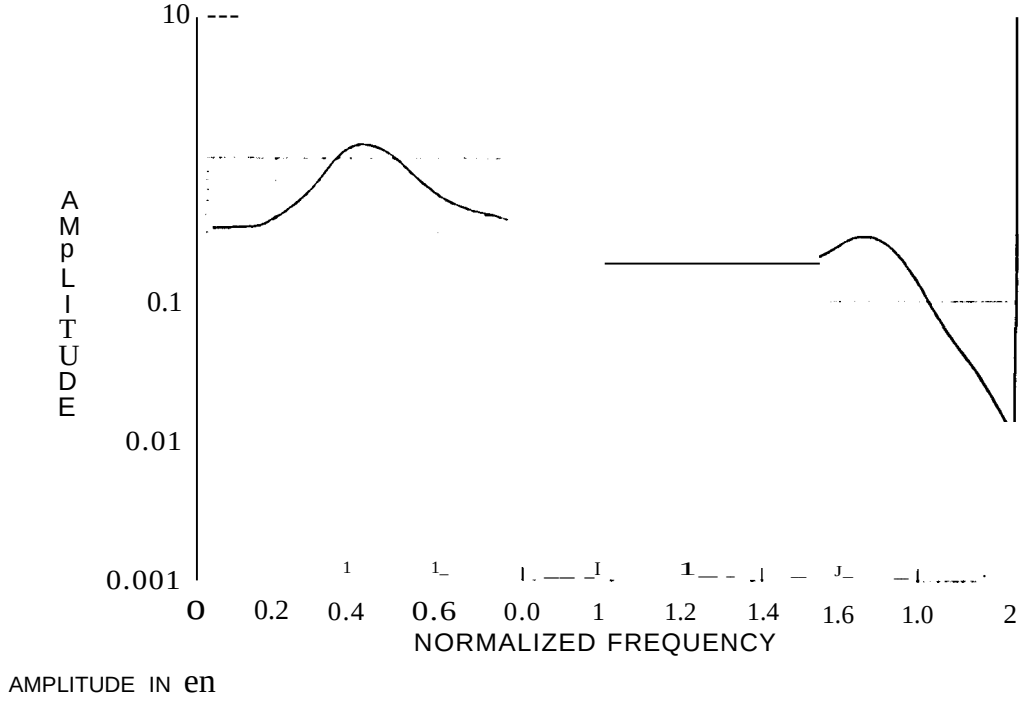


FIGURE 8.2

MEASURED SPECTRA OF RATE R = 1/2, (d, k) =(t. 3) CODE ON LOOARmlMIC SCALE

8.2) SPECTRA OF MODUl..ATION CODES

Table 8.1 summarizes the fourteen modulation codes investigated for usc on mobile communication channels. TIIC selected PN-scqucnce was a random sequence of 2^{9-1} = SI1 bits, obtained from a liP 492511 bit error rate meter [30]; this specific PN-scquence was also used throughout as an information bit source for the codes generated. Further, the rate R = 8/9, (d, k) =(0, 3) code investigated, was developed by Patel [29], while thc rate R = 11/12, (d, k) = (0, 3) code was synthesized with the algorithm described in chapter 6. TIle DC-free codes, rate R = 1/2, (d, 1:, C) = (0, 1, 1); R = 1/2, (tl, k, C) =(1,4,3); R = 1/2, (d, k, C) = 0,5,3) and R = 1/2, (tl, k, C) = (0, 4, 1), arc respectively the more than well-known Manchester [31], IICF 132], Miller[2] [3], and IIcdcman (33) codes. As mentioned in chapter 4, the rate R = 1/3, (d, k) = (3, 7) code was synthesized using the algorithms for sliding block codes, while the rate R = 1/2, (d, k) = (2, 7) is the well-known code uscd in 10M rigid disk drives 13]. Jacoby and Kost developed one of the several rate R = 2/3, td, k) = (1 , 7) codes [34]. This specific version was preferred over the others, because of the simplified decoder. 111C remaining four codes were synthesized hy van Renshurg and Ferreira [35].

The encoders and decoders for the rate R = 1/3, (d, k) = (3, 7) and rate R = 11/12, (d, k) = (0, 3) codes can be found, respectively, in chapter 4 and, as mentioned in chapter 6, on the floppy disk at the back of this thesis, while the other encoders and decoders are available in the enclosed references.

MODULATION CODE	RATE R
(d,k) = (0,3)	8/9
(d,k) = (0,3)	11/12
(d,k,C) = (0.1, J)	1/2
(d,k,C) = (0.2. J)	1/2
(d,k) = 0.2)	1/2
(d,k) = 0 .3)	1/2
(d,k) = 0,7)	2/3
(d,k,C) = 0.4.3)	1/2
(d,k,C) = 0.5.3)	1/2
(d,k) = (2.7)	1/2
(d,k) = (3.7)	1/3
(d,k) = (4.7)	1/4
(d,k) = (4.8)	1/4
(d,k) = (5.9)	1/4

TABLE 8.1
SUMMARY OF CODES INVESTIGATED

Figures 8.7 to 8.34 show the baseband spectra of all the codes summarized in table 8.1 and the PN-sequence on a linear and logarithmic scale; the frequency axis is normalized relative to the data rate. These results were obtained at 1200 baud, with the DSP FSM generator discussed in chapter 7. Programs written in TMS assembler, for the DSP FSM generator, realizing the encoders, and programs written in Turbo C, realizing the decoders, are presented in Appendix II.

8.3) MODULATION CODES THROUGH BANDLIMITED CHANNELS

Results were also obtained by means of bandpass experiments in the baseband, with an experimental set up as depicted in figure 7.5. The codes of table 8.1 were transmitted through highpass, lowpass and bandpass filters with different cut off frequencies and slopes.

The highpass filters simulated a channel with a poor low frequency and good high

frequency response, the lowpass filters simulated a channel with a good low frequency and poor high frequency response and the bandpass filters simulated a channel with poor low and high frequency responses.

First, second and fourth order filters were used. 111e first order filter was a run-of the-mill RC filter, while the second and fourth order filters were *switched capacitor filters*, available from National Semiconductors among others. The component used was a LMF 1000 switched capacitor filter.

A switched capacitor filter is an integrated circuit which contains a bunch of small capacitors that arc switched on and off to sample an input signal. By careful arranging the network of properly switched capacitors. one can favor certain frequencies and reject others. The big advantage of switched capacitor filters arc electronic tunability and minimum cost.

The switched capacitor filtcr sections used, contained second order lowpass and highpass filters in one integrated circuit. By cascading chips together. a higher order filter can be constructed; e.g. two chips cascaded result in a fourth order filter.

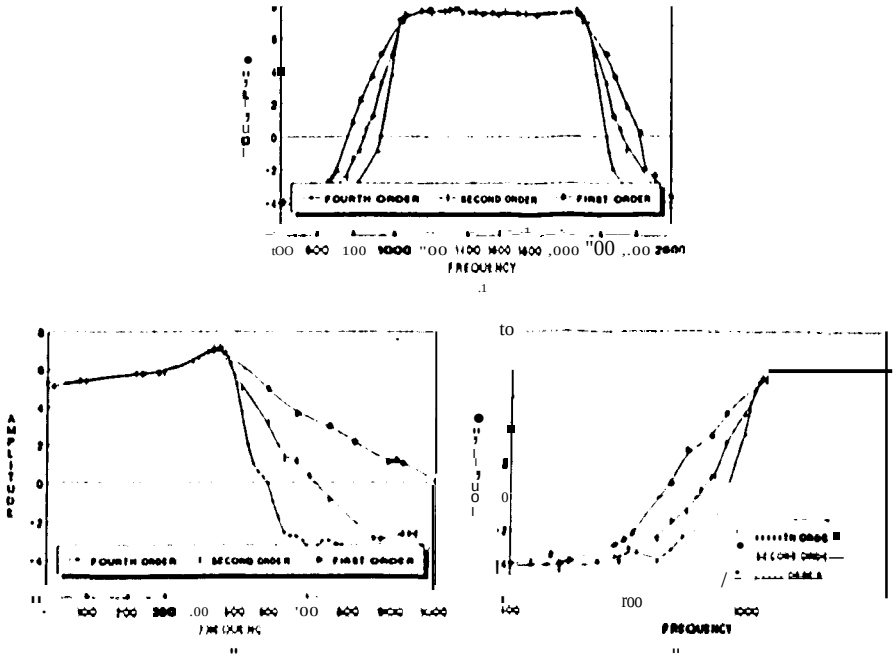


FIGURE 8.3

TRANSFER FUN<"'110NS 011 FILTERS USED

a)BANDPASS b)LoWrASS C)IIIGHPASS

Figure 8.3.a to 8.3.e show, respectively, the amplitude transfer function for the bandpass, lowpass and highpass filters used. 111e respective cut off frequencies for the lowpass and highpass filters in figures 8.3.l> and 8.3.c are 500 liz and 1 kllz, 11m bandwidth of the highpass filters are 1 kllz, between 1 kllz and 2 kllz. Filters with different phase shift responses were investigated. with no apparent effect on the results.

	111011 PASS			Low PASS			nAND PASS		
	1st	2nd	4th	1st	2nd	4th	1st	2nd	4th
PN-SEQ	0.19	0.14	0.10	0.62	0.65	0.70	0.72	0.76	0.80
(0,3)1	0.23	0.19	0.15	0.71	0.74	0.78	0.80	0.84	0.88
(0,3)2	0.21	0.16	0.13	0.67	0.71	0.75	0.n	0.82	0.86
(0,1,1)	0.70	0.64	0.58	1.32	1.36	1.41	1.31	1.36	1.42
(0,2,1)	0.50	0.45	0.39	1.11	1.15	1.20	1.31	1.35	1.41
0,2)	0.81	0.76	0.71	2.06	2.10	2.15	2.t3	2.17	2.22
0,3)	0.49	0.45	0.39	1.38	1.40	1.44	1.44	1.48	1.53
0,7)	0.31	0.26	0.20	1.06	1.10	1.15	1.13	1.17	1.23
0,4,3)	0.46	0.42	0.38	1.46	1.51	1.55	1.50	1.54	1.59
0,5,3)	0.45	0.42	0.37	1.50	1.56	1.60	1.53	1.57	1.62
(2,7)	0.40	0,35	0.31	1.17	1.20	1.25	1.39	1.43	1.47
(3,7)[3]	0.37	0,33	0.27	2.21	2.25	2.30	2.42	2.46	2.52
(4,7)[4]	0.46	0.41	0.38	2.91	2.96	3.10	3.24	3.28	3.36
(4,8)[4]	0.49	0.45	0.40	2.80	2.84	2.90	3.17	3.20	3.25
(5,9)4	0.48	0.44	0.39	2.91	2.96	3.00	3.22	3.26	3.30

1) Rale R = 8/9
2) Rate R = 11/12
3) Rate R • 1/3
4) Rate R • 1/4
Rest: R = 1{2

TABLE 8.2

BANDPASS EXPERIMENT RESULTS

Since the HP 4925B BER meter played such an important role in the experimental set up, Appendix II contains the circuit diagram for the clock recovery and bit synchronization circuits used by the HP 49250 bit error rate meter.

The modulation codes investigated had different code rates, starting as high as R = *11/12*, going down to R = 1/4. To be consistent the data rate for the investigated codes were constant; for a constant data rate and code rate of R = *mln* the bandwidth will increase *nlm* times over the PN-sequence bandwidth.

The emeries in table 8.2 arc presented with ascending *d* parameter and frequencies normalized relative to the *data* rates of the codes. These entries are unitless quantities, indicating the code's performance through bandlimited channels with channel properties as outlined above. A discussion to interprctthe entries in table 8.2 will follow.

Consider the rate R = 1/2, *(d, k)* = 0, 3) fourth order highpass, lowpass and bandpass entries in table 8.2, respectively 0.39, 1.44 and 1.53. If the data rate for the modulation code is 600 bits/s, the entry 0.39 can be intcrpreted as follows: a channel with a poor low frequency response, suppressing frequencies from 0 Hz to a -3 dB cutoff frequency of 600 x 0.39 = 234 Hz will achieve a BER of 10-t for this modulation code; the entry 1.44 can be interpreted as follows: a channel with a poor high frequency response, suppressing frequencies from a -3 dB cutoff frequcncy of 600 x 1.44 = 864 Hz upward will achieve a OER of 10^{-1} for this modulation code and the cntry of 1.53 as: a channel with a poor high frequency and poor low frequency response need a -3 dB bandwidth of 600 x 1.53 = 918 Hz to achieve a BER of 10^{-1} for this modulation code,

Consider figure 8.4; the high pass filter results were obtained in this way. A lowpass filtcr with cutoff frequcncy *nlm* (at the first spectral null of thc modulation code under investigation) was positioned at the high frequency side of the modulation code spectra; starting at the origin (0 1170), thc highpass filter's cut off frequency was tuned higher until a HER of 10^{-1} was achieved, A definite threshold was observed: the bit crror rate increased suddenly from zero to 10^{-1}. Since the *k* parameter determine the amount of energy at the low frequcncy side of the code spectra, this experiment quantified the influence of thc *k* parameter when transmitting the code sequences through a bandpass channel with poor low frequency response.

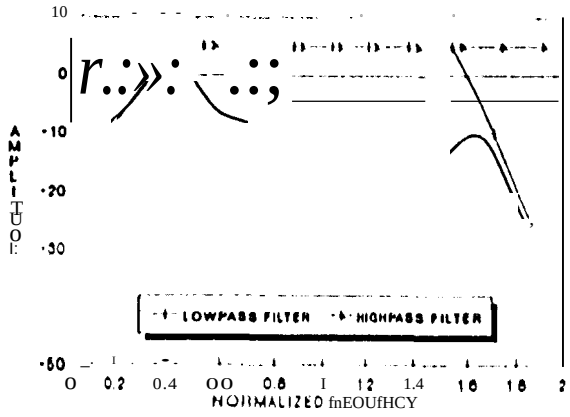


FIGURE 8.4

FILTER SET UP FOR HIGHPASS RESULTS

The lowpass filter experiments were conducted as seen in figure 8.5. Starting at the first spectral null *(nlm),* the cutoff frequency of the low pass filter was lowered until a BER of 10^{-} was achieved. These results give a quantitative indication of the *d* parameter, since the *d* parameter limits the high frequency components of the modulation code spectra.

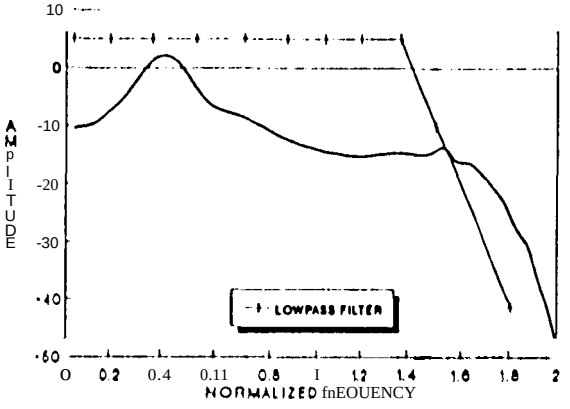


FIGURE 8.5

FILTER SET UP FOR LOWPASS FILTER EXPERIMENT

The bandpass results were obtained (figure 8.6) by a low pass filter at the first spectral null and highpass filter at the origin. By sequentially lowering the lowpass filter's cut off frequency until the threshold was observed, and raising the highpass filter's cut off frequency until the threshold was observed, a point was reached where the BER was 10^{-1}. This experiment gives an indication of the simultaneous influence of the *d* and *k* parameters through a bandlimited channel, since the low and high frequency components are simultaneously affected by the filters.

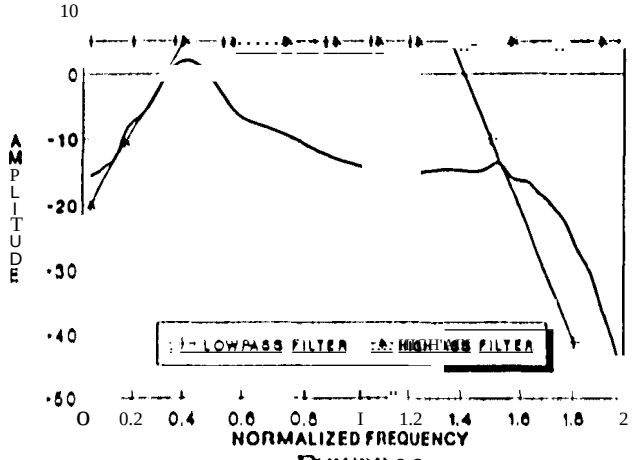


FIGURE 8.6

FILTER srrr UP FOR BANOPASS EXPERIMENTS

In future. reference can he made to these quantitative comparative results when choosing a code for a bandlimitcd channel, Although this list is by no means exhaustive, a proper understanding of the influence of the *d* and *k* parameters through bandpass channels can be gained. For instance: for a large *k* parameter (more energy concentrated at low frequencies) a feeble response through a poor low frequency channel will be achieved. On the other hand a feeble response through a channel with a poor high frequency response will be achieved when the *d* parameter is increased. The DC-free property. the C parameter, will naturally only have an influence on the low frequency side of the spectrum, thus enabling a better response through a channel with poor low frequency properties.

8.4) MODULATION CODES FOR MODILE COMMUNICATION CHANNELS

As a direct result of the bandpass experiments, five codes were selected for usc on mobile radio channels. 111e codes chosen were: the rate R=1/2, (*d*, *k*, C) = (0, I, 1) code, since it is already used on mobile radio channels [5] and [471, the rate R = 1/2. (*d*, *k*) = 0, 3) code, since it was recommended by the CerR [36], the rate R =2/J, (*d*, *k*) =0, 7) code, to look at the influence of the larger detection window (chapter 4) on a mobile radio channel, the rate R = *1/2, (d, k)* = (2, 7) code, 10 consider the influepcc of a larger *d* parameter and, finally. the rete R =1/2. (*d*, *k*, C) =(0,2, J) code, since the bandpass experiments showed that this code, although a rate R =1/2 code, cnn achieve a data rate comparable to a PN-sequence for the same bandwidth constraint. TI1C performance of these five modulation codes and the selected PN-sequcnce on a mobile radio channel is presented in chapter 9.

One might venture to say that the rate R =11/12, (*d*, *k*) =(0, J) code would be preferred on a mobile communication channel, since clock extraction nnd a relative small bandwidth

expansion can be gained from this code. However, the bandpass experiments showed that the rate R = 1/2, (d, k, c) = (0,2, t) code also needed a small bandwidth, with, of course, clock extraction gained. When consulting figures 8.15 and 8.16 it is evident why this code needs such a small bandwidth; almost all the energy is concentrated in the first half (low frequency side) of the spectrum, thus enabling us to achieve a higher data rate through a bandlimited channel. The rate R = 11/12 code was thus rejected: the encoder and decoder complexity were also considerably reduced by using the rate R = 1/2 code. Also, the decoder error propagation for the R = 1/2 code will be notably better.

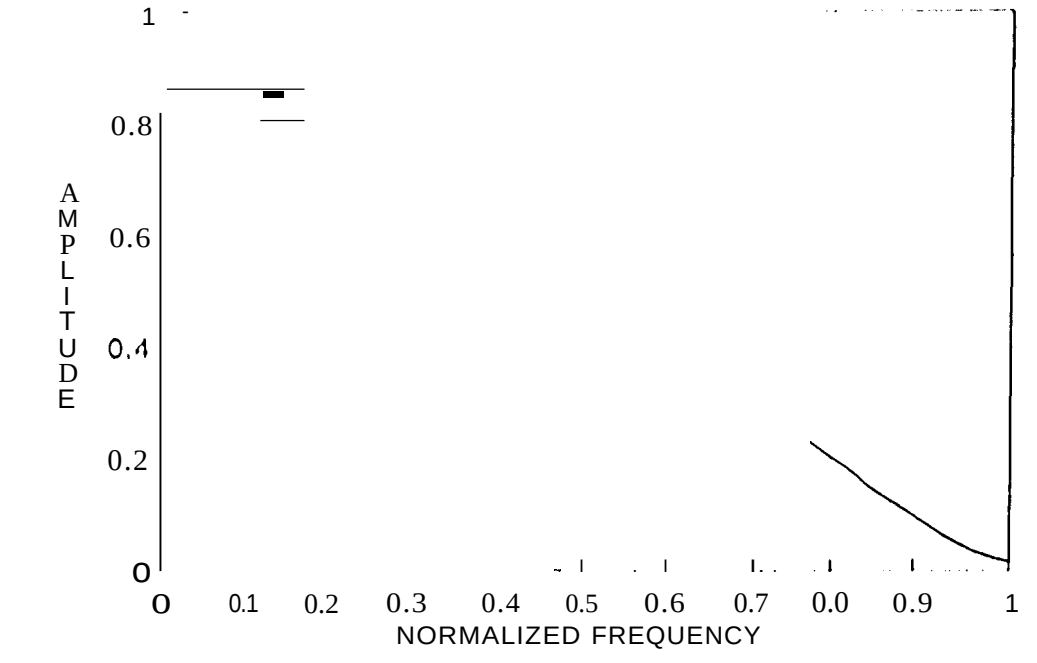
8.5) MODULATION SCHEME SPECTRA

Since indirect modulation was used to transmit the modulation codes over the mobile radio channel, four modulation schemes were considered; g-ary PSK, -t-ary PSK, differential PSK (DPSK) and fast-frequency-shift-keying (FFSK). The 8-ary PSK at 1600 baud, 4-ary PSK at 1200 baud and DPSK at 1200 baud were generated by a Rockwell DTY-500 programmable modem [7]. The FFSK modems employed the FX 419 and FX 429 chip set from Consumer Microcircuits Limited [8] and were generated at 1200 baud; a binary one is represented by one cycle of a 1200 Hz sinusoidal and a binary zero is represented by one and a half cycles of a 1800 Hz sinusoidal.

Figures 8.35 to 8.40 describe the FFSK spectra of the five chosen modulation codes and the selected PN-sequence on a logarithmic scale. Figures 8.41 to 8,43 describe a PN-sequence respectively as DPSK, 4-ary PSK and 8-ary PSK on a logarithmic scale.

THE spectra of the five selected codes after PSK modulation looked very similar to the PSK modulated PN-sequence; the PSK spectra do not have any sharp peaks in the spectrum (figures 8.37 to 8.39), thus the modulation code spectrum is spread out over the whole PSK spectrum, a 2500 Hz bandwidth for the g-ary PSK, which makes it very difficult to observe a difference in the various PSK spectra, A difference between the various modulation codes after FFSK can be observed because of the sharper peaks (more concentrated energy at a specific frequency] in the spectra.

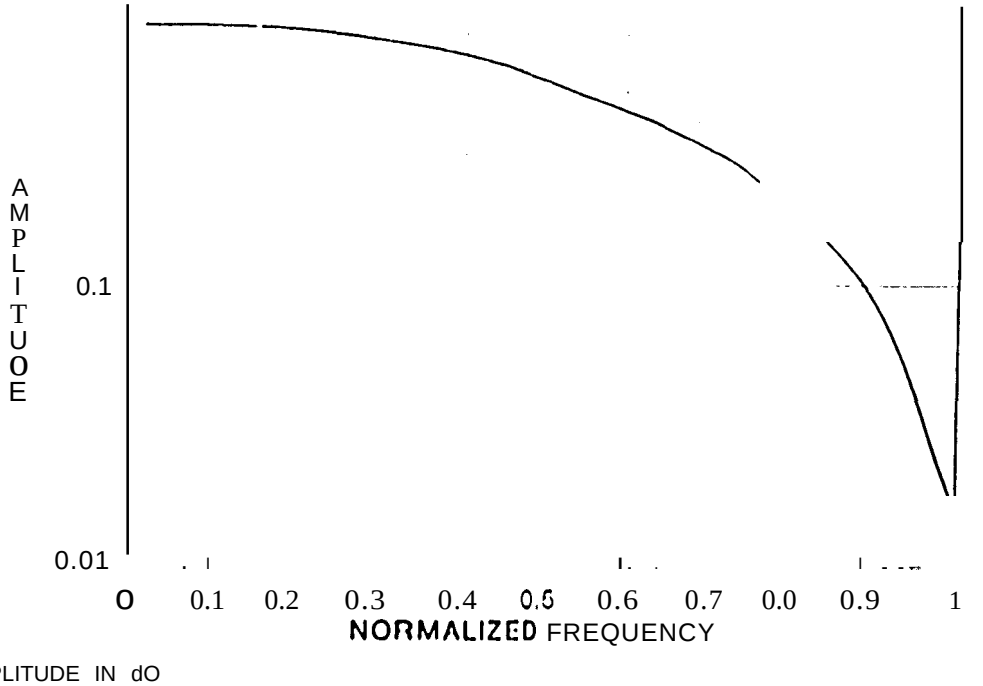
FFSK is known for its symmetric spectra; the sinusoidal transitions from a binary one to a binary zero (or vice versa) always change through the origin (mean value of sinusoidal). The PSK spectra, however, can make sinusoidal phase transitions from a binary one to a binary zero (or vice versa) anywhere, thus resulting in an unsymetric spectra (figures 8.41 to 8.43).



AMPLITUDE IN VOLT

FIGURE 8.7

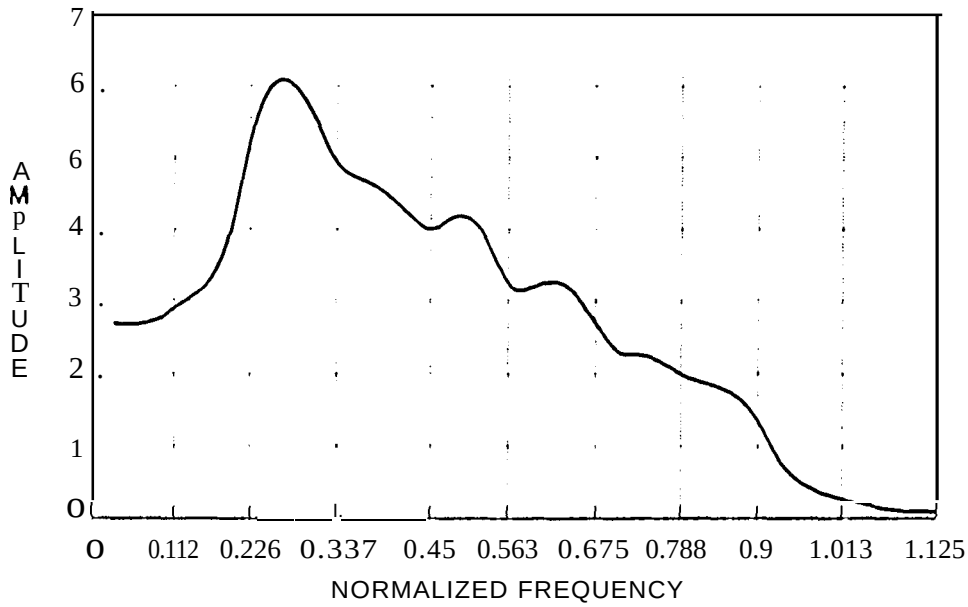
SPECTRUM OF THE SELECTED PN-SEQUENCE ON A LINEAR SCALE



AMPLITUDE IN dO

FIGURE 8.8

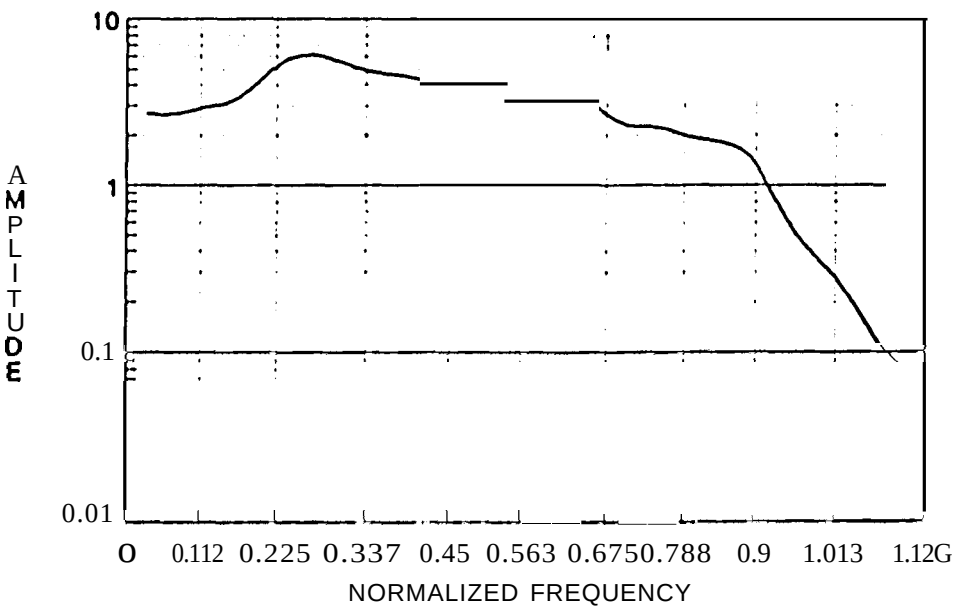
SPECTRUM OF 11m SELECTED PN-SEQUENCE ON A LoOARmIMIC SCALE



AMPLITUDE IN VOLT
R' 8/9

FIGURE 8.9

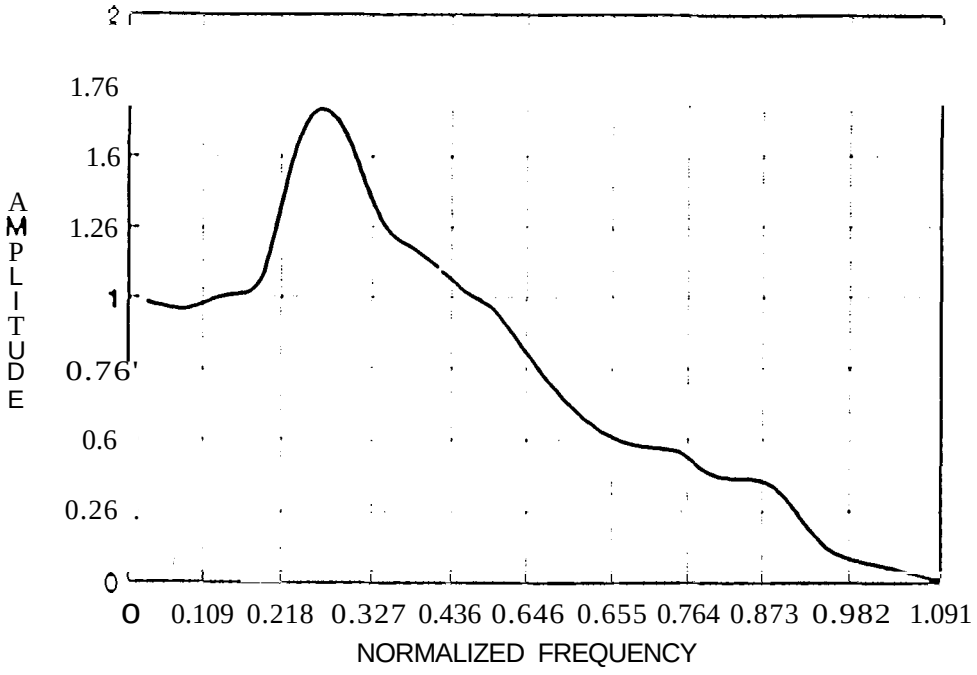
SPECTRUM OF A RATE R =8/9. (d, k) =(0. 3) CODE ON A LINEAR SCALE



AMPLITUDE IN dB
R • 8/9

FIGURE 8.10

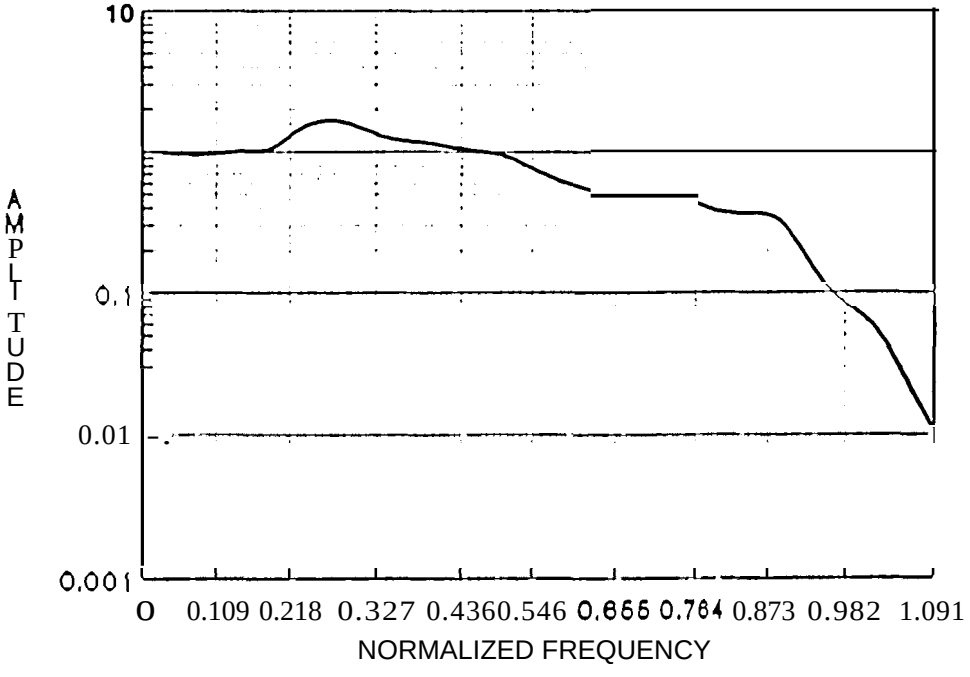
SPECTRUM 01' A RATE R =8/9, (d, k) =(0.3) CODE ON A 1.00ARI111MIC SCALE



AMPLITUDE IN VOLT

FIGURE 8.11

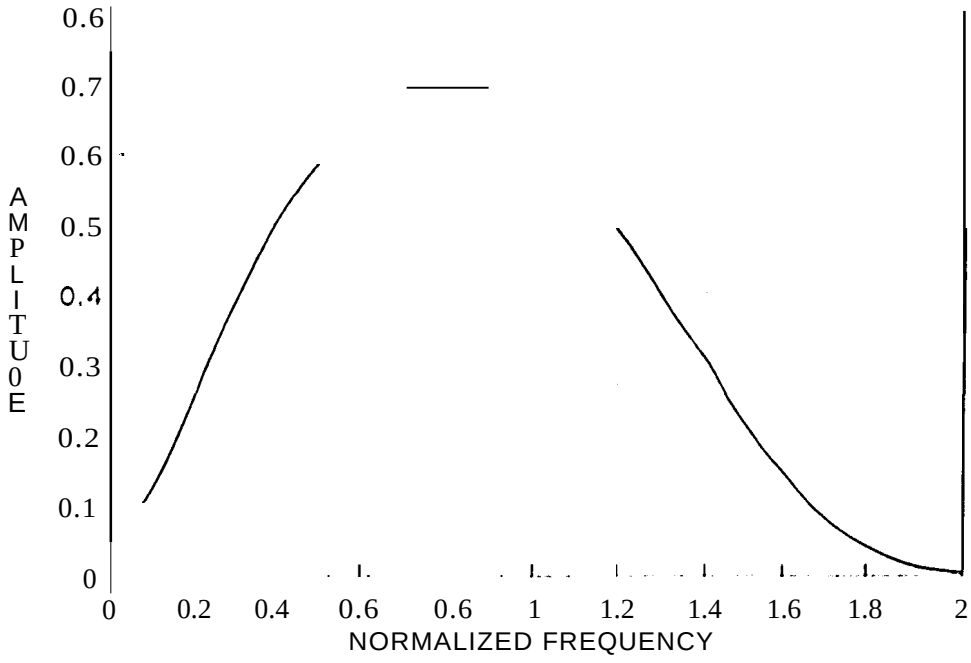
SPECTRUM OF A RATE R = 11/12, (d, k) = (0.3) CODE ON A LINEAR SCALE



AMPLITUDE IN dB

FIGURE 8.12

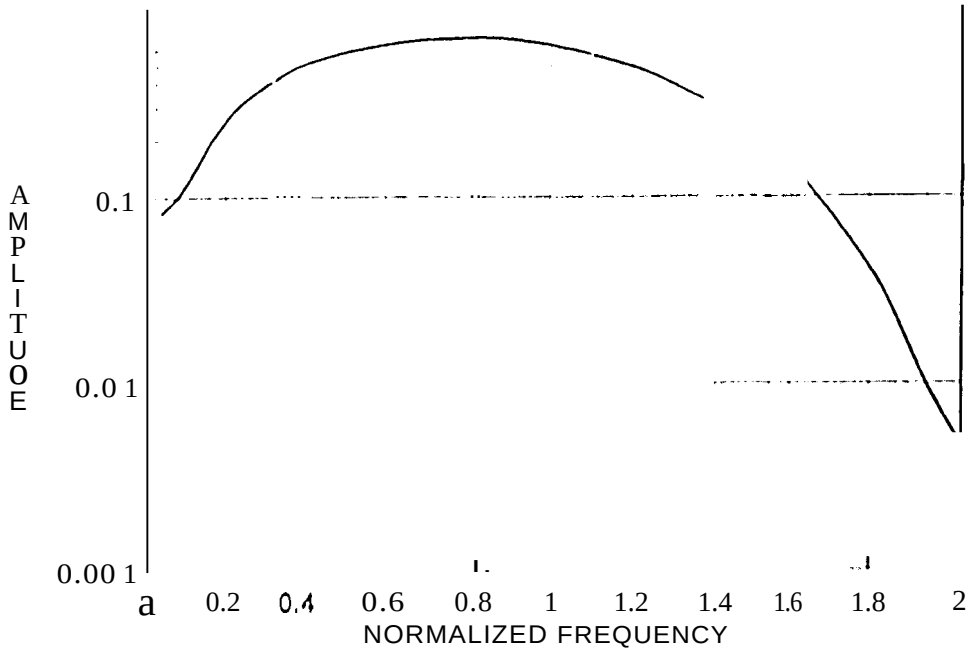
SPECTRUM OF A RATE R =11/12. (d, k) =(0, 3) CODE ON A LOGARITHMIC SCALE



AMPLITUDE IN VOLT

FIGURE 8.13

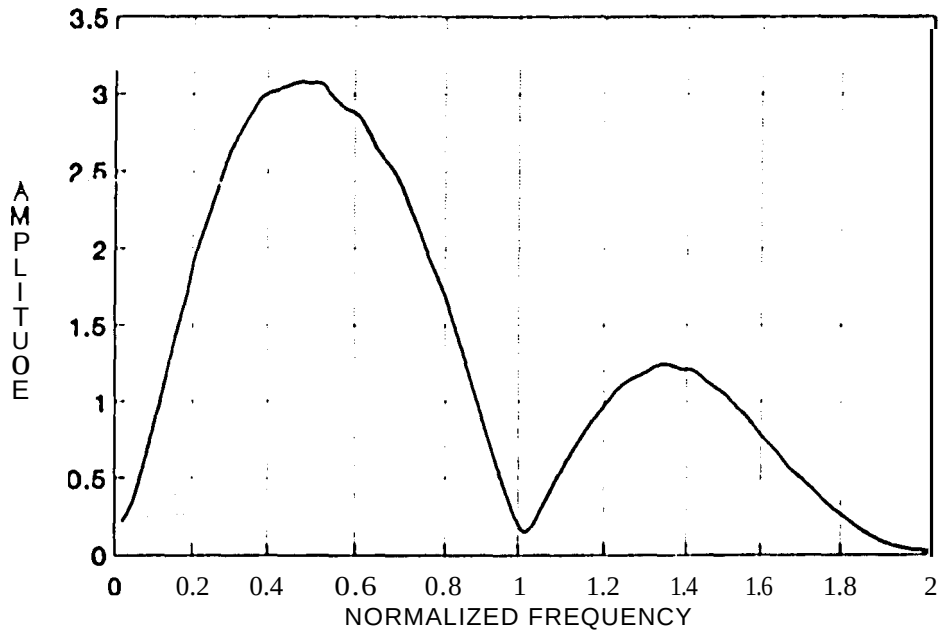
SPECTRUM OP A RATE R =1/2. (d, k, C) =(0. 1. 1) CODE ON A LINEAR SCALE



AMPLITUDE IN en

FIGURE 8.14

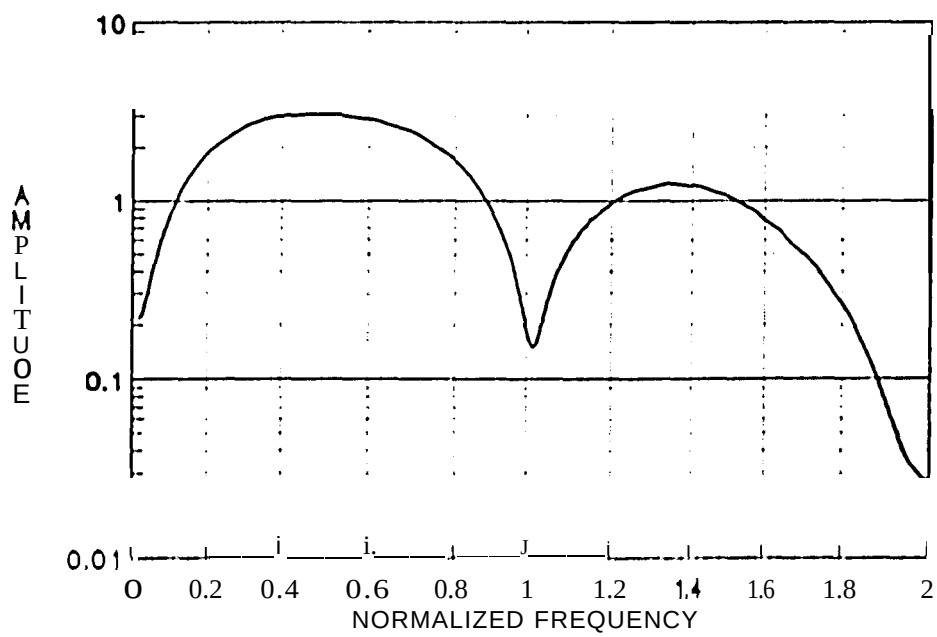
SPECTRUM OF A RATE R =1/2. (d, k, C) = (0, 1, I) CODE ON A LOGARITHMIC SCALE



AMPLITUDE IN VOLT

FIGURE 8.15

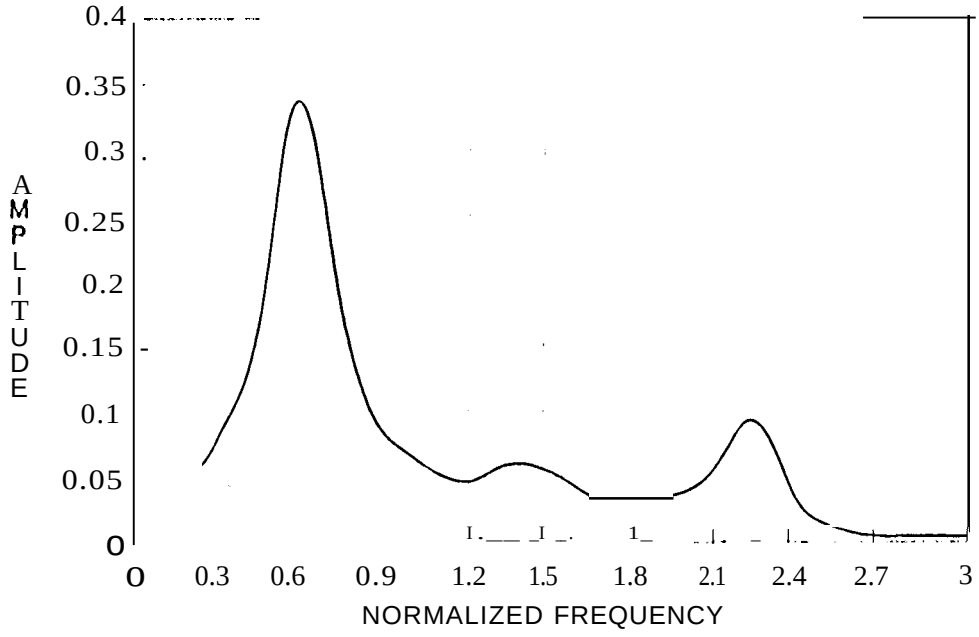
SPECTRUM OF A RATE R = 1/2, $(d, k, C) = (0,2, 1)$ CODE ON A LINEAR SCALE



AMPLITUDE IN dB

FIGURE 8.16

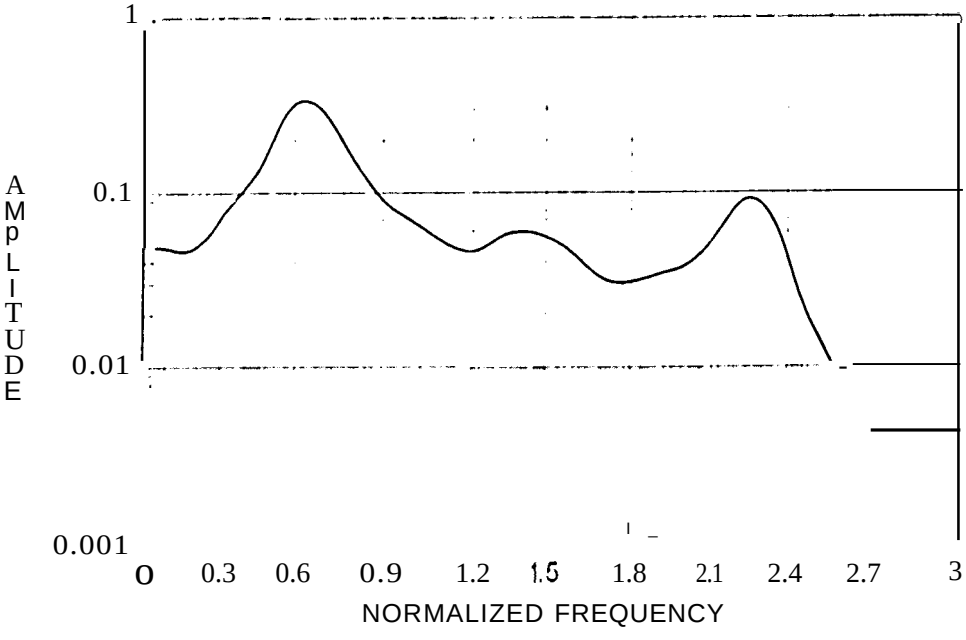
SPECTRUM OF A RATE R = 1/2, $(d, k, C) = (0, 2, J)$ CODE ON A LOOARmlMtC SCALE



AMPLITUDE IN VOLT
n • 1/3

FIGURE 8.17

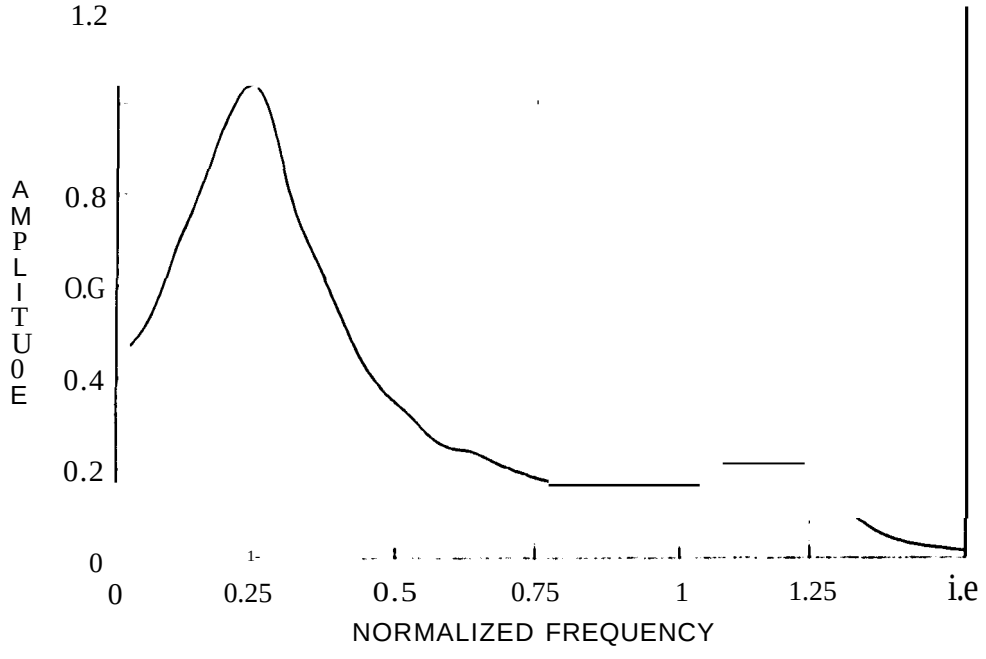
SPECTRUM OF A RATE R =1/2, (d, k) =0,2) CODE ON A LINEAR SCALE



AMPLITUDE IN dO
n · 1/3

FIGURE 8.18

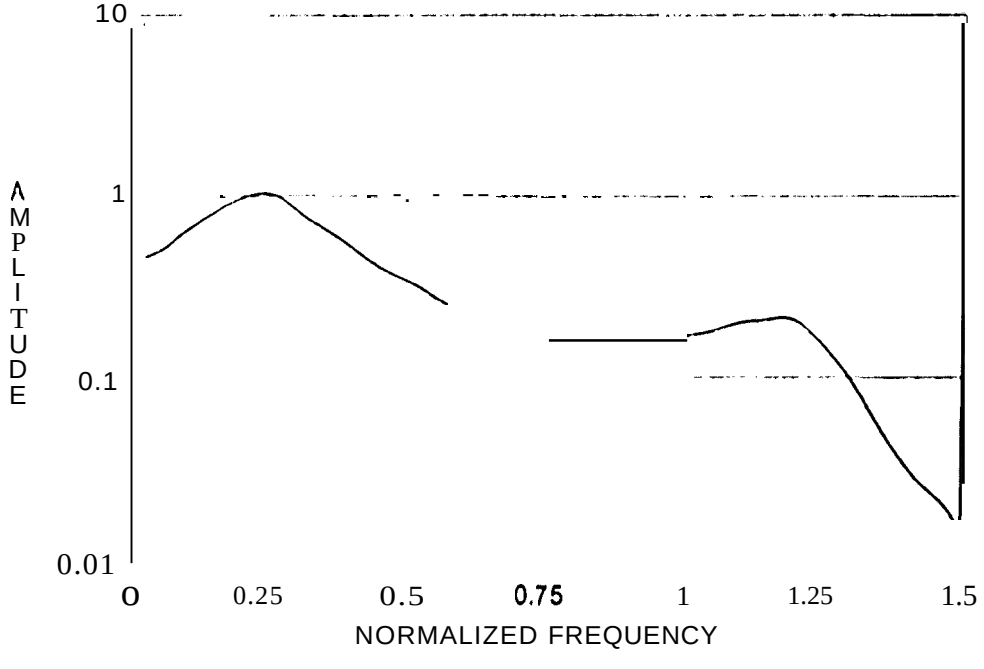
SPECTRUM OF A RATE R =1/2, (d, k) =0,2) CODE ON A LOOARmlMtC SCALE



AMPLITUDE IN VOLT

FIGURE 8.19

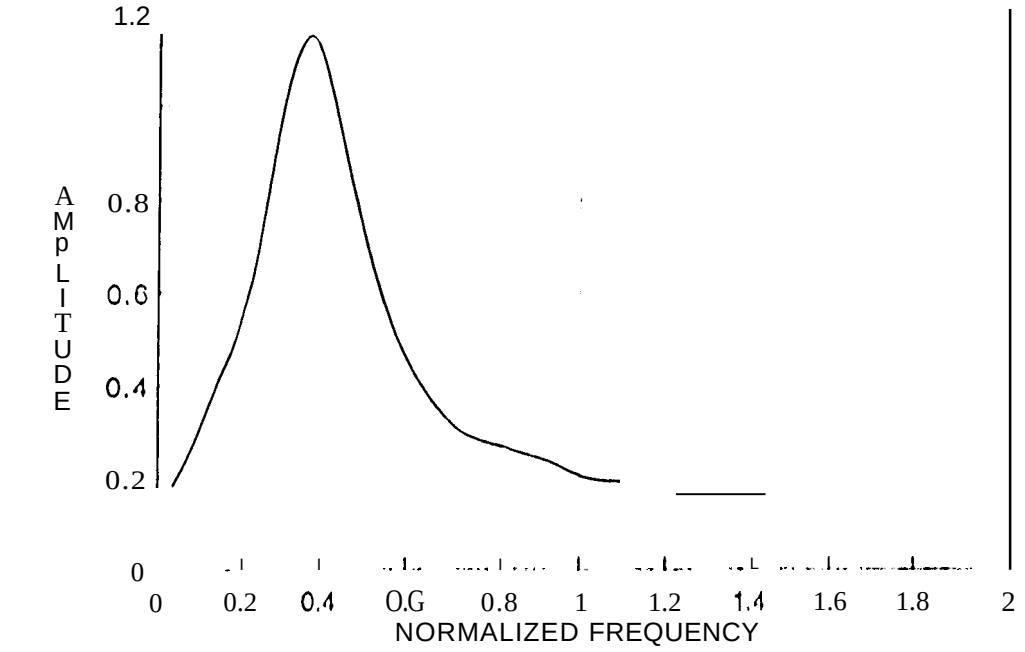
SPECTRUM OF A RATE R =2/3. (d, k) = 0. 7) CODE ON A LINEAR SCALE



AMPLITUDE IN en

FIGURE 8.20

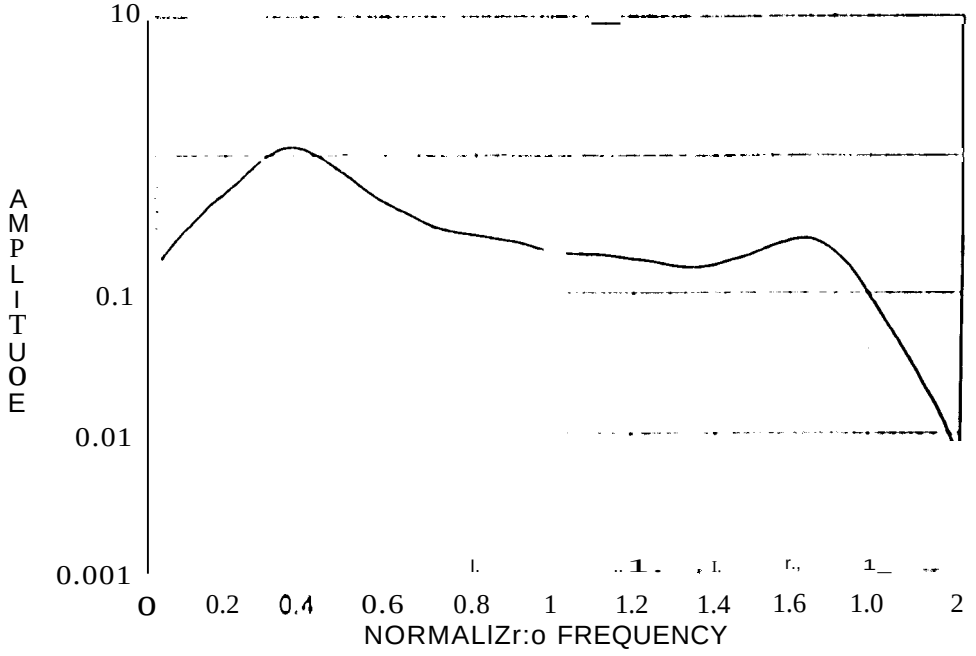
SPECTRUM OF A RATE R =2/3. (d, k) = 0, 7) CODE ON A tooARm'MIC SCALE



AMPLITUDE IN VOLT

FIGURE 8.21

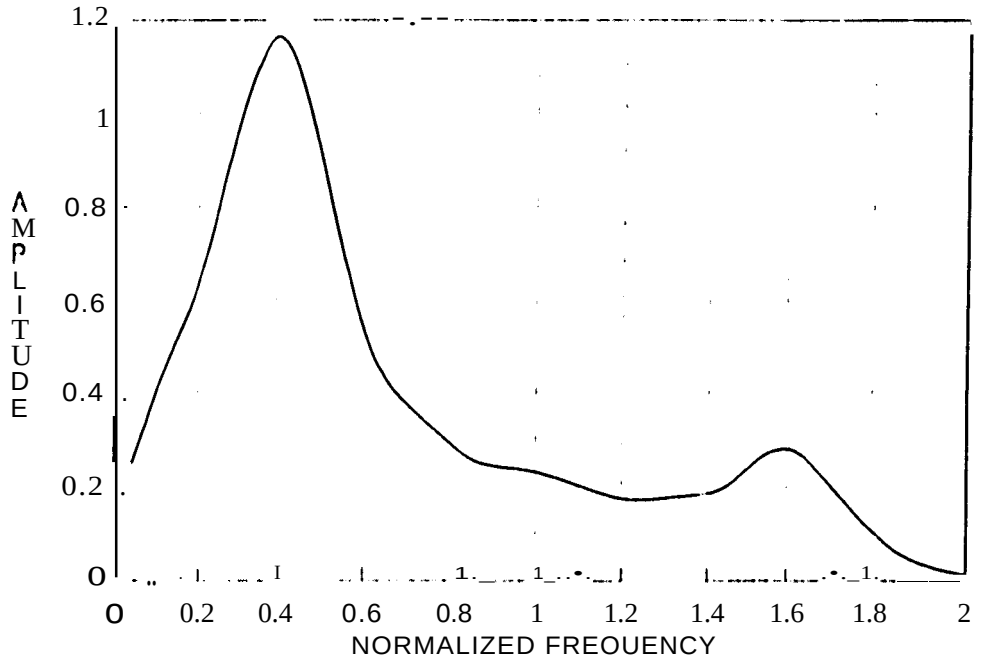
SPECTRUM OF A RATE R =1/2. (d, k, C) =(t. 4.3) COOS ON A LINEAR SCALE



AMPLITUDE IN dB

FIGURE 8.12

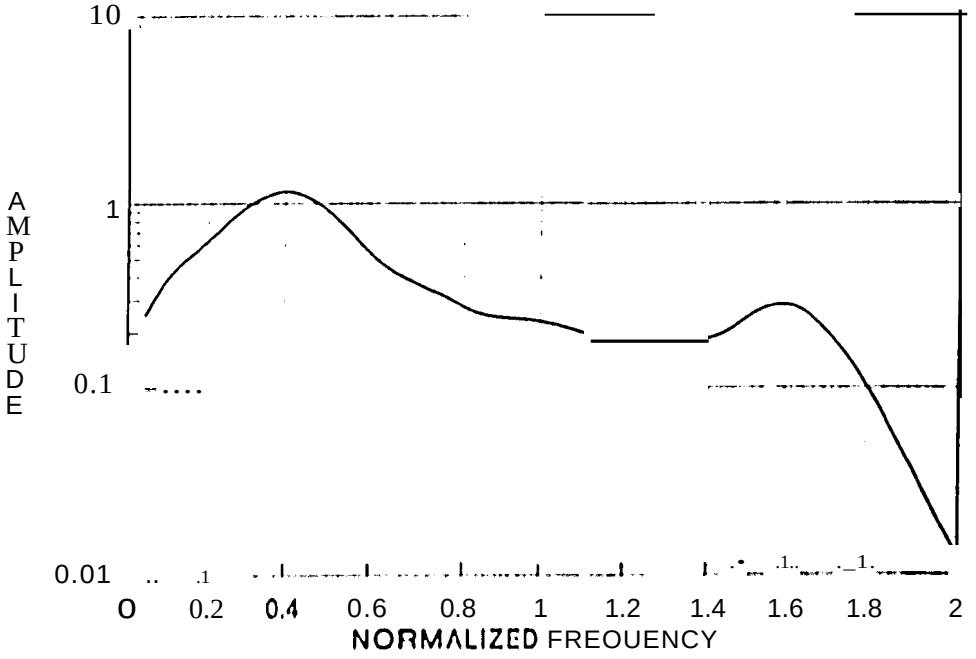
SPECTRUM OF A RATE R=1/2. (d, k, C) = (1,4, 3) CODE ON A LOOARmlMIC SCALE



AMPLITUDE IN VOLT

FIGURE 8.23

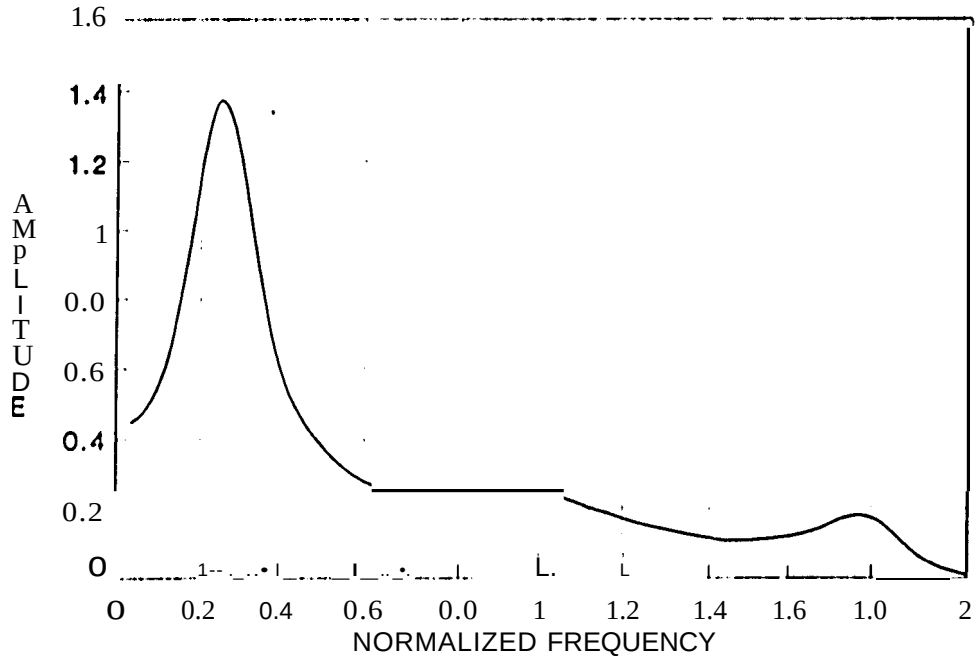
SPECTRUM OF A RATE R =1/2. (d, k, C) = U, 5. 3) CODE ON A LINEAR SCALE



AMPLITUDE IN dD

FIGURE 8.24

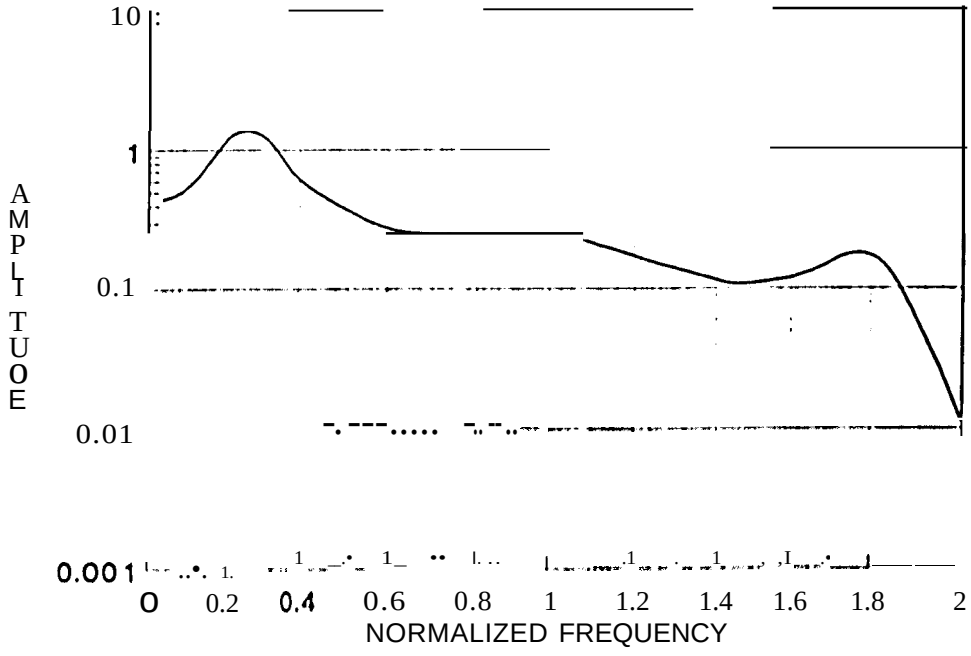
SPECTRUM OF A RATE R =112, U.« C) = (t, 5, 3) CODE ON A LOOARmrMIC SCALE



AMPLITUDE IN VOLT

FIGURE 8.25

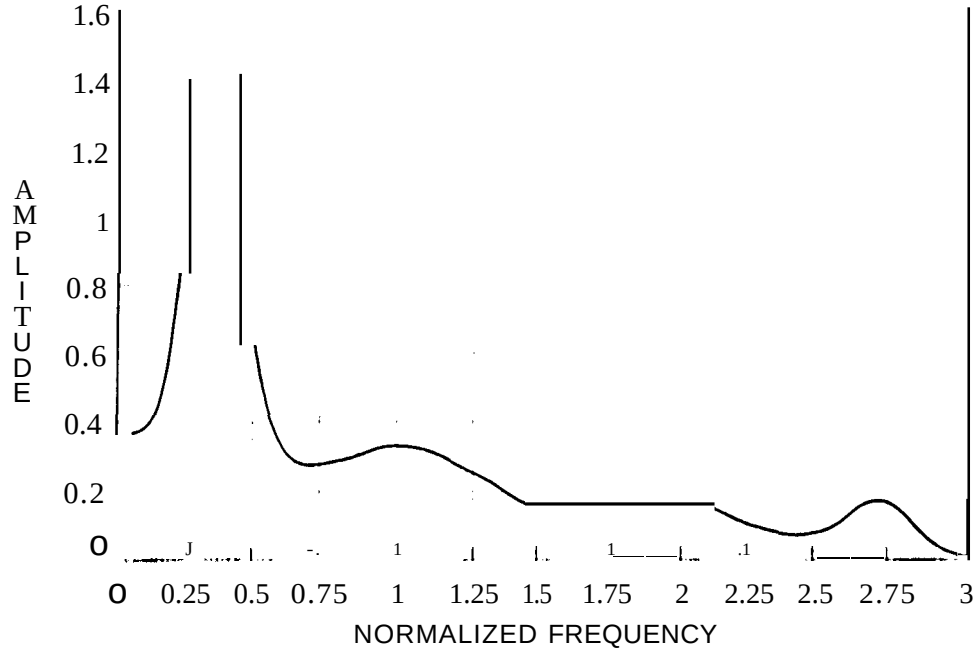
SPECTRUM OF A RATE R = 1/2. (d, k) = (2, 7) CODE ON A LINEAR SCALE



AMPLITUDE IN dD

FIGURE 8.26

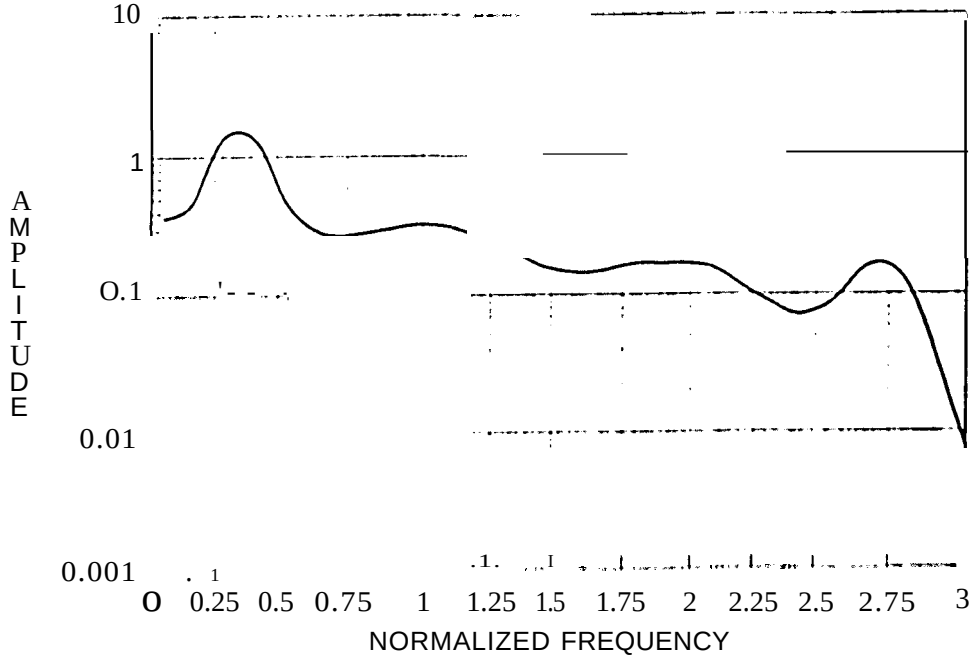
SPRCTRUM OF A RATE R = 1/2. (d, k) = (2. 7) CODE ON A LOOARmlMIC SCALE



AMPLITUDE IN VOLT

FIGURE 8.27

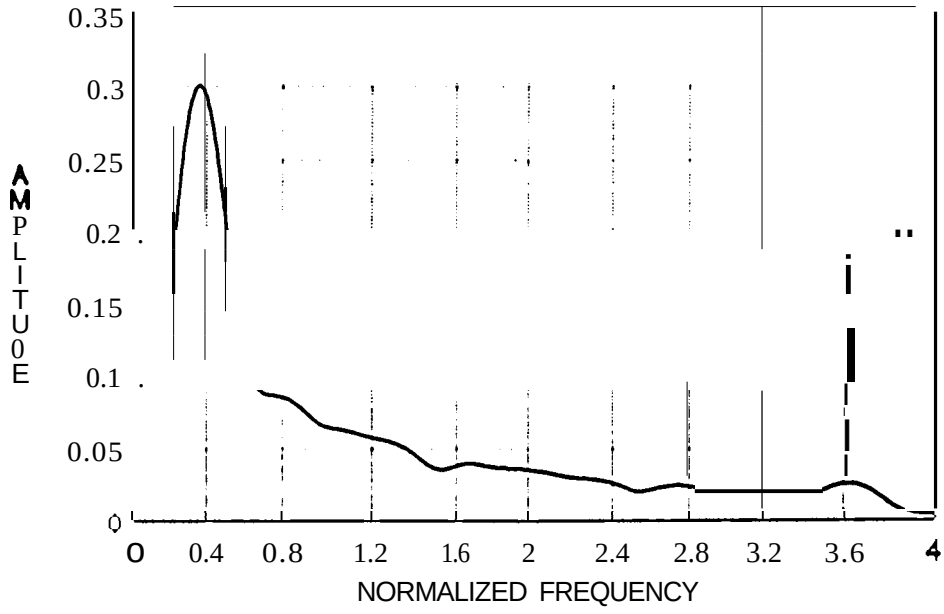
SPECTRUM OF A RATE R = 1/3. (d, k) = (3, 7) CODE ON A LINEAR SCALE



AMPLITUDE IN dD

FIGURE 8.28

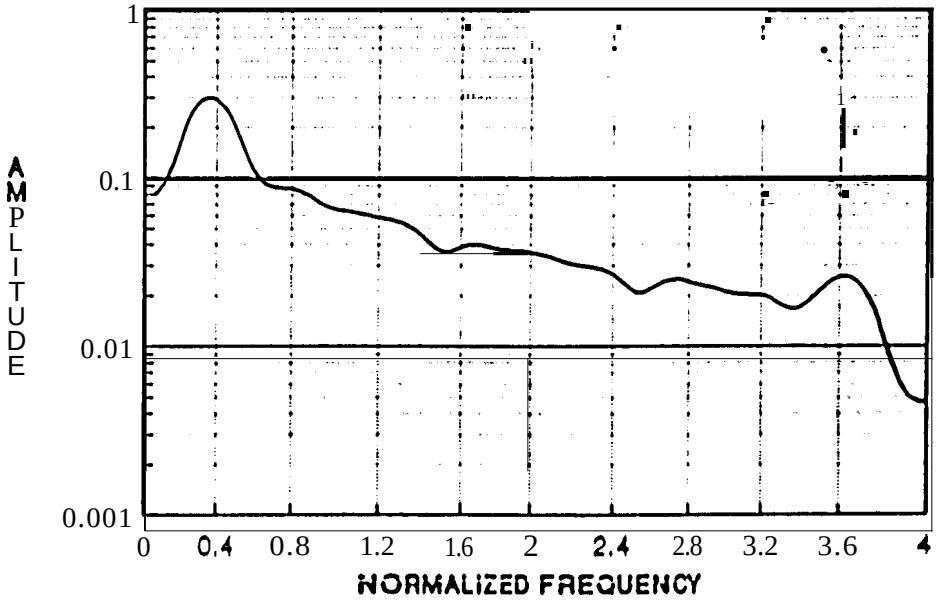
SPECTRUM OF A RATE R = 113. (d, k) = (3, 7) CODE ON A LOOARMIMIC SCALE



AMPLITUDE IN VOLT
R · 1/4

FIGURE 8.29

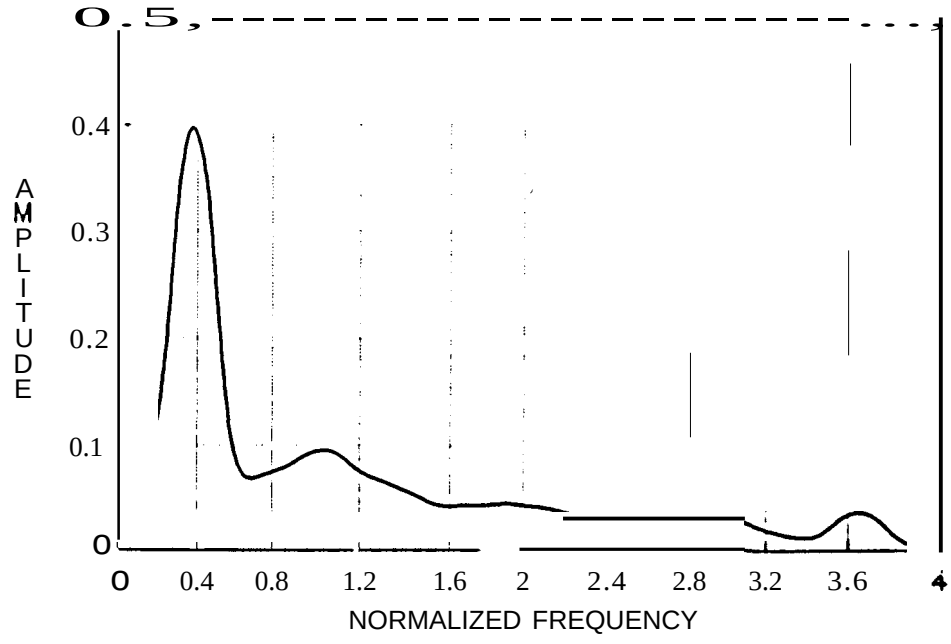
SPECTRUM OF A RAm R = 1/4. (d, k) = (4, 7) CODE ON A LINEAR SCALE



AMPLITUDE IN dB
R • 1/4

FIGURE 8.30

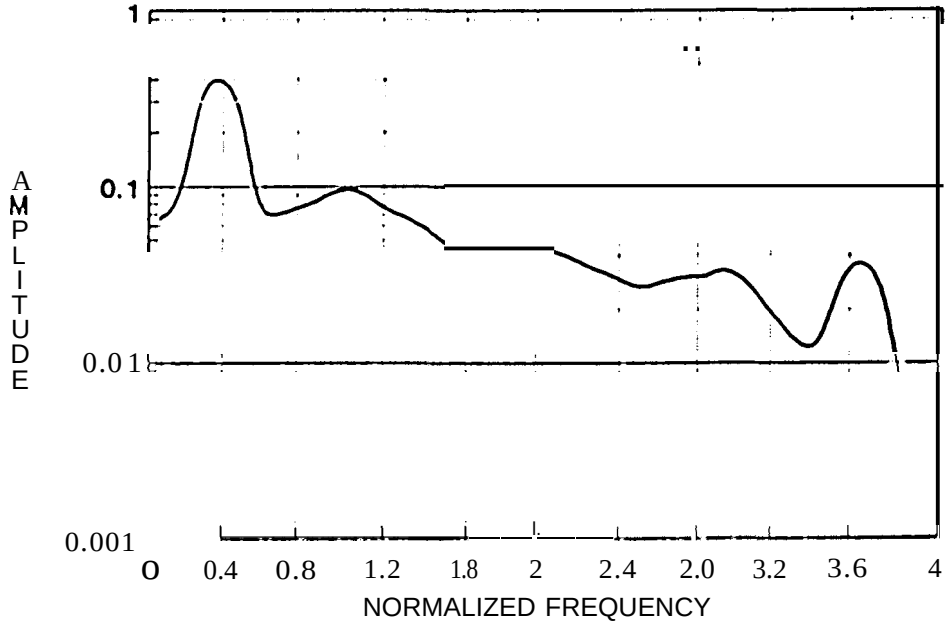
SPECTRUM OF A RAm R = 1/4, (d, k) = (4, 7) CODE ON A LOOARmlMIC SCALE



AMPLITUDE IN VOLT

FIGURE 8.31

SPECTRUM OF A RATE R =1/4, (d, k) =(4. 8) CODE ON A LINEAR SCALE



AMPLITUDE IN dD

FIGURE 8.32

SPECTRUM OF A RATE R =1/4, td, k) = (4. 8) CODE ON A LooARmlMIC SCALE

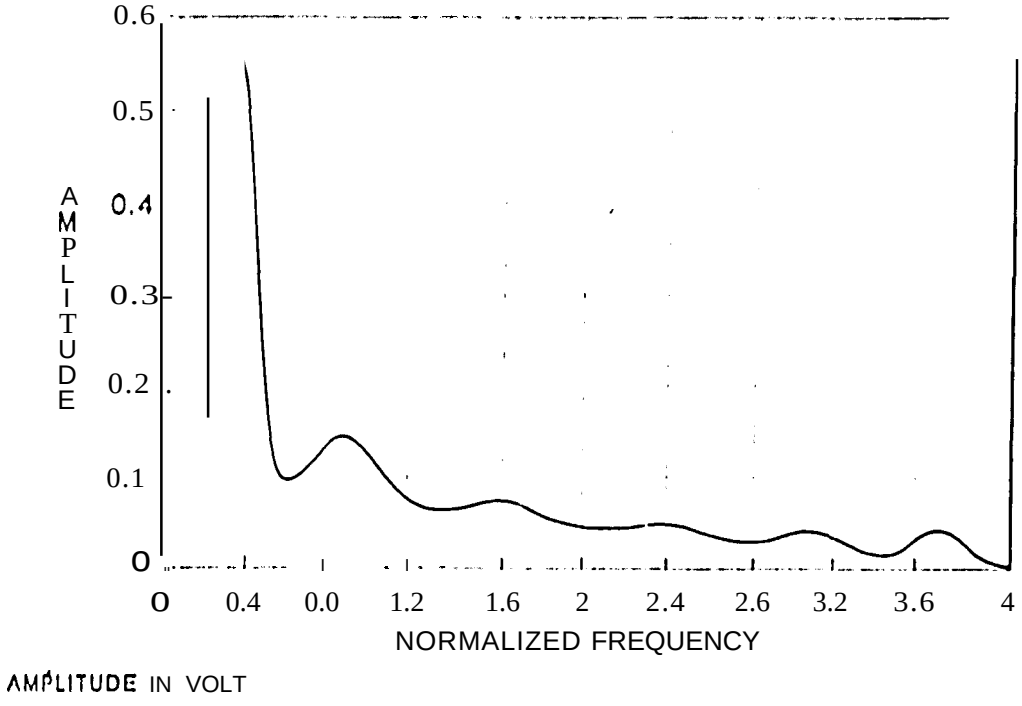


FIGURE 8.33

SPECTRUM OF ARAm R = 1/4, (d, k) =(5, 9) CODE ON ALINEAR SCALE

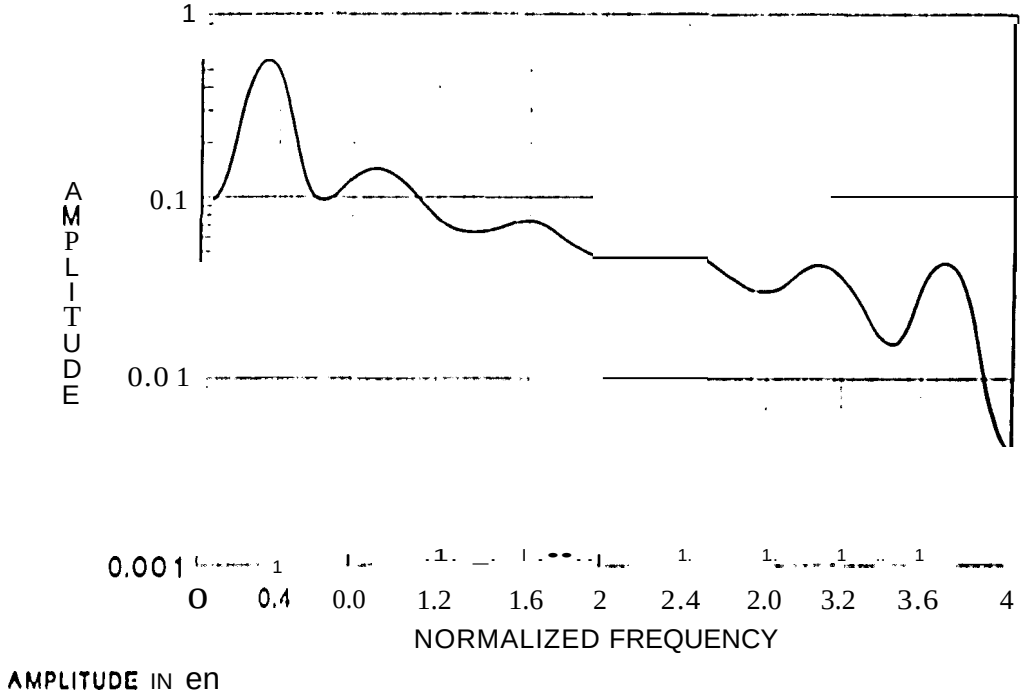
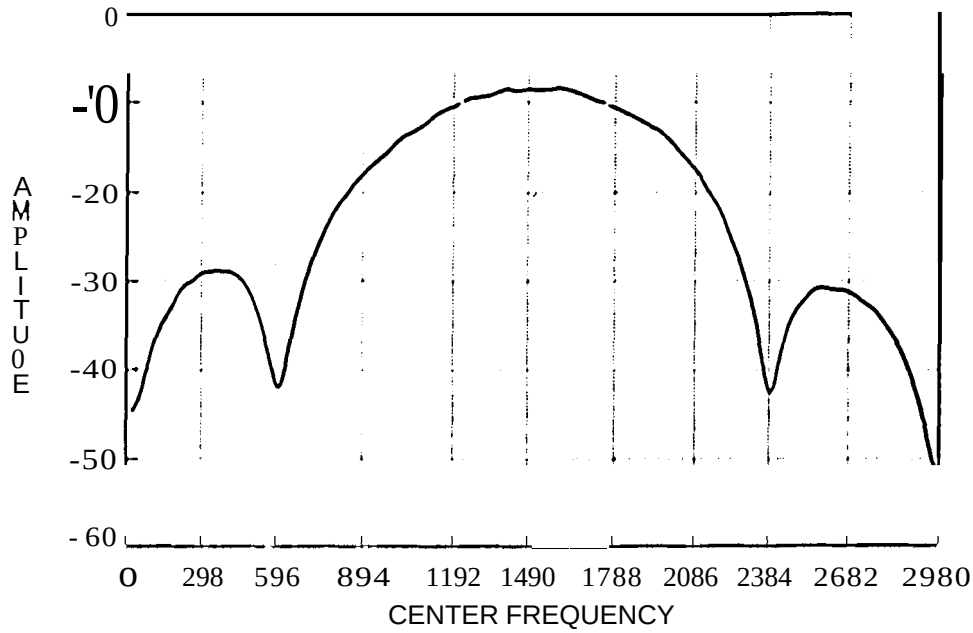


FIGURE 8.34

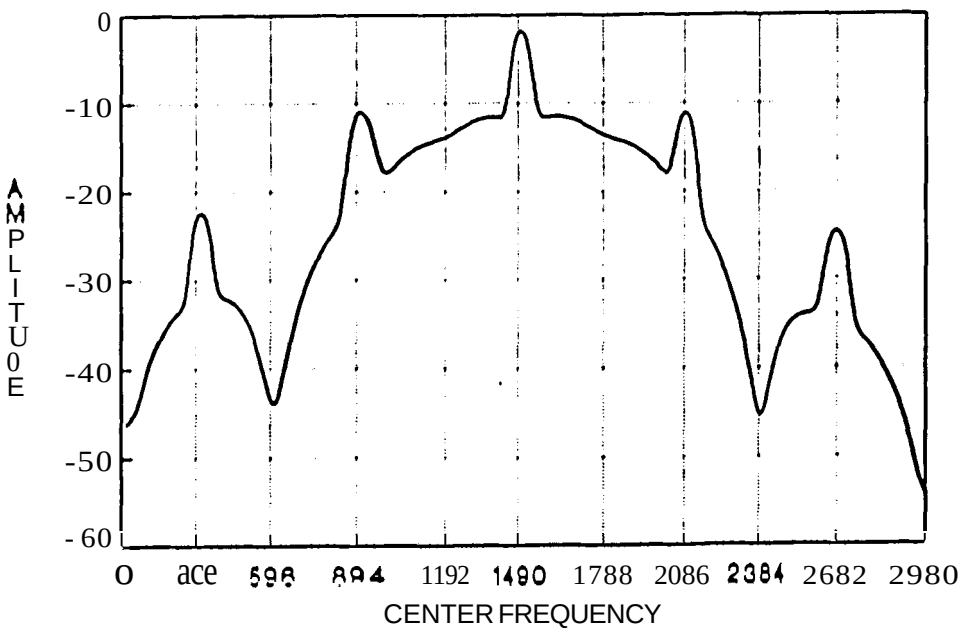
SPECTRUM OF A RATE R= 1/4, (d, k) =(5,9) CODE ON A LOOARmlMIC SCALE



AMPLITUDE IN dB

FIGURE 8.35

FREQUENCY-SHIFT-KEYED SPECTRUM OF THE sazcrai PN-SEQUENCE



AMPLITUDE IN dB

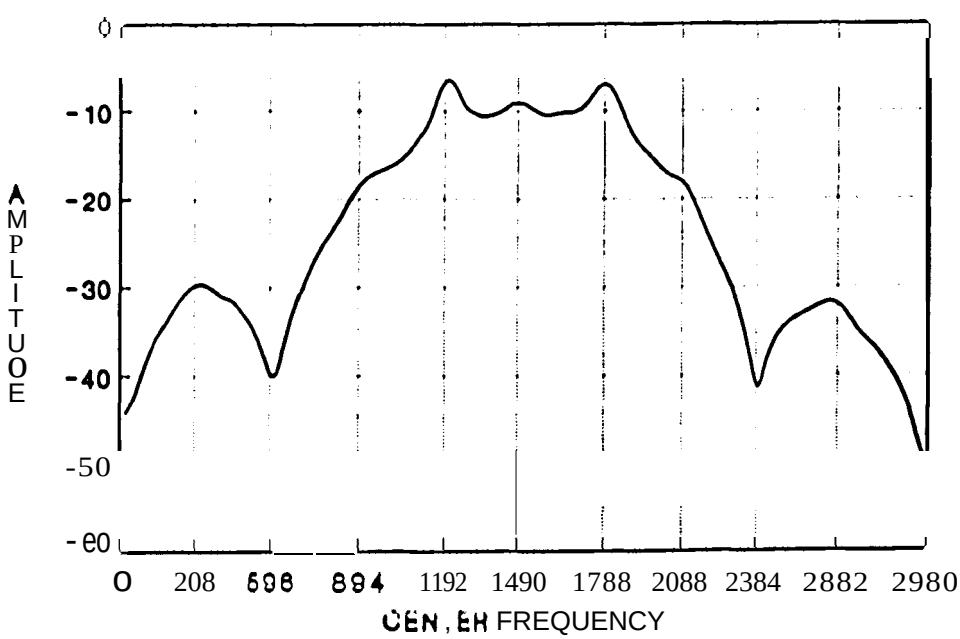
FIGURE 8.36

FREQUENCY-SHIFT-KEYED SPECTRUM OP A RATE R =]/2. (d, k, C) = (0. 1. 1) CODE

AMPLITUDE IN dB

FIOURE 8.37

FREQUENCY-SHIFT-KEYED SPECTRUM OF A RATE R =]/2, (d, k, C) = (0,2, 1) CODE



AMPLITUDE IN dB

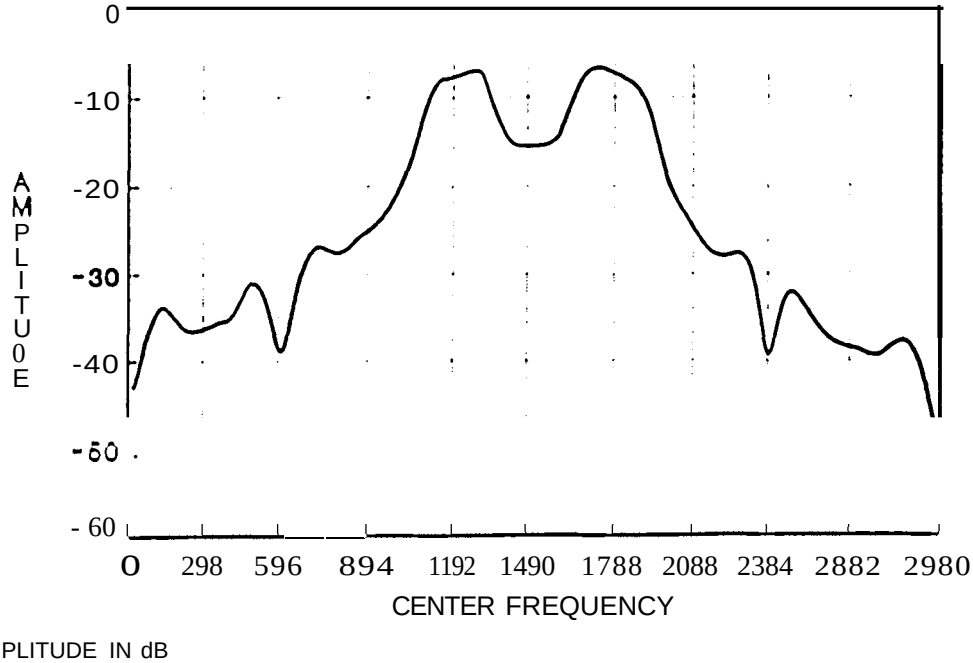
FIOURR8.38

FREQUENCY-SHIFT-KEYED SPECTRUM OF A RATE R =1/2, (d, k) =U, 3) CODE

AMPLITUDE IN dB

FIGURE 8.39

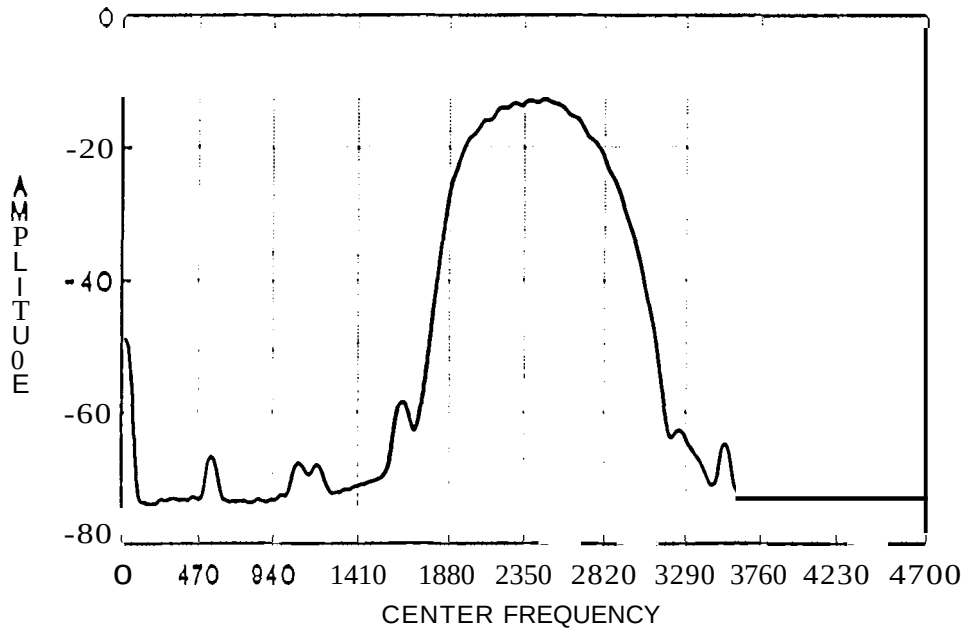
FREQUENCY-SHIFT-KEYED SPECTRUM OF A RATE R = 2/3. (d, k) = (1.7) CODE



AMPLITUDE IN dB

FIGURE 8.40

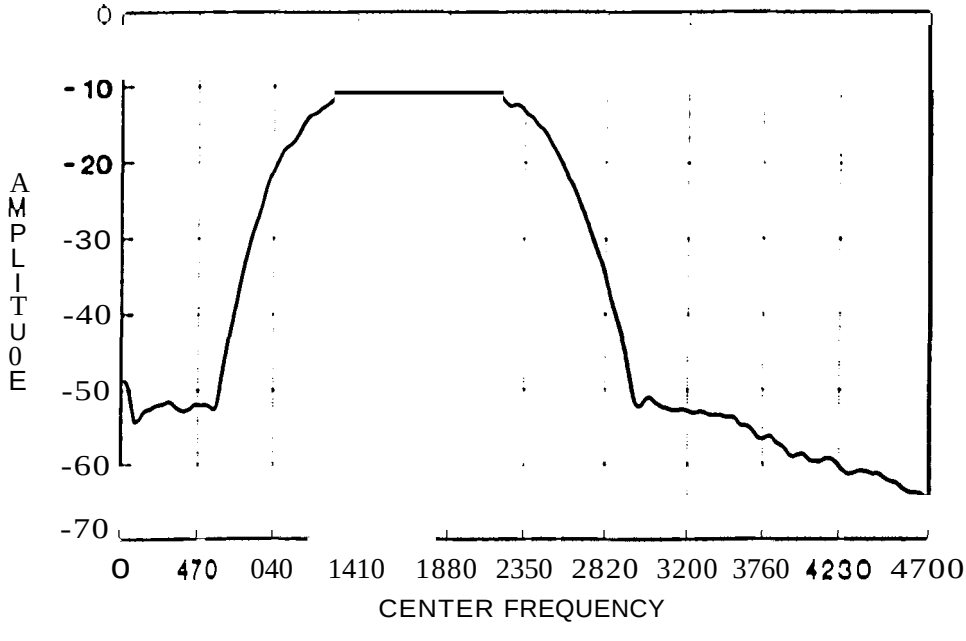
FREQUENCY-SHIFT-KEYED SPECTRUM OF A RATE R = 1/2. (d, k) = (2. 7) CODE



AMPLITUDE IN dB

FIOURE 8,41

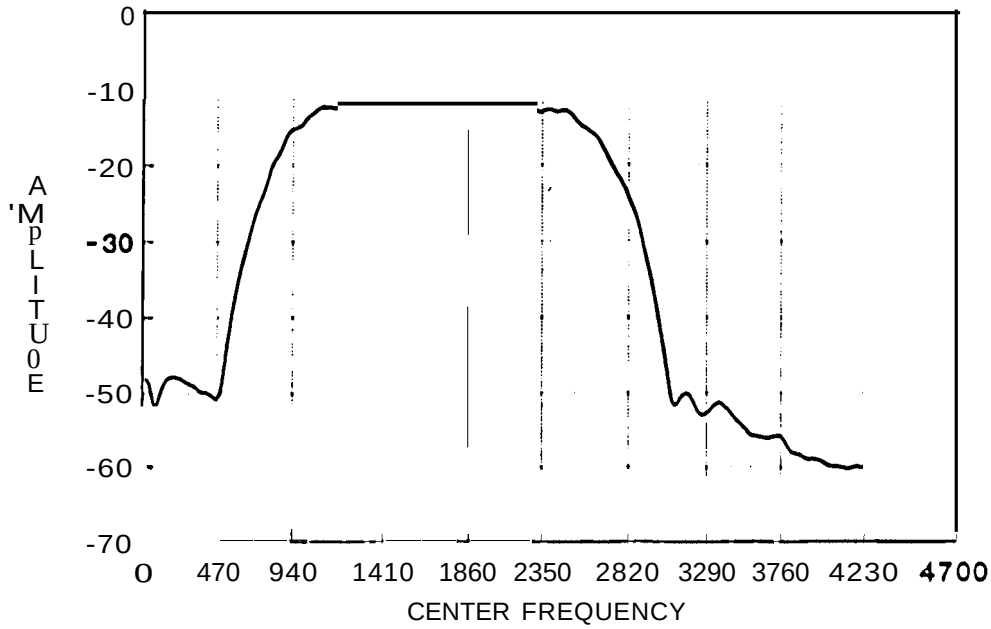
DIFFERENTIAL-PHASE-SHIFT-KEYED SPECTRUM OF THE SELECTED PN-SEQUENCE



AMPLITUDE IN dB

FIOURF. 8.42

4-ARY-PHASE-SHIFT-KEYED SPECTRUM OF THE SELECTED PN-SEQUENCE



AMPLITUDE IN dB

FIGURE 8.43

8-ARY-PHASE-SHIFT-KEYED SPECTRUM OF THE SELECTED PN-SEQUENCE

CHAPTER 9

MODULATION CODES ON DIGITAL MOBILE VHF CHANNELS: REAL TIME EXPERIMENTS

An informal definition of a communication system would be a system for the transmission of information from one point in space and time to another. With this definition in mind, it is not surprising that the communication engineer strives to transmit information as reliable and as fast as possible from one point to the other. Various techniques have been devised to achieve just this for digital communication systems; m·ary modulation coding ore used to promote a higher infonnation throughput for the same bandwidth constmint, error correcting codes are implemented for more reliable information transfer, different modulation schemes furnish different *signal-to-noise ratios* (SNR). 111is study, for instance, considered modulation codes on mobile communication channels to yield selfclocking and the list could go on indefinitely.

The abovementioned techniques, however, must first be tested and verified for reliability: for example, there is not much sense using an error correcting code which cause most of the dota errors itself, i.e, due to a catastrophic decoder] Hence, this chapter will describe the experimental results obtained for modulation codes on mobile VHF communication channels.

The experiments undertaken also resulted in n first-hand ecqualntance with the mobile communication channel and a verification of the theory presented in chapter 2.

9.1) Monn,s VI-IF EXPERIMENTS

Recalling from chapter 8, five modulation codes were chosen for experimental observations on mobile communication channels. Data bits, which were transformed to coded channels bits by the mobile encoder (chapter 7), were generated by a lIP 49258 DER meter, with a PN-sequence of length $2^9 - 1 = 511$ bits. Since indirect modulation was used, two sine wave modulation schemes were tested with the modulation codes; 8-ary PSK and 4-ary PSK. These modulation schemes were employed because high data rates can be achieved within a fixed bandwidth constraint. Two modems were used (transmit and receive) which were fully programmable between these two sine wave modulation schemes; by connecting the modem to the serial port of a digital computer, these modes can be selected at will [7]. Thc modulntion codes were tested under two realistic mobile communication conditions; a city environment and a more open freeway environment to quantify the influence of multipath propagation.

With figure 7.6 in mind; the liP 492Sn, modem, mobile encoder and VHF radio transmitter were installed in a vehicle, while the error-recording equipment [7], receiving end modem, mobile decoder, and VHF radio receiver were installed at a stationary base station. A summary of the mobile radio experiment parameters can be summarized as follows:

MOBILE UNIT

- Vehicle: Ford Sierra
- Radio: Kenwood TR 7500
- Carrier frequency: 145.200 Mhz
- Transmitter power: 5 W
- Modulation: FM
- Digital modulation: 4-ary or g-ary PSK, respectively at 1200 and 1600 baud.
- Antenna: 5/8 λ; vertical polarization
- Set up: The PN-scquencc from the liP 4925D was coded by the mobile encoder; the modem transformed the binary coded data to a phase-shift-keyed analog waveform, which was then transmitted by the VHF transmitter to the base station. This was repented for the five modulation codes and the uncoded PN-sequcnce with 4-ary and a-ary PSK in a city and highway environment.

BASE STATION

- Rand Afrikaans University, Auckland Park, Johannesburg
- Radio: Kenwood 75711 E
- Antenna: Folded dipole
- Set up: The transmit process was in effect reversed; the received VHF signal was demodulated and the PSK signal was transformed back to a digital signal by the modem. The decoder decoded the digital signal received from the modem, where the error-recanting equipment recorded the error sequence of the received signal. The maximum distance the vehicle travelled from the base station was approximately 25 kilometers.

Since clock extraction can be gained from all five modulation codes, their error propagation and other characteristics on a mobile channel had to be investigated in order to discriminate between them. The best, and perhaps the only way to do this, is to record the error distribution within the received bit stream; this was done by dedicated error-recording equipment and accompanying software developed by Swarts [7].

With this equipment it was possible to measure the bit error rate (BER), signal-to-noise ratio (SNR), error-free-run (EFR), cluster distribution, burst distribution, burst-interval distribution, gap distribution and the $P(\mu, v)$ distribution (the probability that a block of ν digits will contain exactly μ errors [31]). For the sake of completeness, figure 9.1 shows a typical error sequence.

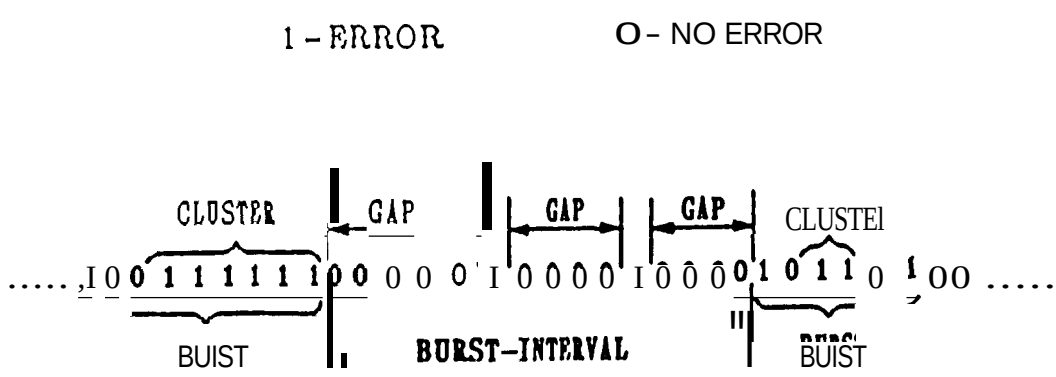


FIGURE 9.1

AN **ERROR SEQUENCE** WITH ERROR **EVENTS**

111e events can be described as follows:

9.1.1) GAP DISTRIBUTION

A gnp. as seen in figure 9.1, is a region of error-free bits between two errors. with the gap length the number of those error-free bits. The gap distribution is a plot of the cumulative relative frequency of the gap length versus the gap length.

9.1.2) BURST DISTRIBUTION

A burst is ıı region in which the ratio of the number of errors to the total number of bits in that region exceeds the burst density Δ_0 [38]. If the successive inclusion of the next error keeps the density above Δ_0, the burst is continued, but the burst ends if the inclusion of an error reduces the density below Δ_0. Further. a burst must start with an error and end with an error; the burst should not start with an error which belongs to the previous burst. For the results obtained, the threshold value for Δ_0 was 10^{-1}.

The burst distribution is a plot of the cumulative relative frequency of the burst length versus the length of the burst.

9.1.3) BURST-INTERVAL DISTRIBUTION

A burst-interval is the region between two bursts as shown in figure 9.1. The burst interval distribution is the plot of the cumulative relative frequency of the burst-intervals versus burst-interval length. It gives some indication of the dependence between bursts.

9.1.4) CLUSTER DISTRIDUTION

A cluster is a region of consecutive errors in an error sequence, The clustcr distribution is the plot of the cumulative frequency of the clusters versus the cluster length.

9.1.5) ERROR-FREE RUN DISTRIDUTION

Fritchman (39] and Tsai (38] defined the error-free run distribution, P(Oc/U. as the probability, given an error, c or more consecutive error-free bits will follow. From the definition it follows that:

$$P(0^{\varepsilon}/1) = 1 \qquad - \qquad (9.1)$$

Tsai [38] also showed that the error-free run distribution can be calculated directly from the gap distribution.

9.2) RESULTS

In this section the observations and results of modulation codes over VHF mobile channels will be discussed. Figures 9.2 and 9.3 show, respectively, the modem signal constellation for 4-ary PSK and 8-ary PSK with no noise present, while figures 9.4 and 9.5 show the same signal constellations when the mobile unit entered a multipath fading condition. The signal points, and thus the phases are scattered all over the constellation, making it difficult and ultimately impossible for the modem to correctly demodulate the phase information to the original transmitted bit stream.

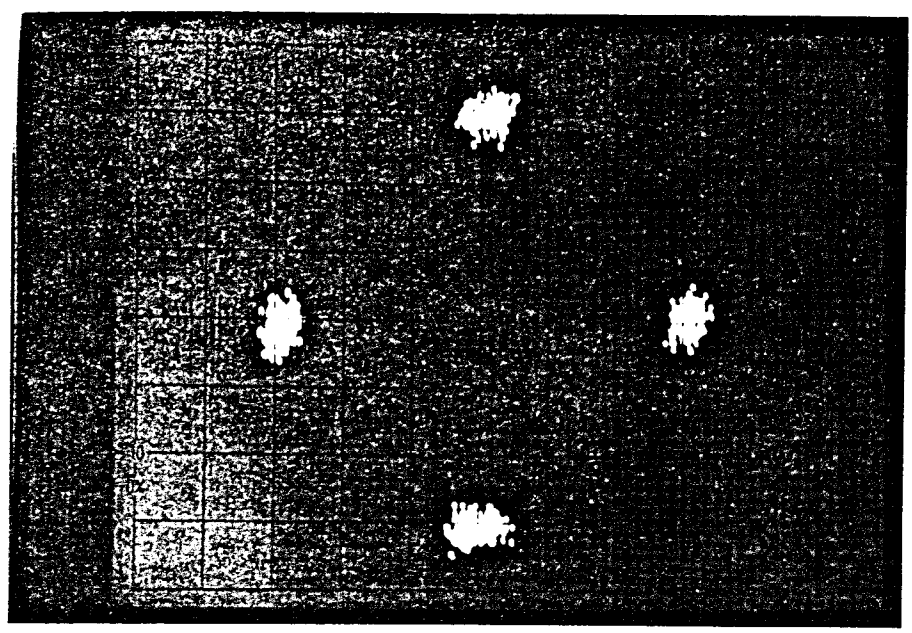


FIGURE 9.2
4-ARY PSK SIGNAL CONSTELLATION

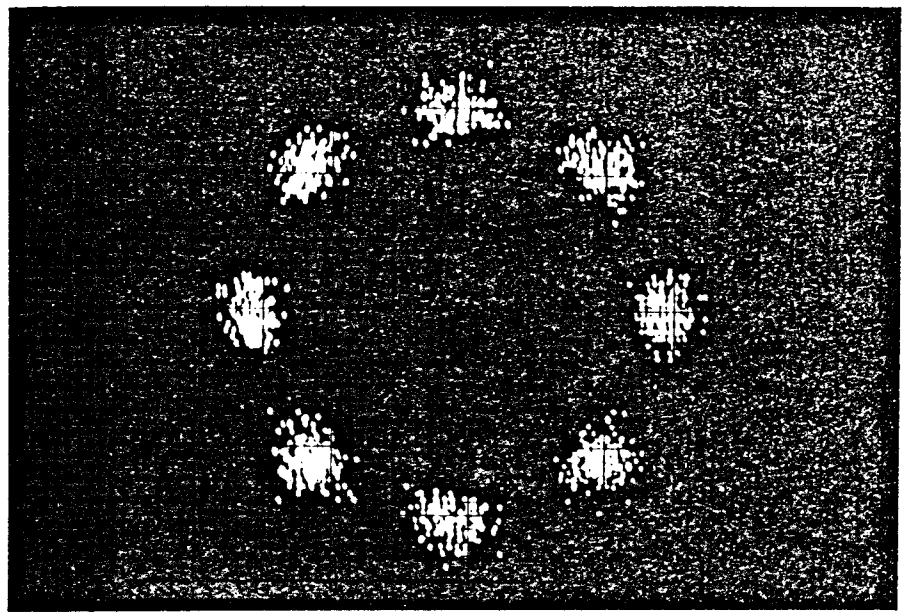


FIGURE 9.3
8-ARY PSK SIGNAL CONSTELLATION

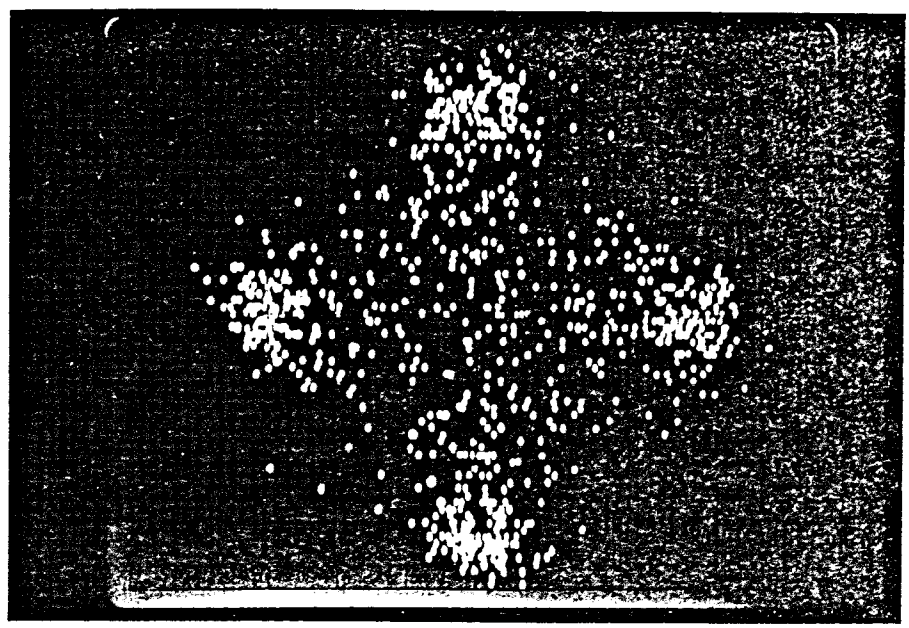


FIGURE 9.4
4-ARY SIGNAL CONSTELLATION IN FADING CONDITION

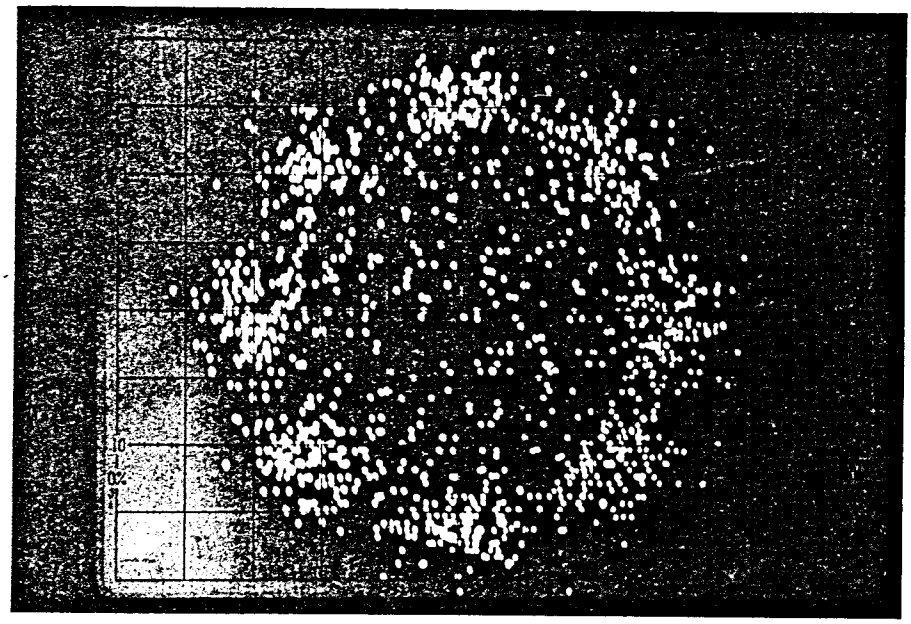


FIGURE 9.5
8-ARY SIGNAL CONSTELLATION IN FADING CONDITION

The multipath fades which the signal encountered can be pictured by the SNR of the received signal. Figure 9.6 depicts the instantaneous SNR and average SNR against the time of day for the experimental set up, employing a rate R = 1/2, (d, k, C) = (0, 2, 1) code; time of day meaning that the point 16.668, on the horizontal axis of the graph, represent the time 16h40, twenty to five pm, of the day the measurement was done. This received signal to noise ratio of figure 9.6 was recorded when the mobile unit was moving in a freeway environment, with an 8-ary PSK modulation scheme. The SNR fluctuated roughly between 23 dB and 12 dB, with an average SNR of 21.7728 dB. (Figures 9.4 and 9.5 were thus photographed when the mobile unit was in one of the low SNR areas of figure 9.6.)

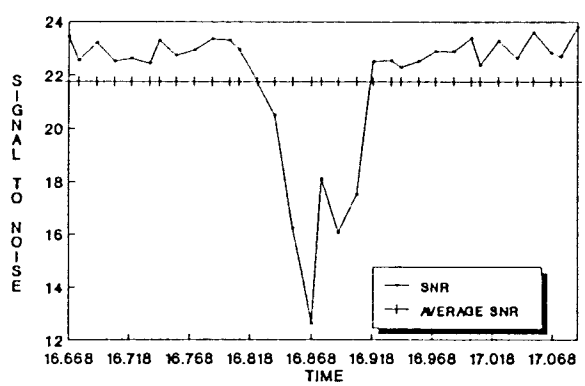
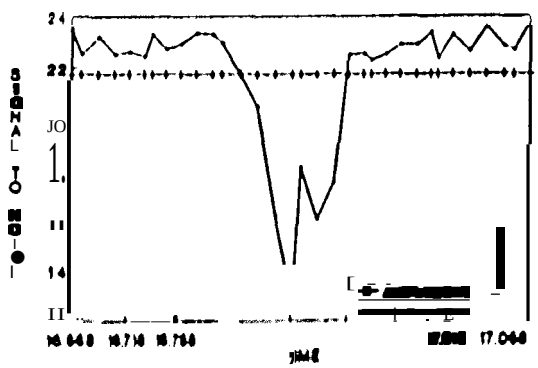


FIGURE 9.6
SNR OF (d, k, C) = (0, 2, 1) CODE ON HIGHWAY

FIGURE 9.5
8-ARY SIGNAL CONSTELLATION IN FADINO CONOmON

The multipath fades which the signal encountered can be pictured by the SNR of the received signal. Figure 9.6 depicts the instantaneous SNR and average SNR against the time of day for the experimental set up. employing a rote R = 1/2, (d, k. C) = (0. 2. 1) code; time of day meaning that the point 16.668. on the horizontal axis of the graph. represent the time 161140. twenty to five pm, of the day the measurement was done. This received signal to noise ratio of figure 9.6 was recorded when the mobile unit was moving in a freeway environment. with an 8-ary PSK modulation scheme. The SNR fluctuated roughly between 23 dB and 12 dB. with an average SNR of 21.7728 dB. (Figures 9.4 and 9.5 were thus photographed when the mobile unit was in one of the low SNR areas of figure 9.60)



FIOURR9.6
SNR OF (d, k, C) =(0. 2. I) CODE ON J1I011WAY

Figure 9.7 also depicts the average and instantaneous SNR against the time of day, for the same code and conditions as figure 9.6, except thnt the mobile unit was in a city environment.

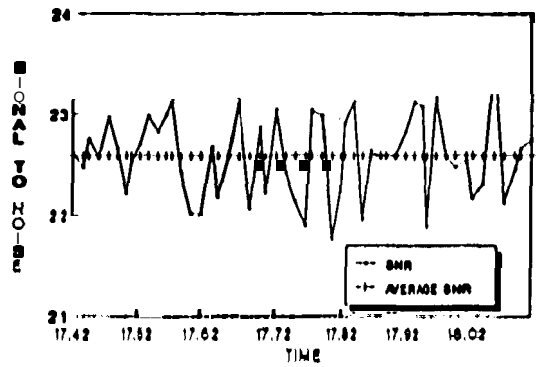


FIGURE 9.7
SNR OF $(n, k, C) = (0, 2, 1)$ CODE IN CITY

It is interesting to note that the SNR fluctuate roughly between 24 dO and 21.5 dB, with an average of 22.5924 dB.

A striking difference between the SNR's for a city and highway environment is the larger SNR fluctuations in a highway environment. An explanation for the previous observation is twofold, the first of which is the higher overage speed on a highway and thus a more dominant doppler shift on the received signal and secondly, the mobile unit travelled a longer distance from the transmitter; the signal strength gets weaker and is more affected by fading conditions. The recorded SNR's for the other codes and the PN-scquence had a similar progress in city and highway environments.

Table 9.1 shows the measured bit error rates and signal-to-noise-ratios for the five modulation codes and the PN-scquence. It must be noted at this stage, that the channel rate (code rate) was fixed for a given modulation scheme, *not* the data rates; coherent modems were used and the channel rote was thus dictated by the modem baud rotc. Since the code rates, R $= min$, varied as indicated, the codes had different data rates.

Consulting Stremlcr (31 J. 4-ary PSK offers a good trade-off between power and bandwidth, requiring very modest increases (OJ· 0.4 dB) in transmitted power for a potential doubling in bandwidth efficiency over that of coherent PSK. While an B-ary PSK modulation scheme offers a potential bandwidth efficiency of 3 bps/Hz, it exacts a SNR penalty of almost 4 dB

CODE	MODULATION SCHEME	ENVIRONMENT	SNR dO	DER x10-3
PN·scq	4-ary PSK	CITY	23.7	2.30
		HIGHWAY	22.2	4.88
	8-ary PSK	CITY	23.1	2.10
		JIIGHWAY	23.0	2.60
(0,1,1)	4-ary PSK	CITY	22.3	2.95
		HIGHWAY	21.1	6.29
	8-ary PSK	CITY	21.9	1.88
		HIGIiWAY	21.0	3.29
(0,2,1)	4-ary PSK	CITY	21.8	1.93
		HlGIIWAY	21.4	4.99
	8-ary PSK	CITY	22.6	1.05
		JIIGIIWAY	21.7	2.77
(I,3)	4-ary PSK	CITY	21.1	2.27
		HIGHWAY	20.2	3.29
	8-ary PSK	CITY	22.8	2.11
		HIGHWAY	20.1	2.57
(I,7)	4-ary PSK	CITY	23.1	2.33
		HIGHWAY	21.6	3.43
	8-ary PSK	CITY	22.7	1.84
		HIGHWAY	22.0	1.51
(2,7)	4-ary PSK	CITY	22.1	1.21
		HIGHWAY	21.3	2.98
	8-ary PSK	CITY	22.7	0.86
		HIGHWAY	21.9	1.36

. TABLE 9.1

MEASURED SNR's AND BER's

over that required for coherent PSK at error rates between 10^{-4} and 10^{-5}.

Interpreting table 9.J with the aforementioned in mind, table 9.1 seems incorrect; the bit error rates for 8·ary PSK arc, without exception, better than the corresponding bit error rates for 4-ary PSK, with approximately the same SNR! This highlight yet another important property of a mobile communication channels; the difference between fast fnding and slow fading (chapter 2). (Fast fading as opposed to slow fading; more fades per bit interval.l Since 4-ary PSK was transmitted at 1200 baud and g·ary PSK at 1600 baud.

the former was more subjccted to fast fading (longer bit interval). while the laner was more subjected to slow fading (shorter bit interval). Natumlly, this interesting trade·off between faster baud rotc and better DER could not be achieved indefinitely. A certain threshold point is thus anticipated where the DER will increase sharply as the baud rate is increased; jitter and noise have more pronounced effects at higher baud rates in bandlimited channels.

μ	$P(\mu, 31)$	$P(\mu, 15)$	$P(\mu, 7)$
0	0.978391	0.986730	0.991035
1	0.004299	0.003010	0.002571
2	0.003485	0.002404	0.002256
3	0.002819	0.001930	0.001899
4	0.002276	0.001568	0.001334
5	0.001833	0.001285	0.000672
6	0.001473	0.001043	0.000203
7	0.001181	0.000809	0.000026
8	0.000945	0.000573	-
9	0.000756	0.000354	-
10	0.000605	0.000182	-
11	0.000484	0.000075	-
12	0.000386	0.000023	-
13	0.000306	0.000005	-
14	0.000238	0.000000	-
15	0.000179	0.000000	-
16	0.000129	-	-
17	0.000088	-	-
18	0.000056	-	-
19	0.000032	-	-
20	0.000017	-	-
21	0.000008	-	-
22	0.000003	-	-
23	0.000001	-	-
24	0.000000	-	-
25	0.000000	-	-
26	0.000000	-	-
27	0.000000	-	-
28	0.000000	-	-
29	6.1E-11	-	-
30	3.9E-12	-	-
31	1.2E-13	-	-

TABLE 9.2

$P(\mu, \nu)$ FOR RATE $R = 1/2$. $(d, k, C) = (0, 2, 1)$ CODE

4-ARY PSK IN CITY ENVIRONMENT

To sum up: in a city environment the average SNR is higher (with a lower BER) than in a freeway environment, with marc acceptable SNR fluctuations: an g-ary PSK modulation scheme at 1600 baud would be recommended for use on a mobile communication channel, irrespective of the modulation code used.

Before recommending an appropriate modulation code (or mobile VHF communication channels, let us first look at the measured probability that a block of ν code bits contain μ error bits. As example, figure 9.8 and table 9.2 depict, respectively, the measured probability, $P(\mu, \nu)$, versus the number of errors in v bits, μ, graphically and in tabulated form for a rate R = 1/2, $(d, k, C) = (0, 2, 1)$ code, with 4-ary PSK in a city environment. Three block lengths are presented on the graph and in the table: v =7, 15 and 31 (which concurs with the codeword length of several Hamming and nCII codes).

TIle quantative values of the probability $P(\mu, \nu)$, for the five modulation codes and the selected PN-scquence, for both modulation schemes in a city and highway environment, are tabulated in Appendix D (or block lengths o(v = 7, 15 and 31. ny comparing these values with the block error probability of a PN-sequence (Appendix D), a modulation code can be more accurately chosen.

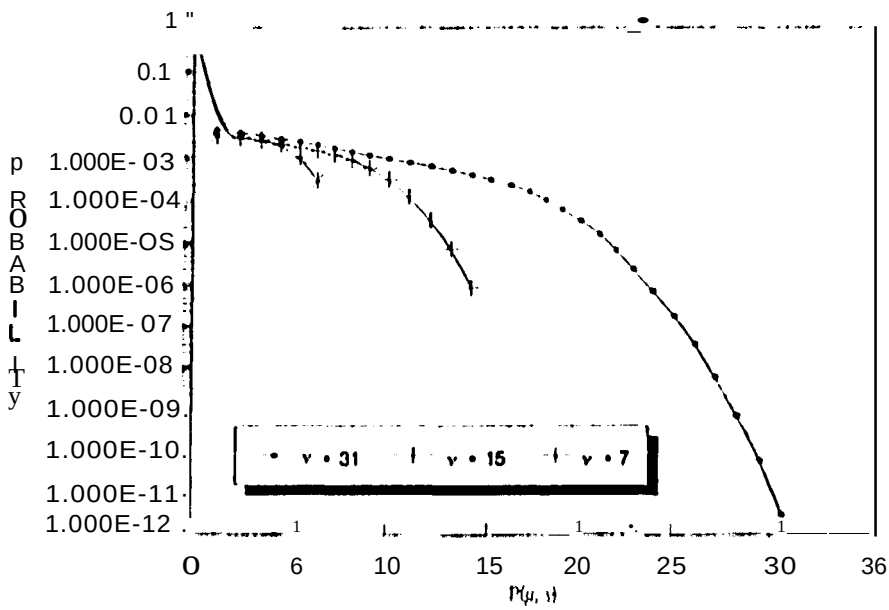


FIGURE 9.8

$P(\mu, \nu)$ FOR RATE R = 1/2, $(d, k, C) = (0, 2, 1)$ CODE

4-ARY PSK IN CITY ENVIRONMENT

As mcntioned earlier, the rate R = 1/2, $(d, k, C) = (0, 1, 1)$ Manchester code is already in use on mobile communication channels 15). [47] and the CCIR recommended (J6] the rate R

$=1/2$. $(d, k) = 0.3$) Miller code for use on mobile communication channels to achieve clock extraction and a bandwidth saving in an environment with an ever increasing spectrum shortage. Hence. this study set out to investigate the CCIR recommendation and to further investigate modulation codes for clock extraction, improved intersymbol interference properties and bandwidth saving. Two of the three properties investigated were fruitful; clock extraction can be gained because of regular transitions in the transmitted waveform (due to D bounded k parameter) and intersymbol Interference can be reduced by choosing a proper *(I* pnrarncter, the third property. however. cannot be achieved with modulation codes, since a fixed data rate result in a bandwidth increase of nlm > lover an uncodcd PN-scquence (chapter 8). Thus, when choosing a modulation code for a mobile VHF channel, these advantages and restrictions must be kept in mind.

Since the signal constellation (modulation scheme employed on carrier) has lillie effect on the modulation code performance, the following discussion will only consider the 4-ary PSK entries in the tables rcferenccd. Each modulation code will be compared with an uncoded PN-sequcnce. using the BER's of table 9.1 and block error probabilities of the tables presented in Appendix D. The spectral efficiency of each modulation code, in comparison with the PN-sequcnce, will also be discussed. The spectral efficiency of a modulation code equal its rate $R = mln$. (Spectral efficiency refers to the number of bits that are transmltted in a given period of time, usually one second. over a radio channel with a defined bandwidth. For instance, a PN-scquence transmitted at 1200 baud with a 1200 Hz-bandwidth has a spectral efficiency of one bit per second per hertz.)

- The rate $R = 1/2$, $(d, k, C) = (0, I, 1)$ code versus the PN-scquence:
  The bit error rate for this modulation code in both the environments is the worst; the spectrum efficiency for the modulation code is 1/2 bps/Hz as opposed to 1 bps/Hz for the PN-sequence; the probability of no errors in blocks of $\nu = 7$, 15 and 31 bits (the first entries in the tables in Appendix 0, tables 0.2, 0.7, 0.21 and 0.22) are lower. Thus, except clock extraction, not very much else is gained from this code; hence, there is hope for a better choice of modulation code over the presently used Manchester code.

- The rate $R = 1/2$, $(d, k, C) = (0, 2, 1)$ code versus the PN-scquence:
  Considering table 9.1, the modulation code and PN-scquencc have similar BERs in a city and highway environment; the spectrum efficiency for the modulation code is 1/2 bps/Hz. Although this may seem low. the bandpass results in chapter 8 showed that a higher data rate can be achieved with this

modulation code if a custom modem is designed to take advantage of its special spectrnl properties. P(O, v) (v = 7. 15 and 31) arc lower for the modulation scheme when the first entries in tables 0.1.0.6. 0.21 and 0.22 arc considered. A major improvement of this code over the Manchester code, is the smaller bandwidth required (chapter 8). Whcn compared to the PN·scquence. clock extraction can be gained in almost the same bandwidth constraint,

- nle rate R = 1/2. (d, k) = (t, 3) code versus the PN-scquencc:
  With this modulation code the influence of a larger d parameter was investigated. A slight improvement in the nER nnd basically the same P(O, v) (tables 0.3. 0.8, D.21 and D.22) were encountered. The spectral efficiency is 1/2 bps/liz. There is thus an inclination to a better code performance with a larger d parameter.

- The rate R = 2/3, (d, k) =(1, 7) code versus the PN-scquencc:
  This modulation code had a similar performance to the Miller code previously described; a slight improvement in the BER and very much the same P(O, v) (tables D.4, 0.9, 0.21 and 0.22) were encountered, The spectral efficiency, however, was improved to 2/3 bps/llz. As mentioned in chapter 8, this code was chosen to verify the influence of a larger detection window over mobile radio channels. Although a larger detection window plays a significant role on magnetic and optical recording channels (chapter 4), it has no significant influence on mobile radio channels.

- The ratc R = 1/2, (d, k) = (2, 7) code versus the PN-scquencc:
  This modulation code performed the best on mobile mdio channels. The HER and P(O, v) (tables 0.5, 0.10. 0.21 and 0.22) were better. The spectral efficiency. however, is 1/2 bps/llz. It is thus evident that the larger d parameter has an influence on a mobile radio channel. This can be attributed to the better resistance against intersymbol interfcrence brought about by the larger d parameter. However, a larger d parameter results in a lower spectral efficiency, since the code rate is lowered accordingly. A possible solution to this problem will be presented in chapter 10 where recommendations (or futuro research will bediseussed.

With the advantages and disadvantages of modulation codes discussed in terms of an uncoded PN-scquence, it is possible to recommend a modulation code for usc on mobile communication channels.

TIle *(0,1:)* codes did not pcrfonn vcry well on a mobile radio channel. The influence of the DC-free property also did not influence the codes to perform better since indirect modulation was used; the modulation code spectrum was frequency shifted by the modem to a higher frequency band, thus canceling the need for DC·freencss. When spectral efficiency is considered, the rate R =1/2, (d, k, C) =(0, 2, 1) code and the rate R =2/3, (d, k) =O. 7) code take the lead. Since the rnte R = 2/3, (d, k) = O.7) code has better block error probability. with approximately the same bandwidth requirements, this code would be preferred over the DC·freecode. The rate R =]/2, (d, k) = (2. 7) code performed the best when error properties are considered, because of the larger d parameter.

With all this in mind it is evident that a better code can be recommended than the presently used Manchester code and CCIR proposed Miller code. Since clock extraction can be gained from the rate R = 1/2, (d, k, C) = (0,2, 1) code and the rate R =2/3, (d, k) = (1,7) code. with good error characteristics, these codes would definitely be recommended for (uture use on mobile radio channels.

If the proposed solution in chapter 10 is found to be valid, modulation codes with a larger d parameter can beused in future to improve intersymbol interference problems.

9.3) CHANNELMODELS

Figure 9.9 shows a graph depicting the burst distribution, burst interval distribution. cluster distribution and error free run for the rate R =1/2, (d, k, C) = (0. 2, t) code in a city environment, with a 4-ary PSK modulation scheme. Appendix D contains similar graphs for the other five modulation codes in city and highway environments for both the modulation schemes.

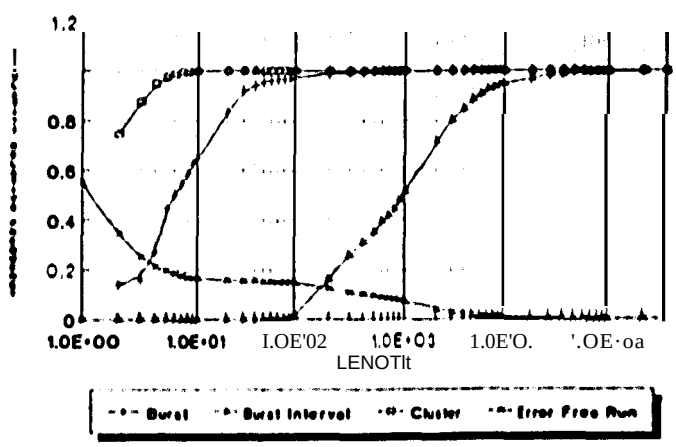


FIGURE 9.9
ERROR DISTRmunoN FOR A RATE R =112. (d, k, C) =(0.2. 1) CODE
4-ARY PSK IN CITY ENVIRONMENT

111is measurements were conducted as shown in figure 9.10: the modulation codes together with the mobile VHF channel were considered as a black box containing the channel. This representation of the channel is known as a *super channel.*

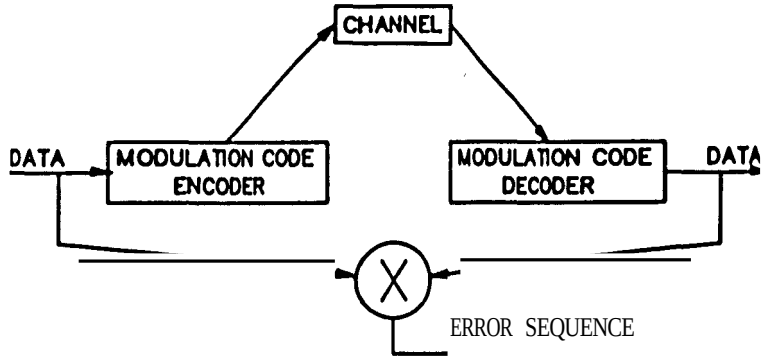


FIGURE 9.10
MODELING OF SUPER CHANNEL

Although this study is not concerned with channel models, this measurements will enable future students to apply the work of Swarts [7], Tsai [38] and Fritchman [39] to find channel models for modulation codes on mobile communication channels. These results are also available on a floppy disk in the cover of this thesis. The filenames which contain these results have the following legend: the first and second letter respectively indicate the modulation scheme and environment, The next letters indicate the (d, k) or (d, k, C) code parameters. For instance, a file with the name 4110_1_2.TXT contains the results for the rate $R = 1/2$; $(d, k, C) = (0, 1, 2)$ code, with a 4-ary PSK modulation scheme in a highway environment.

CHAPTER 10

CONCLUSIONS AND RECOMMENDATIONS

This thesis contains the results of an investigation devoted to a theoretical and experimental study of communication systems, with contributions to the information theory and mobile communication disciplines.

The theoretical results obtained in this thesis can be found in chapters five and six. Chapter five describes a method to map fixed-length coding rules on finite-state machines, realized with the OW algorithm. Although the algorithm can be applied with success to both binary and multilevel coding rules, the algorithm has the restriction that it can only be applied to *fixed-length* coding rules, such as table 4.4, and not to variable-length coding rules, such as table 4.5. The algorithm should thus in future be extended or amended to incorporate both fixed-length coding niles and variable-length coding rules, or a new algorithm specifically devoted to variable-length coding niles may have to be developed; this, however, may not be a trivial task! Although there is mom for improvement in the OW algorithm, it gives future research in this field a good starting point. In the literature there are numerous coding rules that need to be converted to FSM representations, in order to compute spectra e.g., thus making this investigation and the DW algorithm a worthwhile contribution to the information theory literature.

Chapter 6 presented new results for (O.k) modulation codes. together with the 1'8 algorithm. which can be used in future when other (0. k) codes need to be developed. As mentioned. 8 timesaving with the TS algorithm for large" is also anticipated over methods presently used, making the 1'8 algorithm a meaningful contribution to the information theory literature. Although the rate R = 11/12. (ti. k) =(0.3) code developed with the 1'8 algodihm was not selected for experiments on the mobile communication channel. since it can be used on magnetic recording channels or even on optical recording channels. In future the error propagation properties etc. for this code can also be investigated.

Chapter 8 was one of two chapters on the experimental results obtained in this study. Experimentally measured baseband spectra of fourteen modulation codes and the selected PN-scquence are presented on a linear and logarithmic scale. The measured spectra compared favorably with known theoretical spectra found in the literature. see e.g. Immink [3] and Zehnvi and Wolf [48]. to mention ıı few. However. seven modulation codes' spectra were measured that were previously unpublished. the rate R = 8/9. (d. k) = (0.3); the rate R = 11/12. (d, k) =(0,3); the rate R =1/2, (d. k) =0.2); the rate R =1/3. (d, k) =(3,7); the rate R =1/4, (d, k) = (4. 7); the rate R = 1/4. (d. k) =(4,8); and the rate R =1/4, (d, k) = (5.9). The two (d, k) = (0. 3) codes with different code rates had, as anticipated. very similar spectra. see figures 8.9 to 8.12. with the rate R = 1]/12 code needing a smaller bandwidth. The measured spectra give the reader a feeling of the influence of the d, k and C parameters on the energy content at a specific frequency. These measurements also verified the expected influence of these parameters on the frequency characteristics of the code as described in chapter 2; although these parameters arc interrelated. the d parameter has a bearing on the high frequency components of the modulation code spectra. the k parameter determine the low frequency components of the modulation code spectra. and the C parameter determine the accumulated charge of the waveform.

Spectra of modulation codes after FFSK and PSK modulation were also investigated. FFSK modulation is known for its symmetric spectra (resulting in a smaller bandwidth than FSK) which is verified in figures 8.35 to 8.40. It is evident that the DC·free property, present in some modulation coxles, is not a desired property when frequency modulated, since most of the signal energy is needed at the center of the passband, When considering figure 8.35. the FFSK spectrum of the PN·sequence, it is clear that most of the energy is concentrated around the carrier frequency. a desired property. When the FFSK spectra of the rate R =1/2. (d, k, C') = (0. I, 1). Manchester code and the rate R = 1/2. (d, k, C') =(0.2. Π. IIcdeman 11·2 code are considered, figures 8.36 and 8.37. it is evident that most of the energy is concentrated 8t frequencies removed from the cnrrier frequency. Considering the rnte R = 1/2, (d, k) = O.3)

Miller code, the rate R = 2/3. (d, k) = O. 7) Jackoby-Kost code and the rate R = 1/2. (d, k) = (2. 7) IBM code, figures 8.38 to 8.40. it is clear that most of the energy of the signal are concentrated around the carrier frequcncies. The Jackoby-Kost code has a little less energy concentrated round carrier frequencics than the other two codes.

Against this backdrop. it is evident that a certain class of modulation codes would be preferred over an uncoded PN·scquence when energy need to be concentrated round the carrier frequencies. However. this is accomplished at the expense of a lower data rote.

It is interesting to note that the modulation codes did not influence the PSK spectra as much as it influenced the FFSK spectra. It was impossible to distinguish betwecn the various modulation codes and the PN·sequence after PSK modulation. 111is can be ascribed to the PSK spectra (DPSK. 4·ary PSK and 8·ary PSK) that do not have any sharp peaks in the spectrum (no large amounts of energy concentrated at certain frequencies). figures 8.37 and 8.39.

The conclusion can thus be drawn. when comparing FFSK and PSK spectra. that PSK modulation would be preferred over FFSK as modulation scheme when using modulation codes. PSK modulation gives us more freedom in the choice of a modulation code. since the energy of the modulation code is spread out over the whole PSK spectrum, which is not true in the FFSK case,

The bandpass experiment results presented can be used as reference to recommend a modulation code through a bandlimited channel. Although this list is by no means exhaustive. a proper understanding of the influence of the d. k and C parameters through bandpass channels can be gained. (By considering the bandwidth of a given channel the best choice of d, k and C parameters can be determined with table 8.2 at hand.)

111e bandpass filter settings (LPF/HPF) compare best with a mobile communication channel and can be used in future to predict and/or simulate the actual mobile communication channel in laboratory conditions. In this way the actual mobile channel. or for that mauer any other channel. can be classified and verified in a more controlled manner in the laboratory. allowing more reliable conclusions to be drawn regarding the recommended choice of specific modulation codes best suited for the application. These results is thus only a preliminary study and can be repeated in order to achieve more extended results,

Chapter 9 contains results obtained by transmitting modulation codes over mobile VHF channels, Except for a first-hand acqualntance with the mobile communicntion channel and environment. these experiments also made it possible to recommend a certain class of modulation codes and modulation schemes to improve reliable data trnnsmission over these channels,

The error distribution of the five modulation codes were measured to verify the experimentnl results when a proven model of a digital mobile communication channel is available. These results are presented in appendix D.

When clock extraction is needed over a mobile communication channel, a modulation code has to be considered. The choice of modulation scheme depend, to a great extend, on the available bandwidth, the data rate to be achieved, the modulation scheme used and the allowable nER. Table 9.1 and the bandpass results obtained in chapter 8 can be of great assistance when a modulation code is needed. The results of chapter 8 can be a guide to choose a modulation code for a certain bandwidth. Since a high code rate (R) result in a theoretical smaller bandwidth (when compared to a code with a low code ratel, a code with a high rate ($R = m/n$) would be preferred. As previously discussed, PSK modulation would be used with a modulation code since the modulation code spectra is spread out over the whole PSK spectrum. When a low DER is needed, a code such as the rate $R = 1/2$, $(d, k) = (2, 7)$ IBM code would be considered, apparently because of the larger d parameter which reduced intersymbol interference.

Although the PN-scquence resulted in a bit synchronization loss during fading, this phenomenon is not reflected in the results of table 9.1; the error recording equipment did not count bit errors during synchronization loss.

As mentioned, the influence of a larger detection window and higher density ratio was also investigated over a mobile radio channel. A larger detection window had no apparent influence on a mobile communication channel, whilc a higher density ratio, larger d parameter, resulted in a lower nER.

The global conclusion thnt can he drown from this study would he that modulation codes are necessary when clock extraction is needed, hut can be implcmented more efficiently on mobile communication channels. To implcmcnt a modulation code on these channels, depend to a great extend on the type of data which need 10 be transmiued. When data is transmitted, like a PN-scqucnce, where there would not be nny consecutive runs of zeros or ones longer than 20

to 30 bits. modulation codes would not be considered. However. when. (or instance, computer data is transmltted, where o nan of many consecutive zeros or ones ore transmitted, modulation codes would be essential. Even then modulation codes can be implemented more efficiently, which will be discussed shortly.

However, this study showed that a better modulation code than the presently used Manchester code can be recommended. 111e CCIR recommended the Miller code (or a more efficient usc of bandwidth. which proved to be unsuccessful; the Miller code need twice the bandwidth of uncoded data and the same bandwidth as the Manchester code.

With nil the previously mentioned in mind. the five modulation codes can be rank, in order of merit. as follows:

1) Rate R= 1/2. *(d. k, C)* =(0,2. 1);
2) Rate R= 2/3. *(d. k)* = 0.7);
3) Rate R=J/2. *(d. k)* =(2. 7);
4) Rate R= 1/2. *(d. k)* = 0,3);
5) Rate R= 1/2. *(d. k, C)* =(0, 1. 1).

A few meaningful suggestions can be made concerning modulation codes on mobile radio channels. TIle first would be to use direct modulation of the VHF radio carrier; since digital radio will be the next generation of mobile communication systems, the *d* and *k* parameters can be investigated under these conditions.

For n more efficient use of modulation codes. the modulation codes need to be integrated

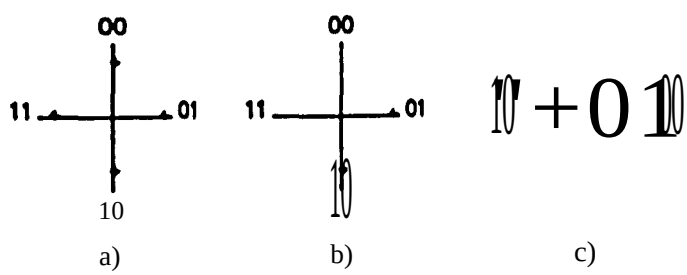


FIGURE 10.1
a) UNCODED 4-ARY PSK b) CODED 4-ARY PSK c) OPTIMAL CODED 4-ARY PSK

with the modulation scheme used. For example. consider a 4-ary PSK modem with signal constellation as shown in figure 10.1a. By choosing a rate R = 1/2. (d, k) = (0,2) modulation scheme which consists of two codcwords 01 and 10. the signal constellation will look like figure lO.1b when transmitted through the -t-ary PSK modem. By designing a dedicated modem the two points can be placed at optimal position as indicated in figure lO.lc. We now have a scheme with a noise immunity similar to 2-ary PSK. clock extraction and the same bandwidth constraint as uncodcd data; in short a desired modulating system! A larger *d* parameter, in the same bandwidth constraint. can also be incorporated with a scheme like this.

This is only a trivial example, but the real power in a scheme like this will become more apparent when the number of points are not equal to *t*.

The usc of (d, k) modulation codes on mobile radio channels can also be integrated with compatible error correction to improve the DER, as well as bandwidth and lSI limiting strategies to enable high data rates on narrowband channels. The latter cnn be achieved by limiting the intersymhol interference variation (ISV) in the same way the digital sum variance (DSV) is limited to construct DC-free codes. Apart from the above. codes can also be designed to simultaneously achieve maximum Hamming distance (or improved error performance.

The usc of partiaI response techniques can also be investigated to achieve better bandwidth efficiencies, with the additional possibility to incorporate error detection/correction with Viterbi decoding to improve the DER.

APPENDIXA

HARDWARE DESCRIPTION

Developments in the field of electronics have constituted one of the great success stories of this century. Beginning with crude sparkgap transmitters and "eat's-whisker" detectors at the tum of the century, we have passed through a vacuum-tube era of considerable sophistication to a solid-state era in which the flood of stunning advances shows no signs of abating. Calculators, computers and even talking machines with vocabularies of several hundred words are routinely being manufactured on single chips of silicon as part of the technology of *large scale integration* (LSI) and current developments in *very large scale integration* (VLSI) promise even more remarkable devices.

Perhaps as noteworthy is the pleasant trend toward increased performance per rand. The cost of nn electronic microcircuit routinely decreases to a fraction of its initial cost as the manufacturing process is perfected. In fact, it is often the case that the panel controls and cabinet hardware of an instrument cost more than the actual electronics inside.

Since Claude Shannon systematized and adapted George Boote's theoretical work in 1938. there has been unprecedented growth in the application of digital concepts. Other fields that emerged in the late 1930's and early 40's have "peaked" and leveled or declined, while the application of digital concepts is stilt growing exponentiatty. Each day digital concepts

are being applied to problems that could only be solved by analog methods several years ago. Fast, reliable nnd modestly priced *analog to digital (NO)* and *digital to analog (D/A)* converters are presently available, facilitating the application of digital concepts for solving complex analog problems using microprocessors and other programmable digital systems. In short, the rapid expansion of discrete practices has served notice to the academic community to restructure curricula to treat discrete mathematics and other discrete sciences. Certainly the "microprocessor" revolution has penetrnted all fields of endeavor and will continue to do so for many years.

Since digital techniques are cheaper, easier and more convenient to usc in a design than most other techniques, the designs undertaken in this study will to a great extend implement digital techniques, including *digital signal processors* (DSP), *mlcrocontrollers, programmable logical devices* (PLD's), *random access memory* (RAM), *read only memory* (ROM) and a handful of the common garden variety *transistor-transistor logic* (TTL) gates, counters, shift-registers etc.

Although sheer "number crunching" is an important application of digital electronics, the real power of digital techniques is seen when digital methods are applied to analog (or "linear") signals and processing. Some elementary analog techniques in the form of low-pass, high-pass, and band-pass filters were also implemented,

As is evident in the walk of life, hybrids do exist. *Phase-locked loops* (PLL) is one of the "hybrids" in the electronics world, where analog and digital techniques are combined, giving us the best of two worlds. A thorough discussion concerning the lock runge, capture transient anel low-pass filter design for a given set of parameters will be presented. This section may well be used as a tutorial guide for future students on the design of PLL's.

With the design of digital systems a few obvious questions arise; when must a DSP processor be used instead of an "ordinary" microprocessor (μP), which type of RAM must be used, static or dynamic, must *complimentary metal oxide semiconductor* (CMOS) technology with their low power consumption be used instead of the faster TTL technology, must the black art of PLD's be conquered? Answers will be given in a corroborate discussion where tradeoffs between cost, efficiency and simplicity wage the overtone.

This chapter will describe the three designs mentioned in chapter 7 on a circuit diagram level, together with a conflnnative discussion why certain components were used,

A.t) PROGRAMMABLE LOGIC DEVICES

As system design methodology continues to evolve, there becomes nn increasing need to not only simplify the design process, but to reduce the overall system size and cost, and increase system reliability. It was this mlndset that led to the development of the first programmable logic devices. In fact, the evolution of programmable logic has changed the way systems are designed, since it offers the designer a single tool that solves nil his needs. Programmable logic is ideal for simplifying the design process because the designer can implement the exact logic functions wherever and whenever required. It is also ideal for reducing system size and cost by offering significantly higher functional density than its *small-* and *medium-scale-integration* (SSI/MSt) predecessors. Finally, system reI inbitity is significantly improved because of simplified designs and lower parts count.

The *programmable array logic* (PAL) device is an extension of the fusible link technology used in bipolar *programmable read only memory's* (PROM's). The fusible link PROM first gave the digital systems designer the power to "write on silicon". In a few seconds he was able to transform a blank PROM from a general purpose device into one containing a custom algorithm, microprogram or Boolean transfer function. This opened up new horizons for the use of PROM's in computer control stores, data storage tables and many other applications.

The PAL device extends this programmable flexibility by utilizing proven fusible link technology to implement logic functions. By using PAL circuits the designer can quickly and effectively implement custom logic varying in complexity from random gates to complex arithmetic functions.

Although PAL devices are superior to PROM's and even *erasable* PROM's (EPROM) they have disadvantages; it offers high speed, but is saddled with high power dissipation because of its bipolar fuse-link tcchnology. This not only significantly increases system power supply and cooling requirements, but also limits the ability of high functional density. Another shortcoming of this technology is the one-time-programmable fuses; no reuse in the event of mistakes during prototyping or errors on the production floor are possible, and any misprogrammed devices must be dlscanlcd.

Ultra Violct CMOS (UVCMOS) addresses many of the shoncomings of the bipolar approach, but introduccs many shortcomings of its own. This tcchnology requires much lower power and, while it has the capability to erase, this comes at expense of slower speeds

and cumbersome erase procedures: exposing the device to ultraviolet light for at least 20 minutes. Additionally. the devices must be housed in expensive windowed packages to allow users to erase them.

Of the three major technology approaches available, *elcctrlcal erasable* CMOS (E^2CMOS), UVCMOS and bipolar. the technology of choice is clearly E2CMOS. for many reasons. including: testability. quality, highspeed, low power. and instant erasure for prototyping and error recovery. *Generic Array Logic* (GAL) devices utili7.c this technology. GAL devices are ideal programmable logic devices because. as the nome implies. thcy are architecturally generic, These devices employ the *Output Logic Macrocell* (OLMC) approach [40]. which allows users to definc the architecture and functionality of each output. 'Thc key benefit to the user is the freedom from being tied to any specific functionality. Comparing the GAL device with fixed-architecture programmable logic devices is much like comparing these same fixed PLDts with SSI/MSI. '111c GAL family is the next generation in simplified system design. Thc user needs not bother searching for the architecture that best suits a particular design. Instead. the GAL family's generic nrchltecture lets him configure as he goes.

Each OLMC can be individually set to active high or active low. with either combinational (asynchronous) or registered (synchronous) configurations. A common output enable can be connected to all outputs. or separate inputs or product terms can be used to provide individual output enable controls. The OLMC provides the designer with maximal output flexibility in matching signal requirements, thus providing more functions than possible with existing 20-pin PAL devices. It should be noted that actual implementatlon is accomplished by development software/hardware and is completely transparent to the user.

In the designs presented. GAL devices were exclusively applied where PLO's were thought to be necessary, because. as ilhistrated in figure A.t. the GAL is a great pcrformerll

A.I.l) DESIGN IMPLEMENTATION OF GAL'S

Thc tools required for designing with GAL products can be separated into two categories:

- Programmable logic development software:
- Device programmers.

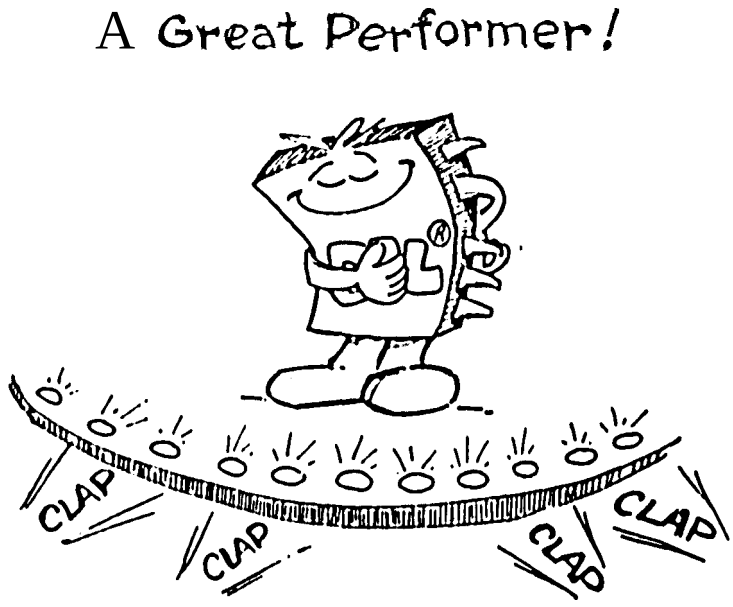


FIGURE A.1
THE GAL, A GREAT PERFORMER!

Universal programming hardware allows the progrnmming of a variety of devices without the aid of custom fixtures or manufacture's adapters. Since the GAL programming algorithm requires no abnormal voltages or timing, as some one-time programmable technologies do, most nil hardware manufacturers support GAL devices on existing models.

Software packages, such as ABEL from Data I/O, CUPL from Assisted Technology and Logic Lab from Programmable Logic Technologies, offer generic development support for all programmable logic devices, and, with periodic updates the user will be kept up on all programmable logic device developments from nil manufacturers.

'111e software offer a PC-based PLD programming langunge, suitable for programming various PLD devices. Once the software knows which device will be used, fields are provided for optional information, such ns design description etc. The device pinout and pin labels need to be specified. Entry of the logic functions is next. Traditionally, this entry is in the fonn of Boolean equations. Current revisions of development software allow truth-table, state-machine and schcmauc-emry formats, as well.

111e development software and device progrnmmer used was the Logic Lob package, intended for programming most PLOts on the market today. An example of the afore

mentioned procedure for programming the devices can be found at the end of this appendix.

A.2) Wlllcn PROCESSOR?

Amid the plethora of alternative microprocessors currently available, it is sometimes difficult to come to a rational decision on the best choice of processors for a given task. There are two mutually opposed common misnpprehensions on this subject; the first is that certain CPU's are generally "beuer" than others. '11e second, that there is lillie to choose between devices and nny CPU will perform any task equally well given the right software.
The truth lies somewhere between these two poles. The choice of processor for a given task depends primarily on the application designer's criteria in a givcn design situation. Alternative factors which could guide or dictate choice are:

- System chip count in a low cost commercial application;
- System throughput as an absolute parameter;
- System efficiency as an absolute parameter;
- System cost effectively represented as throughput per rand cost:
- System adaptability for non-standard implementations;
- System reliability or the capacity of the system for self-maintenance:
- Designer familiarity with the system.

The list could go on indefinitely. but this sample shows just how diverse are the factors which seem significant in varying design situations.

It is important to be clear on the distinction between microcomputers. microprocessors and microcontrollers. A microprocessor is the CPU part of a computer without the memory, I/O and peripherals needed for a complete system. All the other chips in a microcomputer. such as the IBM PC, arc there to add features not within the microprocessor itself.

When a microprocessor is combined with external I/O and memory. the combination is called a microcomputer. A device having the I/O and memory peripheral functions on the same substrate as the CPU to make a complete microcomputer is called a *Single-Chip Microcomputer* (SCM).

Generally. SCM's are designed for very small computer based devices that do not require all the (unctions of a full computer system. In cost sensitive control applications, even the

few chips needed to support a CPU like a 8088 or Z80 may take too much space and power; instead, designers often employ a SCM to handle the control-specific activities. Where single-chip micros arc designed or used in industrial control systems, they are often called microcontrollers. Basically, there is no difference ,between single-chip microcomputers and microcontrollers.

Sometimes the term "embedded controllers" is used instead of microcontroller; Intel, for instance, has adopted the tenn for its controller chips. However, an embedded controller, according to one definition [41], is a computer system hidden within some other device. By another definition [421, it's a computer whose programs cannot be altered by the user. Generally, the term embedded controller suggests a highly compact, although not very powerful, dedicated processor; for example, an SCM controlling a microwave oven.

Intel introduced its first microcontroller architecture, the 8048, in 1976. It was designed for general-purpose 8 bit control, with on-board ROM and RAM, plus 27 I/O lines. Four years later came the 8051, which was up to ten times faster than the 8048 and had a 1μs instruction cycle at 12Mllz.

Intel and other companies sell variations of the 8051 family enhanced with more internal memory, more *VO,* lower power and so forth. All members of the 8051 family have the same core instruction set, but some have one or two additional instructions for features unique to the particular chip. The 8751 is an EPROM version of the 8051 whose on-chip program memory can be electrically programmed and can be erased by exposure to UV light.

The *mobile* decoder used a 8751 EPROM as its "brain", while the encoder used yet another version of the famous 8051, the DSSOOO. The DSSOOO offers "softness" in all aspects of its application. This is accomplished through the comprehensive use of nonvolatile technology to preserve all infonnation in the absence of the system power. Initial loading of the application software into the DS5000 is possible from either a parallel or serial Interface to a host system. Serial loading uses the on-chip serial I/O port to accept incoming data from a host computer with a RS232 port, such as o PC-bnscd development system. 111e device program can thus be rcprogmmmed and altered instanteneousty,

111e reason why only the encoder used the DSsOOO is of an economical nature. All development for both the encoder and decoder was done on the more expensive DS5000, for apparent reasons, and thcn transferred to the 8751 for the decoder.

A.2.1) DIGITAL SIGNAL PROCESSING

Application speeiflc microcomputers and microprocessors, known today as *DIgIta! Signal Processots* possess two important characteristics which distinguish them from ordinary microprocessors. Firstly, they can execute almost their entire Instructlon set in one cycle. Secondly, they arc designed to execute a complete *Mulliply and Accumuillfe* (MAC) instruction also in one clock cycle (By comparison, the previously mentioned 8051 execute a multiply instruction in *48* clock cycles). Numerical algorithms by which DSP applications arc normally characterized are very MAC intensive.

The all important deciding criterion in DSP is real-time-processing. With the continued improvement in dedicated DSP solutions in real speed, architecture and application specific instruction sets, new market segments arc opening and old ones are expanding.

The majority of electronic applications are concerned with the manipulation of signals: filtering of unwanted components from an input signal, extraction of information from a signal, generating waveforms, modifying the amplitude characteristics of an output signal in order to improve the information content. etc. In the past all this manipulation was performed using analog circuits and techniques.

Analog circuits require a special design for almost every application and are therefore not particularly well suited to VLSI applications. Especially when the problem is considered from the aspect of using common analog VLSI function elements.

All analog designs are dependent on sensitive components, ie, resistor and capacitor values are never completely accurate. but more importantly, their values change as a function of time, voltage and temperature.

Although DSP's main application are quantization of an analog signal into digital values and performing a precomposed algorithm stored in memory on the digltized data, the DSP npplication for the purpose of this study was one where real-time processing was needed: a finite-stnte machine generator was to he employed, generating coded data at rates varying (rom a few baud to a possible 1M baud, depending. or course, on the complexity of the coder.

Since the DSP finite-stale machine can operate up to an estimated IM baud, a piece of equipment was thus developed that could be used for at least a few generations of post-graduate studies in real-time coded data.

A.3) RANDOM ACCESS STORAGE

The storage of information in computer systems is accomplished by utilizing collections of individual storage elements, each of which is capable of maintaining a single bit. Thus, for a device to be useful as n memory element it must have two stable states, a reliable mechanism for setting the device to one state or the other, and a mechanism for interrogating the state. Memories have been built of a variety of devices that match this characteristic, including relays, individual vacuum tubes, delay lines (which form a type of serial memory) and obviously semiconductors. In each case, information in the form of bits was entered into the memory, and then at some Inter time extracted for use by the system.

The technique of storing information by the magnetic orientation of a ferrous material was once used for the central memory of a computer [43]. Allhough this is not the case any more, this technique is now used more prevalently for other types of storage in a computer. The magnetic orientation of a region of ferrous material on a surface is used to store a bit, and this surface is most often on a rotating magnetic disk, or on a magnetic tape, (The various codes investigated in this study have implementations on this storage medium, see chapter 8.)

A.3.1) STATIC VS DYNAMIC

Static memories generally have a smaller number of bits per package and a higher power consumption than dynamic memories. The static mechanism of Figure A.2 requires six transistors in every cell; other static memory configurations utilize fewer active elements. One of the tasks of memory designers is to reduce the number of components needed in an individual storage cell, since fewer elements means that each individual cell can be smaller and require less power, which in turn leads to larger memories.

TIle memories with the largest capacities use not a static mechanism, but rather a dynamic mechanism, as shown in Figure A.1 Here the value of the bit is not determined by the

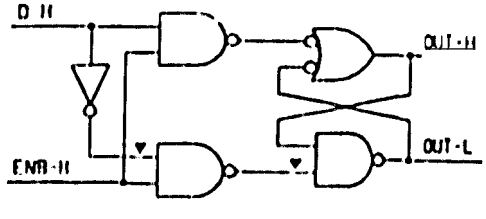
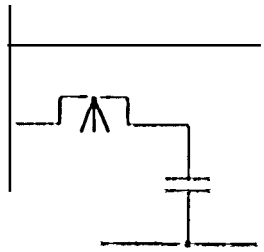


FIGURE A.2
STATIC MEMORY



DRAM CeII

FIGURE AJ
DYNAMIC MEMORY

current flowing through one of the two different paths. but rather the bit value is determined hy the amount of charge stored on a capacitor. The capacitor is created with semiconductor technology and is extremely small. TIle sensing of the charge is also vcry difficult and handled by circuitry on the device itself. 111e infonnation is placed on the capacitor by opening an electronic gate and establishing the proper charge level. Then, the gate is closed and the charge maintained on the node by the electronically isolating it from surrounding influences. However, the time which the charge can be reliably maintained in this manner is not very long, and so it must be re-established periodically. This is done by a "refresh" cycle, which detects the appropriate bit values and refreshes the bits. The length of time between refresh cycles varies from memory to memory, but a common value is 8 ms; each row must be visited at least once every 8 ms, For this reason dynamic memory controllers are designed to periodically access rows to ensure that the data is maintained in the memory cells.

With the afore mentioned as backdrop for selecting suitable memory for the nsp board, there was decided on static memory, since the memory requirements was 4k x 16 bit. Although dynamic memory is cheaper per I mega bit, the overhead is far to expensive when small quantities arc needed: the refresh controller would be more expensive than the actual memory usedI Also. it would complicate the design considerably.

AA) Tun TRI-STATE CONCEPT

As mentioned in chapter 7. there arc tri-state buffers between the PC. TMS and static memory on the DSP finite-state mnchine card, These arc needed to prevent nny clashes on the bus of either processor, The concept of tri·state buffering will now be discussed.

Figure A.4 shows the prevalent mechanism used to accomplish a tn-state action. This is

so called because the output can assume one of three states. Two of the states are the "normal" states of a TTL gate: low and high. The output will be low whcn the logic of the function creates a situation in which transistor "all is turned "on", and transistor "b" is turned "off"; the output will be high when transistor "011 is turned "off" and tmnslstor "b" is turned "on". The third state occurs when the logic of thc function creates a situntion where both transistors "a" and "b" are turned off. In this case. the output is electrically disconnected from the system, since the paths through transistor lin" and through trnnsistor "b" present an extremely high impedance. This third, high impedance state is created by an enable (or disable) input to the function.

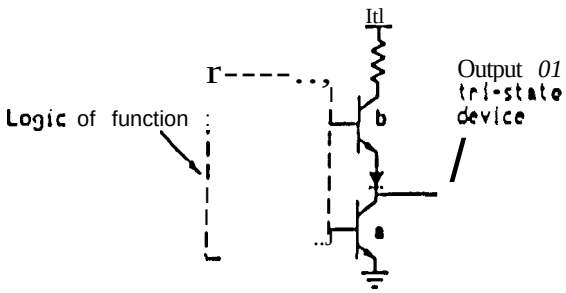


FIGURE A.4
THE TRI-STATE CONCEPT

TIle TTL 74LS245. 8 bit tri-state buffer was used in the DSP board design.

A.5) Tns PHASE LocKED Loop

TIle phase-locked loop is a unique and versatile feedback system that provides frequency selective tuning and filtering without the need for coils or inductors. It consists of three basic functional blocks; *phase comparator. low-pass filter* and *vouage-controlled oscillator* (VeO). interconnected as shown in figure A.5. With no input signal applied to the system. the error voltage. Vd' is equal to zero, 111e VCO operates at a set "free-running"

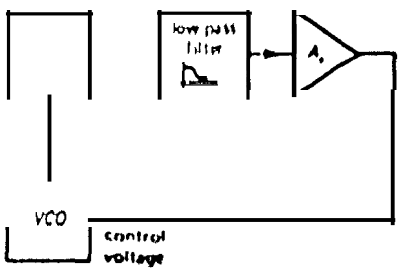


FIGURE A.5
BLOCK DIAGRAM OF ru,

frequency, f_0' If an input signal is applied to the system, the phase comparator compares the phase and frequency of the input signal with the VCO frequency and generates an error voltage. Ve(t), that is related to the phase and frequency difference between the two signals. This error voltage is then filtered and applied to the control terminal of the VCO. If the input signal frequency is sufficiently close to f_0' the feedback causes the VCO to synchronize or "lock" with the incoming signal. Once in lock. the VCO frequency is identical to the input signal. except for a finite phase difference.

Two key parameters of a phase-locked loop system is its "lock" and "capture" ranges, These can be defined as follows:

LOCK RANGE:
THE band of frequencies in the vicinity of f_0 over which the PLL can *maintain* lock with an input signal. It is also known as the "tracking" or "holding" range, Lock range increases as the overall loop gain of the PLL is increased.

CAPTURE RANGE:
THE band of frequencies in the vicinity of f_0 where the PLL can *establish* or *acquire* lock with an input signal. It is also known as the "acquisition" range. The capture range is always smaller or equal to the lock range. It is related to the low-pass filter bandwidth and decreases as the low-pass filter time constant increase.

THE PLL responds only to those input signals sufficiently close to the VCO frequency. f_0' to fall within the "lock" or "capture" range of the system. Its performance characteristics. therefore, offer a high degree of frequency selectivity. with the selectivity characteristics centered about f_0'

A.S.I) THE PHASE-LOCKED LOOP AS FREQUENCY MULTIPLIER

It is more than often in telecommunication circuits necessary to multiply a given operating clock with a certain integer. Genemting a fixed multiple of an input frequency is one of the most common applications of PLL's [44]. This is done in frequency synthesizers, where an integer multiple n of a stable low-frequency reference signal is generated as an output; n is settable digitally. giving a flexible signal source that can be controlled by computer. In more mundane applications. a PLL might be used to generate a clock frequency locked to some reference frequency already available. An example of this would be the generation of the rate R = 2/3. (d, k) = (1. 7) code, discussed elsewhere in this thesis. To maintain

synchronization and no phase error with the input clock, the input frequency (1200 liz, say) must be multiplied by two (2400 liz) and divided by 3 (800 liz), in order to achieve correct encoding and decoding.

Figure A,6 illustrates the standard PLL scheme, with a divide-by-a counter added between the veo output and the phase detector. In this diagram thc units of gain for each function in the loop is indicated as this will be important in stability calculations. Note particularly that the phase detector converts phase to voltage and thot the veo converts voltage to the time derivative of phase, frequency.

This has the important consequence that the veo is actually an integrator; a fixed input voltage error produces a linearly rising phase error at the VCO output. TIle low-pass filter

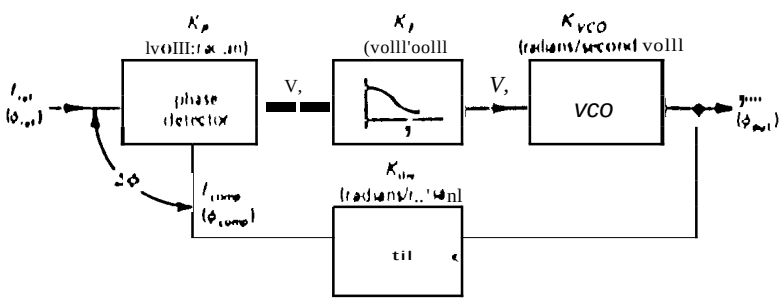


FIGURE A.6

FREQUENCY MULTIPLIER BLOCK DIAGRAM

and the divide-by-a counter both have unitlcss gain.

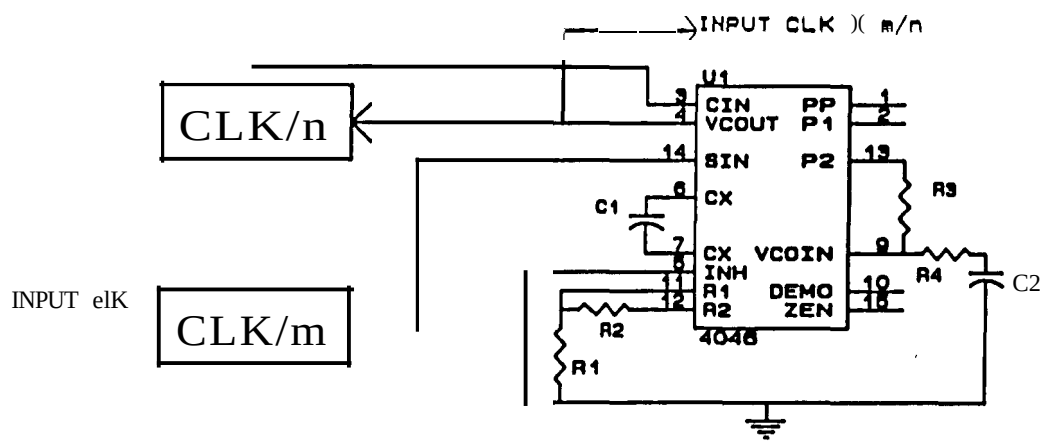
Returning to the rate R $= mln = 2/3$, $(d, k) = 0$, 7) code, a circuit had to be developed to generate a clock at two-thirds the rate of the operating frequency, and still be in phose, to ensure accurate coding of the data. The complete coder circuit diagram is presented later in this chapter.

The infamous 4046 CMOS PLL was used, since both the phase detector and VCO are incorporated in one chip. The edge-triggered type of phase detector is used in this circuit (the 4046 actually contains both kinds 145]). Implcmenting a fILI .., the first step would be to set the veo's lock range and capture range. The VCO allows us to set the minimum and maximum frequencies corresponding to control voltages ofzero and VOD' respectively, by choosing R_I, R_2 and C. according to the design graphs in the dala sheets (45). (mu is controlled by R1-C. and f_{min} is determined by C_1 and the series combination R.-R_2• It

should be noted that, by suitable choice of R. and R_2 values, the restricted-range yeO can be made to "span" any range from 1:1 to ncar-infinity.

Since the operating frequency ranged between 1200 llz and 4800 liz (standard modem rates) there was decided on f_{nun} = 100 and ($_{max}$ = 10 kllz, just to be on the safe side. The component values (or these frequencies is shown in figure A.7.

Ilaving rigged up the veo range, the remaining task is the low-pass filter design. This part is crucial. Since a first-order loop do not have n narrow bandwidth and good tracking capabilities, a second-order loop must be implemented. The trick to a stable *second-order*



FIOURE A.7

PLL MULTIPLIER

phase-locked loop is shown in the Bode plots of loop gain in Figure A.8. The yeO acts as an integrator, with 11/ response and 90° lagging phase shift. In order to have a respectable phase margin, the low-pass filter has an additional resistor in series with the capacitor to stop the rolloff at some frequency. The combination of these two responses produces the loop gain shown. As long as the loop gain rolls of(at 6dll/octave in the neighborhood of unity loop gain, the loop will be stable. The "lend-lag" low-pass filter does the trick, if the properties is chosen correctly.

Begin hy writing down the loop gain, as in table A.l, consldcring each component (refer to Figure A.6). Take special care to keep the units consistent. The only gain term still to be decided is K_F. It is done by writing down the overall loop gllin, remembering that the veo is an integrntor:

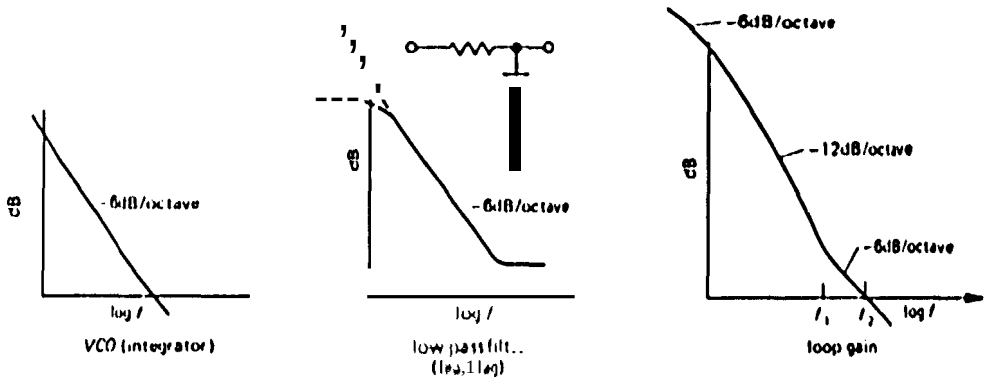


FIGURE A.8
BODE PLOTS OF LOOP GAIN

$$\phi_{OUT} = \int V_2 K_{VCO} dt \tag{A.1}$$

Component	Function	Gain	Gain Calculation
Phase detector	$V_i = K_p \nabla\phi$	K_p	Oto V_{DD} ⇔ 0^0 to 360^0
Low-Pass filter	$V2 = K_F V1$	K_F	$\frac{1+j\,\omega R_4 C_2}{1+jw(R3\,C2+R4C2)}$
VCO	$\frac{d\phi_{OUT}}{dt}$	$K_{y\,CO}$	100Hz ($V2 = 0$) 10kHz ($V_2 = 5V$)
Divide-by-a	ϕ_{COMP}	K_{ory}	$K_{DIV} = 1/n$

TABLE A.1

PLL GAIN CALCULATIONS

Other relations of interest:

$$K_p = V_i/(\Delta\phi\ 180/\pi)\ rad/s \tag{A.2}$$

$$K_F = \frac{1+j\,\omega R_4 C_2}{1+jw(R3\,C2+R_4C2)} \tag{A.3}$$

$$K_{VCO} = \Delta\phi\ 2\pi/V2 \tag{A.4}$$

$$K_{dlv} = \frac{1}{n} \tag{A.5}$$

The loop gain is given by:

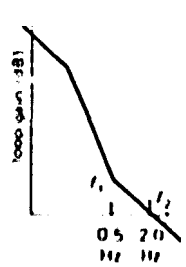
$$\text{Loop gain} = K_p K_p \frac{K_{veo}}{JW} K_{DIV} \tag{A.6}$$

Now comes the choice of frequency at which the loop gain should pass through unity. The idea is to pick a unity-gain frequency high enough so that the loop can follow the input frequency variations desired, but low enough to provide flywheel action to smooth over noise and jumps in the input frequency. A loop such as this one, designed to generate a fixed multiple of a stable and slowly varying input frequency, should have a low unity-gain frequency. That will reduce phase noise at the output and make the PLL insensitive to noise nnd glitches on the input. It will hardly even notice a short dropout of input signal, because the voltage held on the filter capacitor will instruct the VEO to continue producing the same output frequency.

In this case, the unity-gain frequency, f_2, is chosen to be 300 liz, or 1885 radians/second. This is well below the reference frequency (minimum of 1200 JIz), As a rule of thumb, the breakpoint of the low-pass filter's zero should be lower by a factor of 2 to 5, for comfortable phase margin. Remembering that the phase shift of a simple RC goes from 0° to 90° over a frequency range of roughly 0.1 to]0 times the -3dO frequency, with 8 45° phase shift at the -3d8 frequency, In this case the frequency of the zero, f_1, is chosen to be at 150 Hz or 942.5 radians/second (Figure A.9).

The breakpoint fl determines the time constant R_4C_2:

$$R_4C_2 = 1/2\pi f_1 \tag{A.7}$$



FIOURR A.9
LOOrGAIN

Tentatively, take C_2 = 100 nF and R_4 = 10k. Choose R_J so that the *magnitude* of the loop gain equals 1 at f_2, using equation A.B. In this case R_J = lOOk.

Kp	= 0.796 volts/rad
K_{vco}	= 362.2 x 10^3 rod/s·v
K_{div}	= 1/3
f_t	= 150 lIz
f_2	= 300 lIz
R_t	= 100 k
R_2	= 10 M
R_3	= 100 k
R_4	= 10 k
C_t	= InO
C_2	= lOOn

TABLE A.2
CALCULATED VALUrS

$$\frac{\omega^2}{K_p^2\ K_{VO}^2\ K_{DIV}^2} = \frac{1 + \omega^2\ R4^2\ C2^2}{1 + \omega^2\ C2^2(R3+R4)} \qquad \text{(A.B)}$$

A summary of the calculated values is presented in table A.2.

A.6) DESIGN DESCRIPTIONS

Three designs will be discussed on a circuit diagram level in this section; the TMS 320CI0 based DSP finite-state machine Clint, the 8254 timer card lind the mobile coders.

A.6.1) DSP FINrrE-STATE MACHINE GENERATOR

This circuit is based on the TMS320CI0 digital signal processor, which forms the heart of the system. As mentioned. the processor is mounted on n PC bonn! which cen plug into any existing 10M PC or compatible. Since a PC bus cen have many cards attached to it, it must be driven carefully and the interface card must decode its own address, which can

be achieved by using a digital comparator or PLD. Mnny IIlM PC clones run with a faster clock than n true blue IBM PC, so a card design that works welJ on a standard 4.71 MHz IBM PC might not run at all on a 10- or 12 MIIZ PC. The best, and perhaps the only solution to this problem. is to usc high speed chips wherever possible. Since the TMS DSP chip runs at 25 MHz, and all chips hnd to be compatible for that speed, the TMS DSP cnrd will be able to run on almost any PC, including an IBM AT or compatible.

The TMS require a memory access time of at least lOOns. The static memory used (6116-55) had nn access time of 55n8, which wns 8 bits wide and 2K deep. The TMS. however, needed 4K x 16 bit memory. because of it's 16 bit wide address bus. lienee, there were four 6116-55 memory chips necessary. Figure A.I0 shows the four memory chips *(VO·V3)* with two bidirectional tri-state buffers (U4 and US) which isolate the PC and TMS address busses and memory write signal. Two other bidirectional tri-state buffers (U6 and U7), which isolate the data busses of the two processors are also included in this figure,

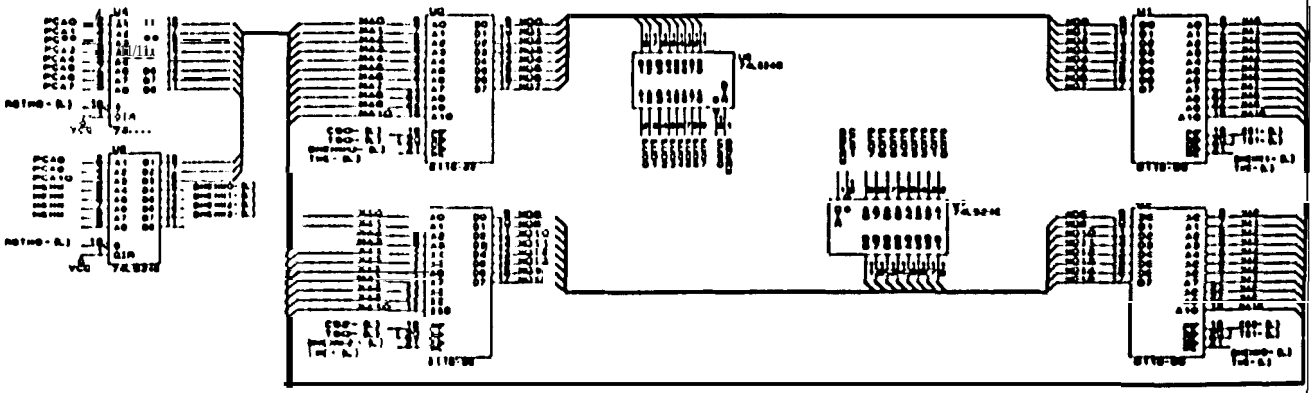


FIGURE AIO
MF.MORY ELEMENTS

Decoding of the PC address bus was done using a GAL20V8. A meaningful comment at this stage would be that the DSP FSM card's chip count was reduced from 31 to 20 by the usc of PLOts! The GAL (UB) generates the chip select pulses (SO·S3) for the four memory elements, the control signal to start the TMS (RSTMS) and two control signals (CCSO and CCSn for the memory datn bus. The signal S4 decodes an address to latch the state of RSTMS high or low via one of the PC data bits, DO. TIle PC memory mapping is as follows:

S0	CCOO:OOOO • CCOO:07FF
S1	CCOO:0800 • CC00:0FFF
S2	CDOO:OOOO - CDOO:07FF
S3	CDOO:0800 - CD00:0FFF

'111e TMS reads the memory in the following fashion (TSO and TS1 ore the TMS chip selects):

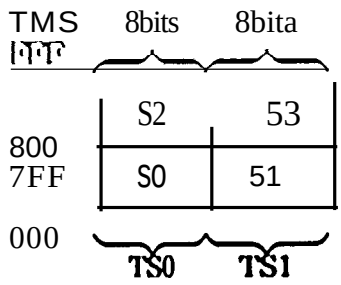


Figure A.11 shows the GAL used for the PC decoding (U8) together with U9, a bidirectional tri-state buffer which isolate the PC and TMS chip selects. The GAL programs can be found in Appendix F.

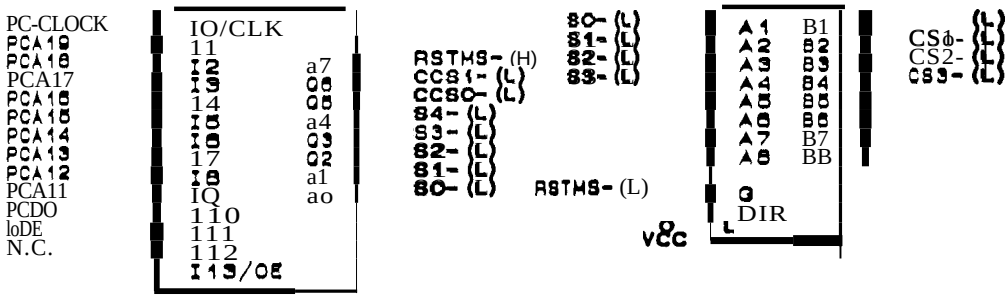


FIGURE A.11

PC ADDRESS Dp.coDiNO

The only part of the design left to discuss, is the decoding and I/O port design of the TMS. Decoding in the TMS sense; generating chip selects to select two or the four memory elements which have to be rend. A GAL was again used for this purpose (UIO) and to generate a pulse (POE) for selecting the I/O ports. A bidireetional tri-state huffer was used (UI J) to buffer Ihe TMS chip selects and memory write signals. 'This IWO chips can be seen in figure A.12.

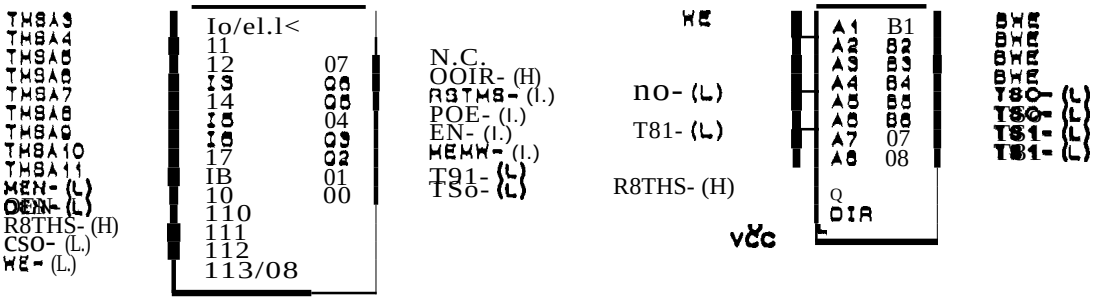


FIGURE A.12

MEMORY DECODING FOR THE TMS 32OC10

Figure A.13 shows the TMS 320CIO digital signal processor (U12) with four tri-state buffer chips (U13 • U16) used to respectively buffer the TMS address bus and data bus from the PC address and data busses. Figure A.14 shows a 3-to-8 decoder, with address lines AO to A2 as inputs, to select the I/O port buffers (UJ8 and UI9). For more infonnation concerning the TMS 320CIO digital signal processor, please consult the TMS 320CI0 user's guide [46]. A composed circuit diagram (figure A.lS) for the TMS DSP board is presented at the end of this appendix, while a program written in Turbo C to transfer the precompiled TMS program memory (rom the PC to the DSP card is presented in Appendix G.

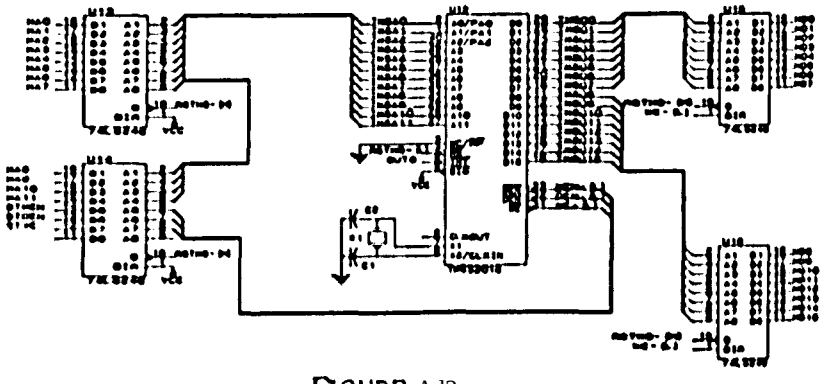


FIGURE A.l3

TMS 32OCIO DIGITAL SIGNAl. PROCESSOR

A.6.2) TIMRR CARD

Decoding of the PC address bus was again accomplished with a OA1_ ntis time a GAL 16V8 was used since fewer input pins were necessary 140). The timer card design was relatively simple; only 7 chips were needed (again the GAL must be given credit for the simplification).

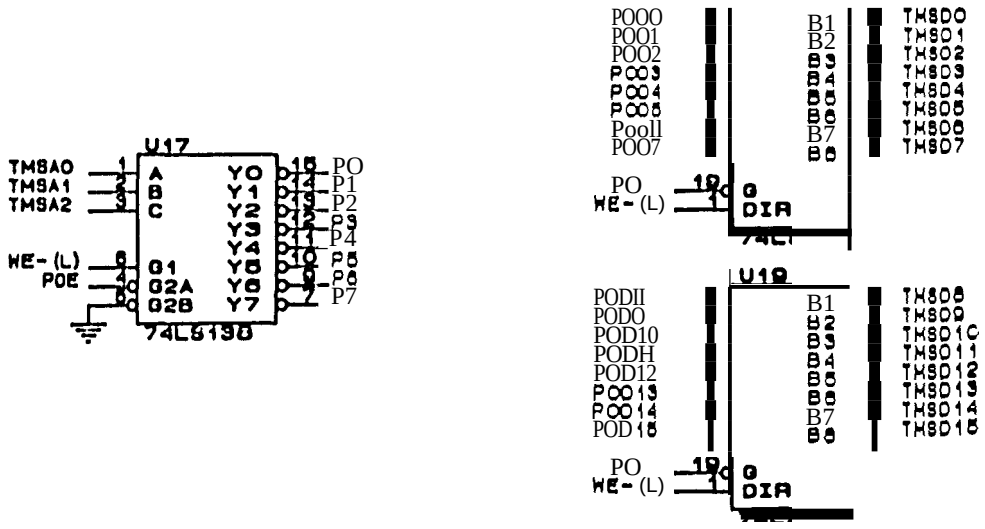


FIGURE A.14
TMS 32OCI0 I/OPoRTS

TIle GAL (US) generates two chip select pulses (CSO and CSJ), respectively for the 8254 PEC *(V7)* and the 8255 PPJ (U6). TIle jumper JPl can bet set to position 1 or 2, as indicated in the circuit diagram; by selecting position I, CSO and CSI will, respectively, decode I/O port addresses 0,,380 - 0,,383 and 0,,384 • 0,,387. With the jumper in position 2, CSO and CSl will, respectively, decode I/O port addresses 0,,390 • 0,,393 and 0,,394 - 0,,397.

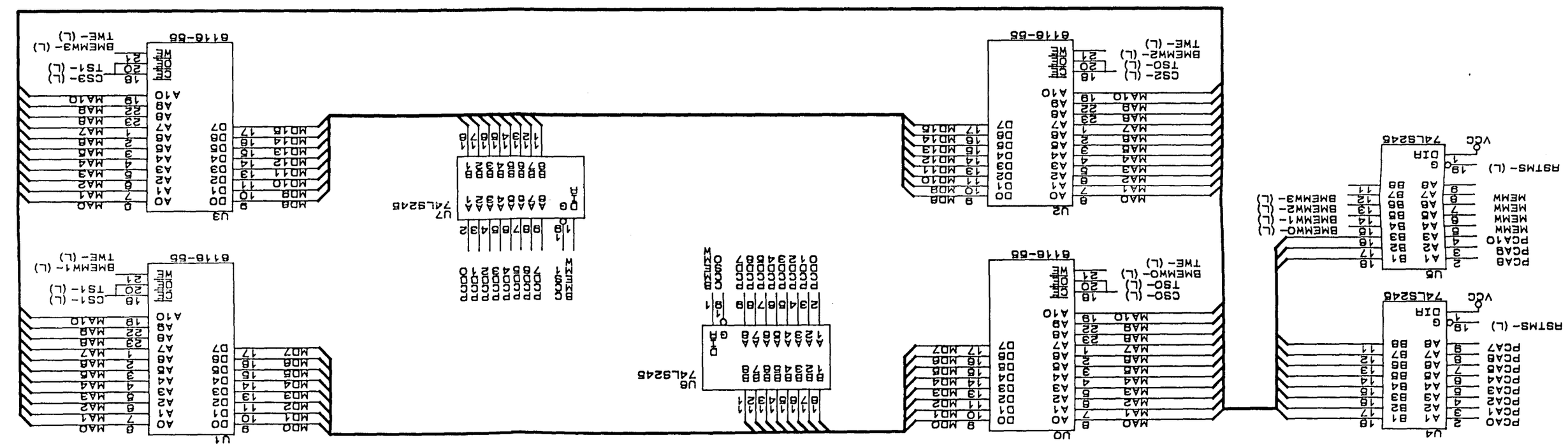
The control signal CCS is used to control the bidirectional data buffer UI, while the other two buffers (U2 and U3) are used to buffer the address bus and other control signals from the PC bus. The PLL (U4) is used as frequency multiplier as outlined in section A.S, with component values as indicated in table A.2.

11lis design made it possible to decode modulation codes via the 8255 and to generate frequency multiples of the given modem clock. Figure A.16 shows the complete circuit diagram of the timer card.

A.6.3) MOBILE CODERS

111C mobile coders basically consisted of the Intel 8751 (VI) with D multiply by *min* circuit (consisting of two 4018 frequency deviders, U2 and U3, and a 4046 PLL, U4), RS 232 • TTL converter (US) and a TTL . RS 232 converter (U6), The converter was necessary to

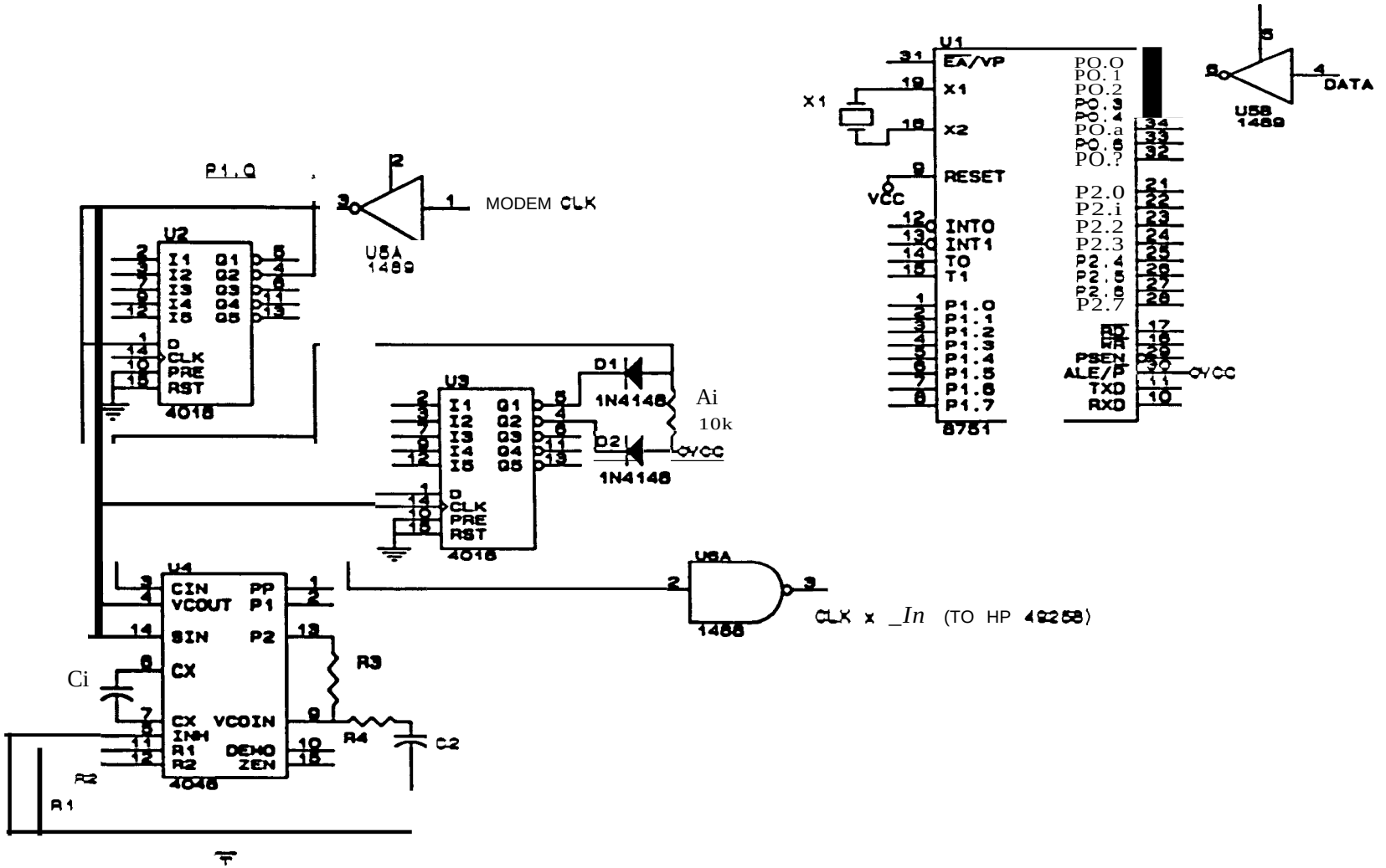
interface the coders with the modems; the modems operated at RS 232 levels. TIle diodes, 01 and D2, and the resistor, RI, function as a logicnl AND function. As already mentioned the mobile encoder nnd decoder hnd the same circuit diagram, hence, figure A.I? shows the circuit diagram for only one of the designs.



INTERFACE BETWEEN PC AND TMS32010

Size	Document Number	Rev
C	FIGURE A.15 172.A	

Date: May 1, 1991 Sheet 91



CIRCUIT DIAGRAM "DR MOBILE CODER

Size A | Document Number FIGURE A.17 | /72 C

Date:

APPENDIXB

PROGRAMS FORDSP FSMAND PC DECODER

This appendix contains two sample program listings; a program for the DSP finite-state machine encoder, written in TMS assembler, and the other for the PC based decoder, written in Turbo C. These programs realized a rate R = 1/2. (d, k) = (I, 3) code.

Other programs written in a similar fashion are presented on a floppy disk at the back of this thesis. The filenames which contain these program listings have the following legend: the letters before the file extension indicate the (d, k) or (d, k, C) parameters. The file with an .ASM extension indicate the TMS assembler encoder programs and the .C extension indicate the PC decoder programs. For instance, a file with the name 1_3.ASM contains the program listing for the (d, k) = (I, 3) encoder, written in TMS assembler,

```
●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●
•           PROGRAM WRITTEN IN TMS ASSEMnLER TO GENERATE A.
•                          RATE R =1/2. (d, k) = 0, 3)•
●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●

          IOT       '(d, k) =0. 3)'
OSTATE    EQU       O
DCODEO    EQU       1
DCODEI    EQU       2
DDTA      EQU       3
          AORG      >0
          n         BEGIN
          AORG      >A

• INITIALIZING MEMORY ADDRESSES WITH CODE INFORMATION.

PSTATE    DATA      >0
PCODEO    DATA      >0
PCODEI    DATA      >1
          AORG      >100

• PROGRAM START.

nEGIN     DINT
          LACK      PSTATE
          TnLR      DSTATE
          LACK      PCODEO
          TBLR      DCODEO
          LACK      PCODEt
          TnLR      DCODEI
          E1NT
GCODE     IN        DDTA.a
          ZALS      DSTAm
          XOR       DCODEO
          nz        STATEO
          nNZ       STATEt
STATEO    ZALS      DOTA
```

```
        XOR     DCODEO
        nz      TRANO
        nNZ     STAY0
TRANO   OUT     DCODEO,t
        OUT     DCODEO,t
        LACK    PCODEt
        TfiLR   DSTATE
        n       GCODE
STAYO   OUT     DCODEO,t
        OUT     DCODEl,t
        n       GCODE
STATEl  ZALS    DDTA
        XOR     DCODEO
        nz      STAY1
        DNZ     TRANt
STAY)   OUT     DCODEl,t
        OUT     DCODEO,t
        n       GCODE
TRANt   OUT     DCODEO,l
        OUT     DCODEl,O
        LACK    PCODEO
        TBLR    DSTATE
        n       GCODE
        END
```

• TMS ASSEMBLER PROGRAM END •

```
/*****************************************************************t
/*                    Program writtcn in Turbo C to decode a      .,
/*                      rate R=1/2, (d, k) = (I, 3) code          .,
/*****************************************************************,

#includc<stdio.h>
#includc<stdlib.h>
#inclmlc<dos.h>

mninO

   int cbitt =0, cbit2 =0, ddtn =0;
   int z =0, x;

   clrscrf):

   olltportb( Ox30f, 0:<89 ); /* initialize 8255 (J2) .,
   outportbl Ox30b, Ox9b ); /* initialize 8255 (J3) .,

   outportb( Ox313, Ox16 ); /* initialize 8254 .,
   outportb( Ox310, Ox02); /* devidc external elk by 2.'

   outportb( Ox3Oc, Ox01 ); /* port A of 8255 .,

   puts( "Decoding MILLER code...." );

   disablef): /* disable intCITUpts.'
   whilc( z = 0)
   (
       if( (x =inportb( 0:<3Oc )) = 1 )
       (
           outportb( 0x30d, ddta ): /* decoded dala bit 01 port n of J2 .,
           cbitl = inportb( 0,,308); /* first coded bit It port A of J3 .,
           outportbl Ox3Oc, Ox02); /. initialize (or next elk .,
       )
       if( (inporth( 0:<3Oc )) = 2 )
       (
```

```
        cbit2 = Inportbf Ox308);  /* second coded bit at port A of JJ .,
        ddta = cbit 1 ^ cbit2;  /* decode received bits .,
        outportbt OxJOc. 0x0l );  /* initialize (or next elk .,


)
/* End o( decoding program .,
```

APPENDIXC

PROGRAMS FOR MOBILE EXPERIMENTS

Again, this appendix contains two sample program listings: realizing a rate R =1/2. (d, k) = 0,3) code for the mobilc flnhe-state machine encoder and the 8751 based decoder, both written in PLM/51.

Other programs written in a similar fashion are presented on a floppy disk at the back of this thesis, with filenames having the following legend: the first three letters indicate if the program is used for encoding (ENe) or decoding (DEC). TIle next letters indicate the (d, k) or $(d, k;$ C) code parameters. with file extension .PI.M. As an example consider the filename ENCl_3.PLM; a (d, k) =0,3) encoder written in PLM/SJ.

TIle programs are self explanatory with meaningful comments where necessary and will thus not be further descrilxed. This programs were used to obtain the results as outlined in chapter 9 and Appendix D.

```
/*****************************************************************/
                Program written in PLM/51 to realize a          ./
                rate R = 1/2 (n, k) = (1. 3) encoder            */
/*****************************************************************/

sian:          do;
$inc\udc(reg51. inc)

  declare dec Iitemlly 'declare';

/*****************************************************************/
  dec plO bitot (90h) reg; /* Port address declerations ./
  dec p11 bitat (91 h) reg;
  dec pl2 bit at (9211) reg;
  dec p13 bitat (93h) reg;
  dec pl4 bitat (94h) reg;
  dec pIS bitat (95h) reg;
  dec mcon byte at (Oc6h) reg;

/* initializing and declerntion of variables .,

  dec (tmpcd0, tmpcd1. pndta, code. estate) byte;

  dec (tcmpbt. temp. cdbitO. cdbitl) bit;

  dec dala (4) byte main;
  dec nstate (4) byte moin;

  data(O) = 2;
  dala(I) = 0;
  data(2) = I;
  dala(3) = I:

  nstate(O) = 0;
  nstatc(l) = 0;
  nsla tc(2) = I;
  nstatc(3) = I:
```

```
/*****************************************************************,
 temp=I;
 cdbitO =0;
 cdbltl =I;
 ie = 0h;
 peon =011;
 meon =Oil;
 estate =0;
,*****************************************************************,

 do while temp = 1;
  if pII =1 then /* loop waiting for n positive edged clock .,
  do;
    pl4 = cdbitO; /* reading first data bit .,
    ternpbt =edbitO;
    plO = 0;
    pI2=1;
  end;

  if pl3 =I then
  do;
    pl4 = cdbitl; /* reading second data bit .,

    if pIS = 0 then
  pndta = 0;
    else
  pndta = I;

    code =datn(cstntc + pndtn·2);
    trnpcdO =I nnd code;
    if tmpcdO = 0 then
  cdbitO =0;
    else
  cdbitO =I;

    code =shr(code. I);
```

```
      tmpcd1 =1 nnd code;
      if tmpcdl = 0 then
    cdbitl = 0;
      else
    cdbitl = 1;

      edhitl =cdbitl xor tempbt;
      edbitO =cdbitO xor edbitl; /* encoding darn hits .,
      estate =nstatelcstate + pndla.2);
      p12 = 0;
      pl0 = 1;
    end;
  end;

/*****************************************************************,
end start;
/* end of encoding .,
```

```
/*                                                                    */
/*                    Program writtcn in PLM/51 to realize a          */
/*                    rille R = 112. (d, k) =0, 3) decoder            */
/*                                                                    */

start:          do:
$include(reg51.inc)


 declare dec literally 'declare';

/*                                                                    */
/* decleration and initialization of variables */
 dec plO bit lit (90h) reg:
 dec P11 bit lit (91 h) reg:
 dec p34 bit lit (01)411) reg;
 dec p35 bit at (Ob5h) reg;
 dec p20 bit at (00011) reg;
 dec p21 bit at (On 1h) reg;

 dec (temp, cdbittl, cdbit I, dta) bit;

/*                                                                    */
 temp =1:
 dta =0;
 ie =0;

/*                                                                    */
 do while temp =I: /* decoding loop */
  if p34 =I then
  do;
   p21 =dtn;
   cdbitl =p20: /* reading first channel bit */
   plO =0;
   p11 =1;
  end;
```

```
  if p35 =1 then
  do;
    cdbitO =p20: /* reading second channel bit .,

    din =cdbilO xor cdhitl; ,. decoded data hit .,
    pll =0;
    pl0 =I;
  end;

 end;

/*****************************************************************,
end start;
,. End of decoding .,
```

APPENDIXD

ERROR DISTRIBUTION RESULTS

Tables depicting $P(\mu, v)$, $v = 7$, 15 and 31, together with graphs showing the burst distribution, hurst interval distribution, cluster distribution and error free run for the five modulation codes, chosen for observation on mobile radio channels, in city and highway environments for both modulation schemes nrc presented in this appendix. A thorough description of these results can be found in chapter 9.

II	P(μ, 31)	P(μ, 15)	P(μ, 7)
0	0.978391	0.986730	0.991035
I	0.004299	0.003010	0.002571
2	0.003·185	0.002404	0.002256
3	0.002819	0.001930	0.001899
·1	0.002276	0.001568	0.00133·1
5	0.001833	0.001285	0.000672
(.	(I.U() J47j	U.110 1Il.1J	O.O()020.3
7	O.U()IIRI	U.(J(I()IIUI)	0.OOOO2()
8	0.11009.15	1I.UO057.3	.
9	0.000756	0.UO0.15·1	.
10	0.1100(,05	O.()(10182	.
II	0.1100·18-1	O.()}(1075	.
12	O.OOO38f,	O.()()()23	.
1.1	0.000306	1I.0I)()()O5	.
I,'	0.000238	0.0()0U00	.
15	0.000179	0.000000	.
16	0.000129		.
17	0.000088		.
18	0.000056	.	.
19	enooora	.	.
20	0.000017		.
21	0.000008	.	.
22	0.000003		.
2.1	().00()00 I		.
24	0.000000		.
25	0.000000	.	.
26	0.000000	.	.
27	0.000000	.	.
28	0.000000	.	.
29	S.ns-u	.	.
30	3.9E-12	–	.
31	1.2E·1J	.	.

TABLE D.1

PROBABILITIES P(μ, ν): (d, k, C) =(0, 2, 1): 4-ARY PSK IN CITY

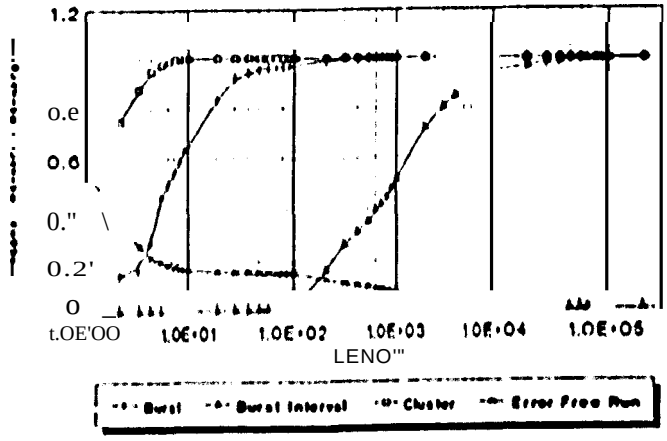


FIGURE m

ERROR DISTRIBUTION: (d, k, C) =(0. 2, I): 4-ARY PSK IN CITY

l'	$P(\mu, 31)$	$P(\mu, IS)$	$P(\mu, 7)$
0	0.980616	0.987673	0.991519
I	0.003367	0.002560	0.002196
2	0.002840	0.0021()'1	0.002027
3	0.002387	0.001727	0.001881
4	0.001999	0.001428	0.001419
5	0.001667	O,llOI203	0.000720
6	0.001385	0.001028	0.000208
7	0.001145	0.000856	0.000025
8	0.000943	0.000650	.
9	o.ooonJ	0.000421	.
10	0.000632	0.000221	.
11	0.000515	O.00llOH9	.
12	0.000420	0.000026	.
13	0.0003·12	0.00llOO5	.
1·1	0.llOO2n	0.llOllOOO	.
15	0.000219	0.000ll00	.
16	0.000167	.	.
17	0.000120		.
18	0.000080		.
19	0.000048		.
20	0.000026	.	.
21	0.000012		–
22	0.000005		–
23	0.000001	.	.
2·1	0.000000	.	.
25	0.000000	.	.
26	0.000000	.	.
27	0.00000o	.	.
28	0.000000	.	.
29	6.2E-11	.	.
30	3.6E-12	.	.
31	1.0E-1J	.	.

TABLE D.2

PROBABILITIES $P(\mu, \nu)$: $(d, k, C) = (0.\ 1.\ I)$: 4·ARY PSK IN CITY

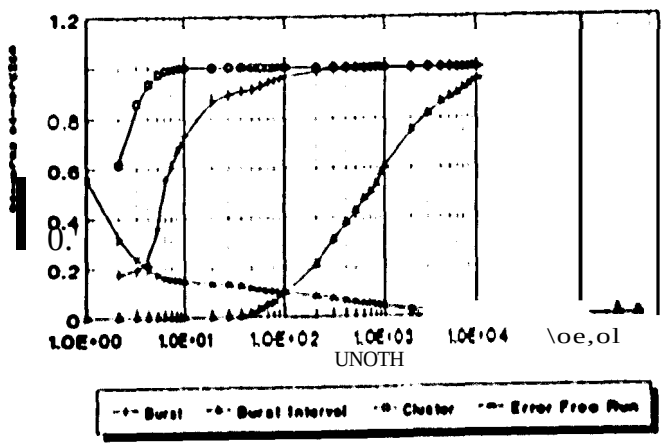


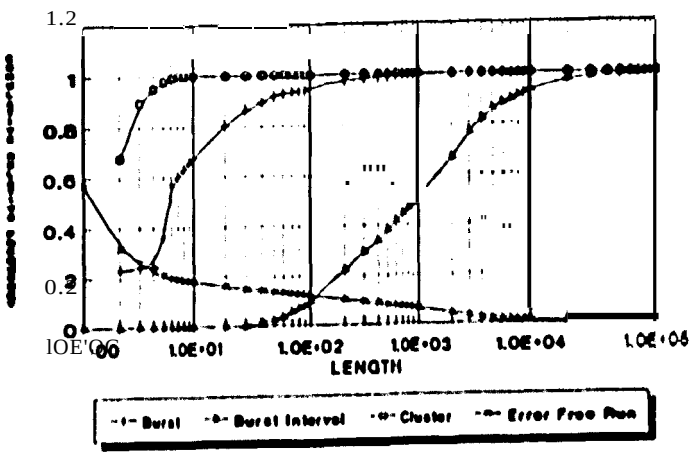
FIGURE 0.2

ERROR DISTRIBUTION: $(d, k, C) = (0.\ 1.\ J)$: 4-ARY PSK IN CITY

ν	$P(\mu, 31)$	$P(\mu, 15)$	$P(\mu, 7)$
0	0.978571	0.986707	0.991310
1	0.004303	0.0032.16	0.002606
2	0.003521	0.002549	0.002127
3	0.002866	0.001990	0.001793
4	0.002320	0.001544	0.001307
5	0.001866	0.001197	0.000652
6	0.001492	0.000932	0.000181
7	0.001185	0.000718	0.000020
8	0.000935	0.000522	.
9	0.000732	0.000332	.
10	0.000569	0.000173	.
11	0.000438	0.000069	.
12	0.0003.15	0.000020	.
13	0.000254	0.000004	.
14	0.000191	0.000000	.
15	0.000111	0.000000	.
16	0.000102	.	.
17	0.000070	.	.
18	0.000046	.	.
19	0.000027	.	.
20	0.000014	.	.
21	0.000007	.	.
22	0.000002	.	.
23	0.000001	.	.
24	0.000000	.	.
25	0.000000	.	.
26	0.000000	.	.
27	0.000000	.	.
28	0.000000	.	.
29	2.BE·II	.	.
30	1.5E·12	.	.
31	4.0E·14	.	.

TABLE D.3

PROBABILITIES $P(\mu, \nu)$: $(d, k) = (1, 3)$; 4-ARY PSK IN CITY



FIOURR D.3

ERROR DISTRmtmON: $(d, k) = (I, 3)$; 4-ARY PSK IN CITY

μ	$P(\mu, 31)$	$P(\mu, 15)$	$P(\mu, 7)$
0	0.988963	0.991542	0.992912
I	0.000900	0.000nl	0.001101
2	0.000836	0.000743	0.001594
3	0.000m	0.000767	0.001849
.1	0.000722	0.000872	0.001491
5	0.000671	0.001030	O.OOOn9
6	0.000625	0.0011·18	0.000239
7	0.000587	0.001116	0.000032
8	0.0005(,2	0.000907	.
9	0.000553	0.000600	.
10	0.000563	0.000316	.
II	0.000588	0.000129	.
12	0.0)0)(,16	0.U000·I0	.
13	0.000627	0.000008	.
1·\	0.000606	0.00llOOI	.
15	0.000542	0.000000	.
16	0.000445	.	.
17	0.000330	.	.
18	0.000221		.
19	0.000132		.
20	0.000070		.
21	0.000033	.	.
22	0.000013	.	.
23	0.000004		.
24	0.000001	.	.
25	0.00000o	.	.
26	0.00000o	.	.
27	0.00000o	.	.
28	0.00000o	.	.
29	2.IE·IO	.	.
30	J.4E·II	.	.
31	4.5E·13		.

TABLE 0.4

PROOAnJUTIES $P(\mu, \nu)$: $(d, k) = 0, 7)$; 4-ARY PSK IN CITY

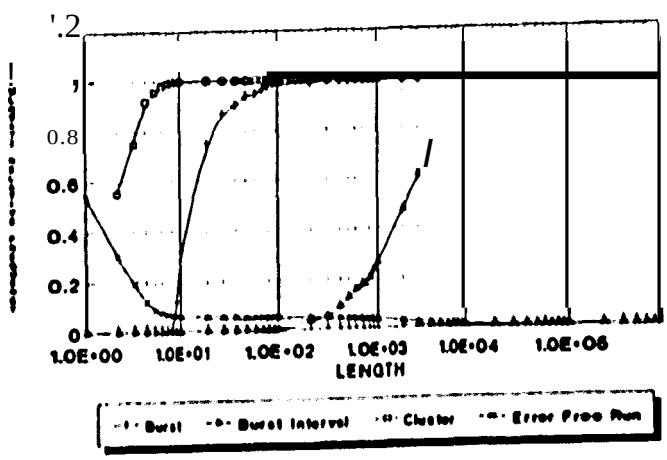


FIGURE D.4

ERROR DISTR1I1l1110N: $(d, k) = 0, 7)$; 4-ARY PSK IN em'

μ	$P(\mu, 31)$	$P(\mu, 15)$	$P(\mu, 7)$
0	0.991214	0.994681	0.996439
1	0.001754	0.001202	0.000965
2	0.001418	0.000951	0.000856
3	0.001145	0.000756	e.ooom
4	0.000922	0.000605	0.000575
5	0.000741	1I,II0ll·IIM	0.000293
6	0.000594	0.00ll·1I I	0.000086
7	0.000·175	0.00113,17	0.000010
8	0.OOOJ79	0.000253	.
9	0.000302	0.00016·1	–
10	0.0002·10	0.00()0X7	–
11	0.000191	0.OOOOJ5	.
12	0.000153	0.000010	–
13	0.000122	0.000002	.
14	0.OOOO9H	0.0n000n	.
15	uOCOO»	0.0000U0	.
16	0.000058	.	–
17	0.000042	.	.
18	0.000028	.	–
19	0.000017	.	–
20	0.000009	.	–
21	0.00000·1	.	.
22	0.000001	.	–
23	0.00e000	.	.
24	0.000000	.	–
25	0.00000o	.	.
26	0.000000	.	–
27	0.000000	.	–
28	0.00000o	.	.
29	2.7E·lJ	.	–
30	1.6E·12	.	.
31	4.8E·14	.	–

TABLE D5

PROBABILITIES $P(\mu, \nu)$: (d, k) = (2, 7): 4-ARY PSK IN CITY

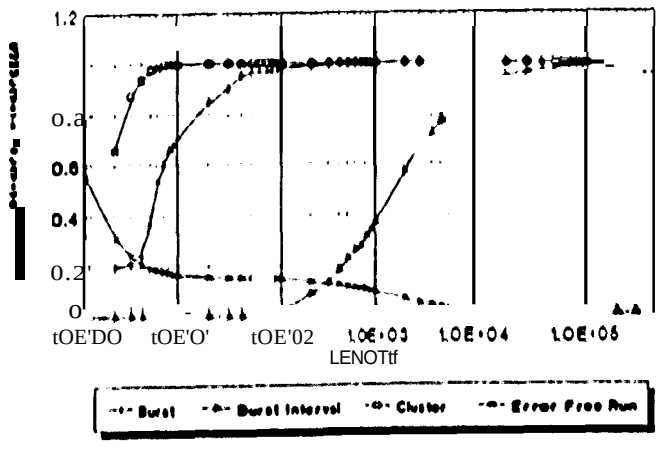


FIGURE 05

ERROR DISTRIIIU110N: (d, k) = (2, 7); 4-ARY PSK IN CITY

r	$P(\mu, 31)$	$P(\mu, 15)$	$P(\mu, 7)$
0	0.915162	0.944923	0.961818
I	0.013991	0.011210	0.009989
2	0.012019	0.009.132	0.009181
3	0.010269	0.007759	0.008388
4	0.008725	0.006490	0.006343
5	0.007373	0.005495	0.003244
6	0.006194	0.004668	0.000926
7	0.005175	0.003838	0.000108
8	0.004300	0.0(12/1/17	.
9	0.003554	0.II0I867	.
10	0.002925	0.0009/11	.
II	0.002·100	0,II00.198	.
12	0.001962	0.0II0117	.
13	0.001595	0.0()002J	.
1·1	0.001281	0,000002	.
15	0.001004	0.000000	.
16	0.000756	.	.
17	0.000537	.	.
III	0.000353	.	.
19	0.000212	.	–
20	0.000114	.	–
21	0.000054	.	.
22	0.000022	.	.
23	0.000008	.	.
2·1	0.000002	.	.
25	0.000000	.	–
26	0.000000	.	.
27	0.000000	.	–
28	0.00000o	.	–
29	2.3E·10	.	.
30	r.as-u	.	.
31	3.4E·13	.	.

TABLE 0.6

PRoOADIUTmS $P(\mu, \nu)$: $(d, k, C') = (0.2, 1)$; 4-ARY PSK IN IUOIIWAY

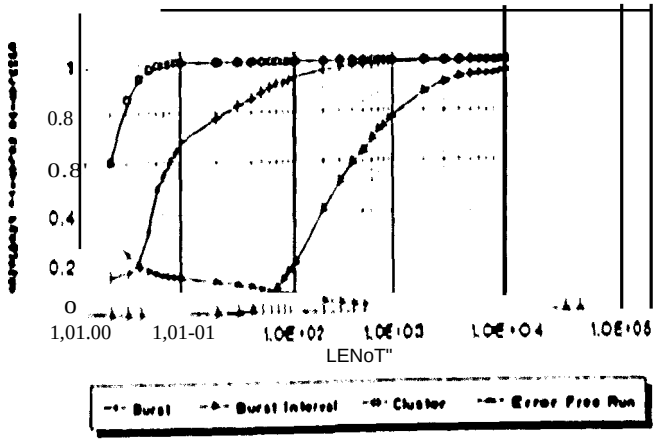


FIGURE 0.6

ERROR DISTRIBUTION: $(d, k, C) = (0.2, I)$; **4-ARY PSK** IN 1II001WAY

l'	$P(\mu, 3J)$	$P(\mu, 15)$	$P(\mu, 7)$
0	0.938364	0.955899	0.967065
I	0.007006	0.006750	0.007121
2	0.006472	0.006153	0.007547
3	0.005959	0.005601	0.oon96
4	0.005466	0.005144	0.006222
5	0.004993	0.004810	0.003213
6	0.004539	0.00415I2	0.000922
7	0.004105	0.00·1035	0.000110
8	0.003692	0.00.1209	.
9	0.003301	0.002133	.
10	0.002936	0.001128	.
11	0.002599	0.000·155	.
12	0.002288	0.00013·1	.
13	0.001997	0.000027	.
14	0.001711	0.000003	.
15	0.001420	0.000000	–
16	0.001119	.	.
17	0.000821	.	–
18	0.000552	.	.
19	0.000334	.	.
20	0.000180	.	–
21	0.000085	.	.
22	0.000035		.
23	0.000012	.	.
24	0.000003	.	–
25	0.000000	.	.
26	0.000000	.	–
27	0.00000o	.	.
28	0.00000o	.	–
29	0.000000	.	–
30	1.9E·1I	.	–
31	5.3E·13	.	.

TABLE D.7

PROBABILITIES $P(\mu, \nu)$: (d, k, C) =(0. 1.1); 4-ARY PSK ON 1II00IWAY

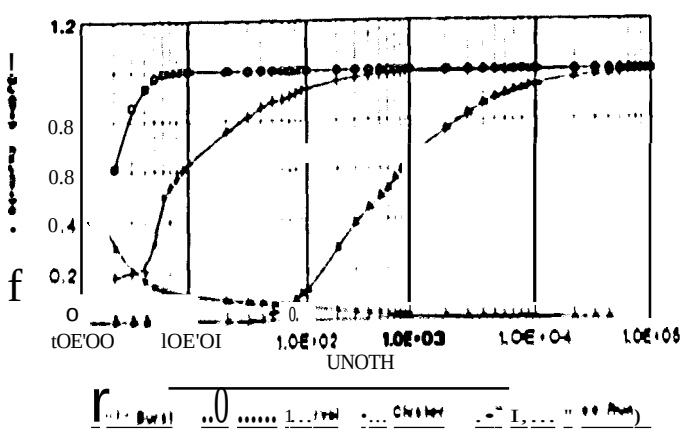


FIGURE D.7

ERROR DISTRJntmON: (d. k, C) =(0. 1,1); 4-ARY PSK ON 1Ir00IWAY

ν	P(μ, 31)	P(μ, IS)	P(μ, 7)
0	0.976544	0.984694	0.989423
I	0.003891	0.003130	0.002816
2	0.003322	0.002626	0.002537
3	0.0021BO	0.002192	0.002313
4	0.002403	0.00182·1	0.001748
5	0.002033	0.001523	0.000883
6	0.001714	0.001274	0.0002·18
7	0.001438	0.001039	0.000028
8	0.001200	0.000781	.
9	0.0()()995	0.000505	.
10	0.000820	U,IIOll2M	–
II	0.000670	O.OOUI06	–
12	0.000543	0.1)011031	.
13	0.000436	0.00000(1	–
14	0.000345	0,0()00000	.
15	0.000267	0.000000	.
16	0.000199	.	.
17	0.000140	.	–
18	0.000092	.	.
19	0.000055	.	–
20	0.000029	.	–
21	0.000014	.	–
22	0.000005	.	.
23	0.000002	.	.
24	0.000000	.	.
25	0.000000	.	.
26	0.000000	.	.
27	0.00000o	.	.
28	0.00000o	.	–
29	5.4E-11	.	–
30	3.0E·12	.	–
31	7.9E-14	.	–

TABLE 0.8

PROBABILITIES **P(μ, ν): (d, k)** =0. 3);4-ARY PSK ON 1II011WAY

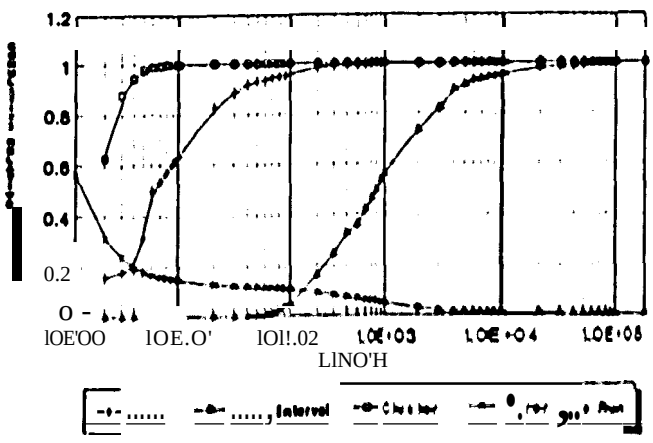


FIGURE 0.8

ERROR DISTRJDUTION: **(d, k)** =U, 3);4·ARY **PSK** ON 1II001WAY

r	$P(\mu, 11)$	$P(\mu, 15)$	$P(\mu, 7)$
0	0.960426	0.974352	0.982347
1	0.006551	0.005254	0.004594
2	0.005614	0.004374	0.004165
3	0.004719	0.003623	0.003455
4	0.001067	0.003000	0.002972
5	0.003436	0.002508	0.001550
6	0.002889	0.002119	0.000457
7	0.002115	0.001758	0.000056
8	0.002007	0.000850	.
9	0.001658	0.000126	.
10	0.001361	0.000183	.
11	0.001111	0.000201	.
12	0.000905	0.000061	.
13	0.000729	0.000012	.
14	0.000584	0.000001	.
15	0.000460	0.000000	.
16	0.000351	.	.
17	0.000254	.	.
18	0.000171	.	.
19	0.000106	.	.
20	0.000058	.	.
21	0.000028	.	.
22	0.000012	.	.
23	0.000004	–	.
24	0.000001	.	.
25	0.000000	.	.
26	0.000000	.	.
27	0.000000	.	.
28	0.000000	.	.
29	1.7E-10	–	.
30	9.8E-12	.	.
31	2.8E-13	.	.

TABLE D.9

PROBABILITIES $P(\mu, \nu)$: $(d, k) = (1, 7)$; 4-ARY PSK ON JIIOIIWAY

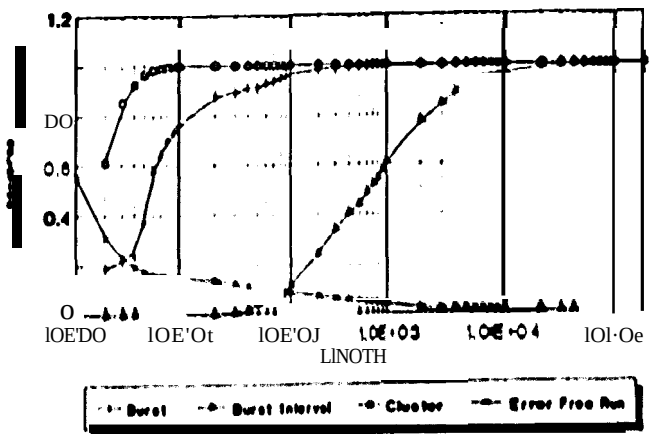


FIGURE D.9

ERROR DISTRIOlmON: $(d, k) = (J, 7)$; 4-ARY PSK ON IIJOIIWAY

μ	P(μ, 3J)	P(μ, 15)	P(μ, 7)
0	0.969705	0.979983	0.985489
I	0.004587	0.003551	0.003493
2	0.003960	0.003052	0.003503
3	0.003412	0.002663	0.003301
4	0.002933	0.0<12378	0.002485
5	0.002516	0.002157	0.001285
6	0.002155	0.001929	0.000389
7	0.001844	0.0<1I(,24	0.000050
8	0.001579	0.001222	.
9	0.001356	0.000785	.
10	0.001168	O.Oroll J	.
II	0.001010	0.000171	.
12	0.000872	0.000053	.
13	0.000745	0.000011	.
14	0.000622	0.000001	.
IS	0.00().199	0.000000	.
16	0.000379	.	.
17	0.000269	.	.
18	0.000175	.	.
19	0.000104	.	.
20	0.000055	.	.
21	0.000026	.	.
22	0.000011	.	.
23	0.00000r	.	.
24	0.000001	.	.
25	0.000000	.	.
26	0.o00ooo	.	.
27	0.o00ooo	.	.
28	0.o00ooo	.	.
29	1.8E·10	.	.
30	1.1 E·11	.	.
31	3.4E·13	.	–

TABLE 0.10

PROBABILITIES P(μ, ν): (d. k) =(2. 7); 4-ARY PSK ON IIIOnWAY

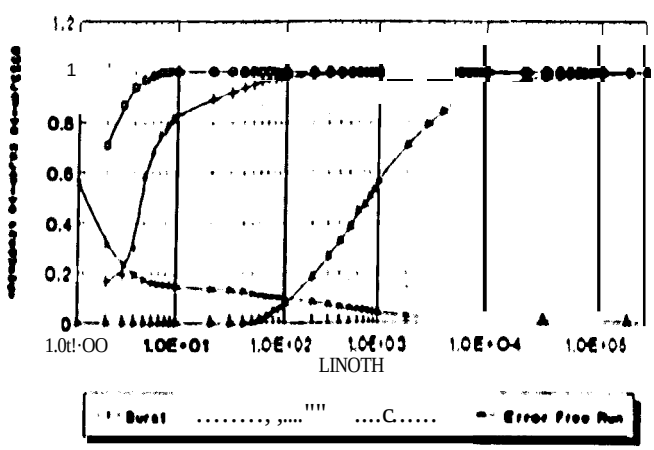


FIGURE D.lO

ERROR DfSTRmtmoN: (d, k) =(2, 7); 4·ARY **PSK** ON 1I1011WAY

	P(μ, 31)	P(μ, 15)	P(μ, 7)
0	0.992938	0.995526	0.996891
1	0.001335	0.000970	0.000906
2	0.001096	0.000796	0.000824
3	0.000899	0.000661	0.000682
4	0.000736	0.000556	0.000441
5	0.000601	0.000466	0.000196
6	0.000491	0.0003n	0.000052
7	0.000402	0.000282	0.000006
8	0.000328	0.000186	–
9	0.000269	0.000 IIJ.I	–
10	0.000220	0.000Q.17	–
11	0.000179	0.000017	–
12	0.0001·14	0.00000·1	–
13	0.000113	0.1100000	–
14	0.OOOIIR5	0.(1)00/III	–
15	0.000061	0.000000	–
16	0.000041	.	–
17	0.000025	.	–
18	0.00001·1	.	–
19	0.1100007	.	–
20	0.000003	.	–
21	0.000001	.	–
22	0.000000	.	.
23	0.o00ooo	.	–
24	0.000000	.	–
25	0.000000	.	–
26	0.o00ooo	.	–
27	0.000000	.	–
28	4.JE-11	–	-
29	3.8E·12	.	-
30	2.2E·13	.	-
31	6.m·15	.	–

TABLE 0.11

PROBABILITIES P(μ, V): (d, k, C) = (0,2. 1); 8-ARY PSK IN CITY

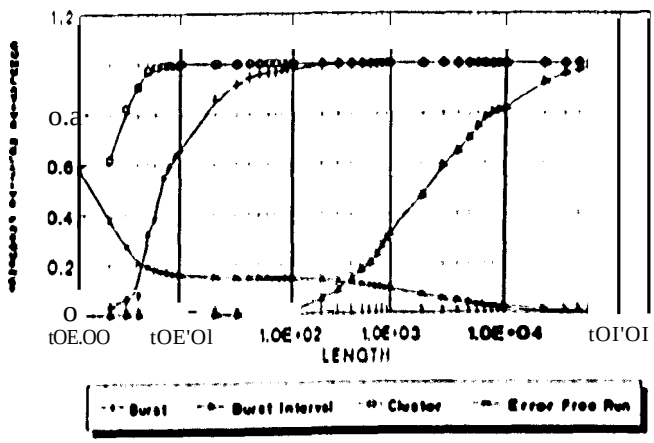


FIGURE 0.11

ERROR DISTRIDITI10N: (d, k, C) = (0, 2. 1); 8-ARV PSK IN CITY

//	P(μ, 3J)	P(μ, 15)	P(μ, 7)
0	0.982961	0.989004	0.992191
I	0.002833	0.002123	0.001988
2	0.002399	0.00171H	0.001936
3	0.002027	O.OOISIJ	0.001764
4	0.001709	0.001312	0.001272
5	0.001437	0.001158	0.000632
6	0.001207	0.001008	0.000188
7	0.001012	0.000825	0.000024
8	0.000848	0.llOO602	.
9	0.000712	0.000374	.
10	0.000599	0,111101)()	.
II	0.000506	0.OllOO76	.
12	0.000·127	0.000023	.
JJ	0.000356	0,llOOOO4	.
14	0.000290	0.000000	.
15	0.000227	0.000000	–
16	0.000168	.	.
17	0.000116	.	.
18	0.()(1007·1	.	.
19	0.0000.12	.	.
20	0.000022	.	.
21	0.000010	–	–
22	0.000004	.	.
23	0.000001	.	.
24	0.o00ooo	.	.
25	0.000000	.	.
26	0.o00ooo	.	.
27	0.o00ooo	.	.
28	0.o00ooo	.	.
29	6.3E-ll	.	.
30	4.1E·12	.	.
31	1.3E-13	.	.

TABLE 0.12

PROBABILITIES P(μ, ν): (d, k, C) = (0, 1. 1): 8-ARY PSK IN CITY

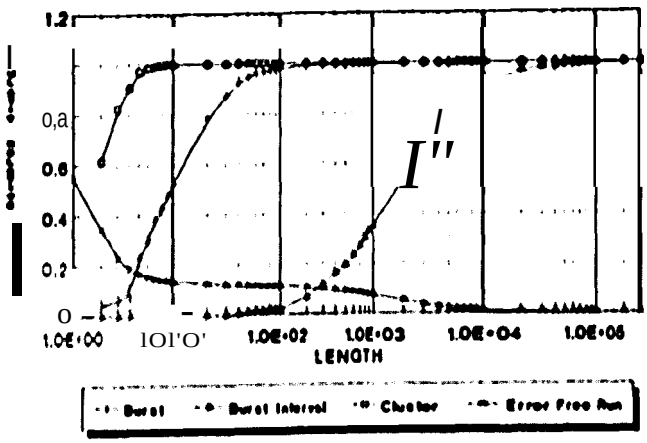


FIGURE n.12

ERROR DISTRIBUTION: (d, k, C) = (0, I. 1): 8-ARY PSK IN CITY

l'	$P(\mu, 31)$	$P(\mu, 15)$	$P(\mu, 7)$
0	0.986278	0.989747	0.991739
1	0.001176	0.001221	0.001755
2	0.001262	0.001184	0.002147
3	0.001158	0.001224	0.002059
4	0.001063	0.001018	0.001417
5	0.000980	0.001381	0.000662
6	0.000909	0.001315	0.000191
7	0.000854	0.001088	0.000026
8	0.000817	0.000761	.
9	0.000795	0.000441	.
10	0.000n9	0.000208	.
11	0.000755	0.000078	.
12	0.000708	0.000022	.
13	0.000632	0.0000{),	.
14	0.000527	0.000000	.
15	0.000107	0.000000	.
16	0.000289	.	.
17	0.000187	.	.
18	0.000110	.	.
19	0.000058	.	.
20	0.000027	.	.
21	0.000011	.	.
22	0.000004	.	.
23	0.000001	.	.
24	0.000000	.	.
25	0.000000	.	.
26	0.000000	.	.
27	0.000000	.	.
28	0.000000	.	.
29	5.4E-11	.	.
30	J.6E-12	.	.
31	1.2E-13	.	.

TABLE D.13

PROBABILITIES $P(\mu, \nu)$: (d, k) = (1,3); 8-ARY PSK IN CITY

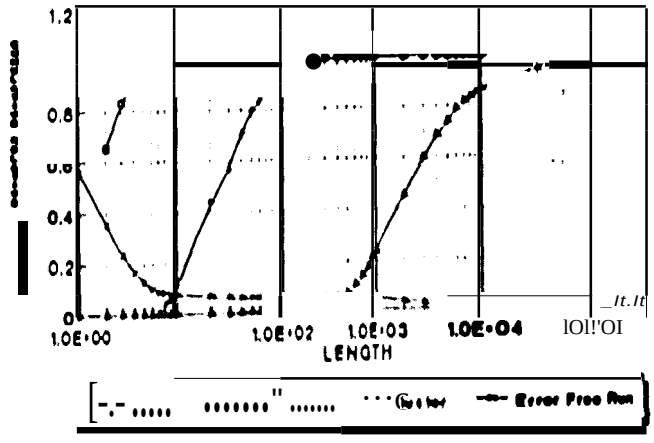


FIGURE nu

ERROR DISTRIBUTION: (d, k) = (1,3); 8-ARY PSK IN CITY

#'	P(μ, 31)	P(μ, 15)	P(μ, 7)
0	0.983961	0.989326	0.992205
1	0.002453	OJIO 1890	0.001956
2	0.002108	0.001642	0.001955
3	0.001809	0.001456	0.001783
4	0.001550	0.001318	0.001276
5	0.001328	0.001196	0.00062.1
6	0.001138	0.001050	0.000177
7	O.000978	0.000852	0.000021
8	0.000843	0.llOO611	.
9	0.000731	0.000372	.
10	0.000635	0.000184	.
Jl	O.()(10552	0.000072	.
12	0.000474	0.000021	.
IJ	0.000399	0.000004	.
1·1	0.000.124	o.0()0000	.
15	0.000250	0.000000	.
16	0.000181	.	.
17	0.000122	.	.
18	0.000075	.	.
19	0.0000·12	.	.
20	0.000021	.	–
21	0.000009	.	.
22	0.000003	.	–
23	0.000001	.	.
24	0.000000	.	.
25	0.00000o		–
26	0.000000	.	.
27	0.00000o	.	–
28	0.00000o	.	.
29	J.9E·1l	.	.
30	2.3E·12	.	.
31	6.7E-14	.	–

TABLE D.l4

PROBABILITIES P(μ, v): (d, k) = (1.7): B-ARY PSK IN CITY

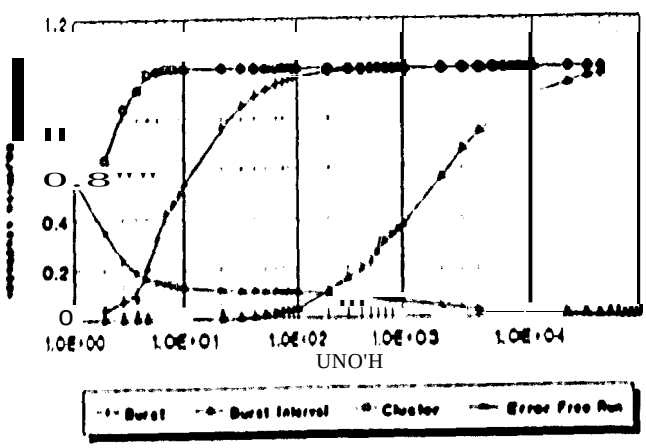


FIGURE D.l4

F..RROR DlSTRmunoN: (d, k) = (1,7); 8-ARY PSK IN CITY

l'	$P(\mu, 3J)$	$P(\mu, 15)$	$P(\mu, 7)$
0	0.985207	0.990691	0.993495
I	0.002715	0.001943	0.001751
2	0.002243	0.001589	0.001674
J	0.001850	0.001315	0.001468
4	0.001523	0.001116	0.000997
5	0.001251	0.000965	0.000464
6	0.001027	0.0/101121	0.000130
7	0.000841	0.000648	0.000016
II	0.0006119	0.000·151	.
9	0.000566	0.000264	.
10	0.000467	O.()(IOI2()	.
II	0.000387	0.000047	.
12	0.000320	0.000013	.
13	0.000262	0.000002	.
14	0.000208	0.00000o	.
15	0.000158	0.000000	.
16	0.000113	.	.
17	0.000074	.	.
18	0.000045	.	.
19	0.000024	.	.
20	0.000012	.	.
21	0.000005	.	.
22	0.000002	.	.
23	0.000000	.	.
24	0.00000o	.	.
25	0.00000o	.	.
26	0.000000	.	.
27	0.00000o	.	.
28	2.2E·10	.	.
29	2.1E·J)	.	.
30	1.3E·12	.	.
31	4.0E·14	.	.

TABLE D.IS

PRonADILmES $P(\mu, \nu)$: (d, k) = (2, 7); 8-ARY PSK IN CITY

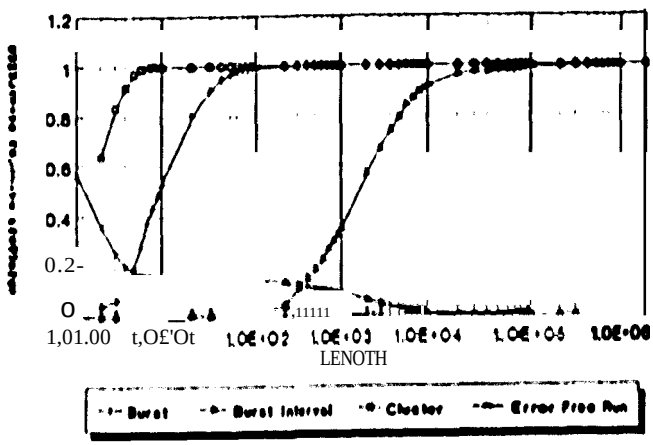


FIGURE 0.15

ERROR DISTRIBUTION: (d, k) = (21 7); 8-ARV PSK IN CITY

//	P(μ, 31)	P(μ, 15)	P(μ, 7)
0	0.970107	0.9/10111	0.985551
I	0.()().1568	0.OOJ5·16	0.003571
2	0.003925	0.OOJ079	0.003572
3	0.003369	0.002709	0.003340
4	0.0021191	0.002422	0.002425
5	0.002480	0.1I()21I1S	0.001173
6	0.(102129	0.0019J2	0.000325
7	O.OOHUO	0.1101595	0.()()OO311
II	0.ooIsn	0.001164	.
9	0.001362	0.000714	.
III	O.IKIII7H	0.000354	.
II	O.OOIOIK	O,()()()135	.
12	0.000873	0.1I111110JK	.
IJ	0.000735	n,IJOIIII1I7	.
14	0.000601	O.1I0()()(11I	.
15	0.000469	0.000000	.
16	O,OOOJ·15		.
17	O.OOO2J5	.	.
18	0.0001·16		.
19	0.OOOOK2	.	–
20	0.0000·11	.	.
21	0.000018	.	.
22	0.000007		–
23	0.000002		.
24	0.000000	.	.
25	0.000000	.	–
26	0.000000	.	.
27	0.000000	.	–
28	0.00000o	.	.
29	6.1E-11	.	.
30	3.5E-12	.	.
31	9.5E-14		.

TABLE D.I6

PROBABILITIES P(μ, v): (d, k, C) =(0,2, 1); 8-ARY PSK ON HIGHWAY

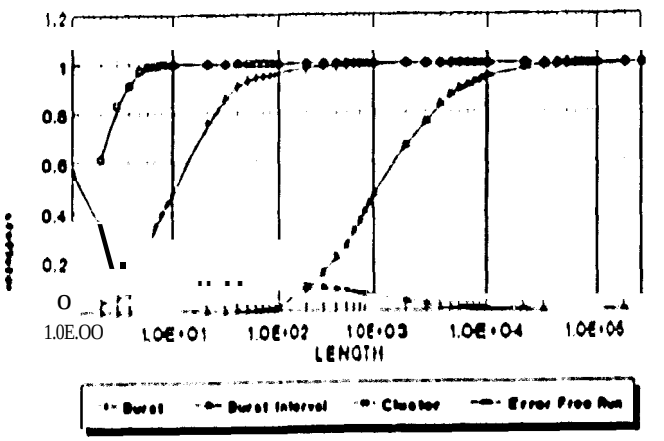


FIGURE 0.16

ERROR DJSTRIOunON: (d, k, C) =(0. 2, I): 8-ARY PSK ON 1II00rwAY

l'	$P(\mu, 3)$	$P(\mu, 15)$	$P(\mu, 7)$
0	0.983424	0.988375	0.991093
1	0.001955	0.001648	0.001788
2	0.001765	0.001482	0.002068
3	0.001589	0.001365	0.002153
4	0.001428	0.001312	0.001685
5	0.001280	0.001307	0.000889
6	0.001146	0.001286	0.000281
7	0.001024	0.001167	0.000010
8	0.000917	0.000922	.
9	0.000826	0.000610	.
10	0.000751	0.000327	.
11	0.000690	0.000148	.
12	0.000638	0.000041	.
13	0.000585	0.000010	.
14	0.000522	0.000001	.
15	0.000415	0.000000	.
16	0.000355	.	.
17	0.000262	.	.
18	0.000176	.	.
19	0.000107	.	.
20	0.000058	.	.
21	0.000028	.	.
22	0.000012	.	.
23	0.000004	.	.
24	0.000001	.	.
25	0.000000	.	.
26	0.000000	.	.
27	0.000000	.	.
28	0.000000	.	.
29	0.000000	.	.
30	1.8E-11	.	.
31	6.1E-13	.	–

TABLE D.7

PROBABILITIES $P(\mu, V)$: $(d, k, C) = (0, 1, 1)$; 8-ARY PSK ON HIGHWAY

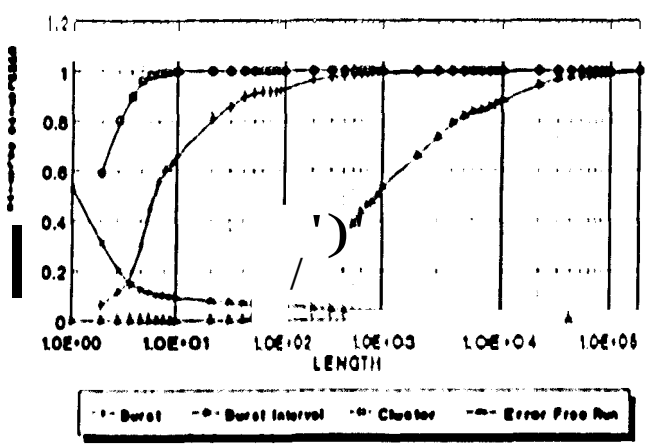


FIGURE D.17

ERROR DISTRIBUTION: $(d, k, C) = (0, 1, 1)$; 8-ARY PSK ON HIGHWAY

μ	$P(\mu, 3J)$	$P(\mu, 15)$	$P(\mu, 7)$
0	0.977821	0,983054	0.986012
I	0.002012	0.001769	0.002791
2	0.001856	0.001759	0.003612
3	0.001712	0,001901	0.003562
4	0.001581	0.002155	0,002480
5	0.001·167	0.002357	0.001161
6	0.OOLl76	0,002311	0.OOOJ34
7	0.00L116	0.001946	0.000045
8	0.001294	0,001374	.
9	0.001303	0.000800	.
10	0.001326	0,000378	.
11	O,OOLl33	0,000141	.
12	0.001290	0.000040	.
13	0.001178	0,000008	.
14	O,(lOIOOI	0.000001	.
15	0.00071B	0,000000	.
16	0,000560	.	.
17	0.000364	.	.
18	0.000214	.	.
19	0,000113		.
20	0.000054	.	.
21	0.000023	.	.
22	0,000008	.	.
23	0.000002	.	.
24	0.000000	.	.
25	0.000000	.	-
26	0.000000	.	-
27	0.000000	.	.
28	0.000000	.	-
29	9.7E·11	.	-
30	6.4E·12	.	-
31	2.IE·1J	.	-

TABLE D.I8

PROBABILITIES $P(\mu, \nu)$: (d, k) =(J. 3); 8-ARY PSK ON JIIOIIWAY

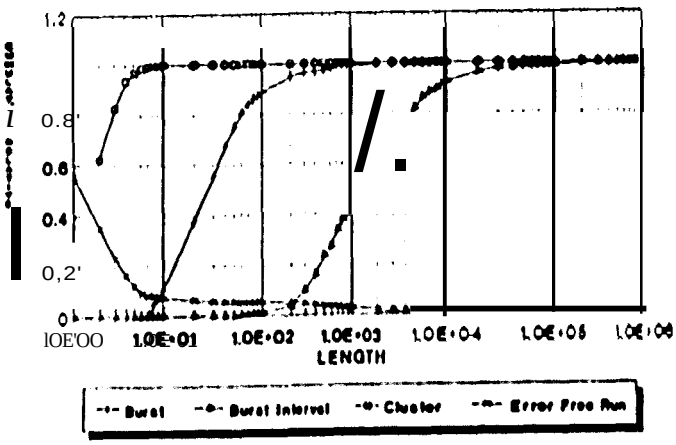


FIGURE D.18

ERROR DISTRIBUTION: (d, k) =(I, 3); 8-ARY PSK ON 1II00IWAY

r	P(μ, 3J)	P(μ, 15)	P(μ, 7)
0	0,984101	0,989727	0,992664
1	0.002601	0,001942	0.oorsn
2	0,002208	0.001633	0.001809
3	O.001H70	0.001391	0,001670
4	0.001581	0.1101216	0.001212
5	0.00133·1	0,001086	0,000609
6	0.001123	0.0(095}	. 0.000186
7	0.1)()09·14	0.000795	O.OOO(}25
8	0,000794	0,000585	.
9	0,000669	0.000367	.
10	0,000566	0,000189	.
11	0,000482	0,0000n	.
12	0,000410	0.000023	.
u	0.0003·16	0.00000S	.
14	0.000286	0.000000	–
15	0.000227	0.000000	.
16	0.000170	.	.
17	0.000118	.	.
18	0.000076	.	.
19	0.0000·44	.	–
20	0.000023	.	.
21	0.000011		–
22	0.00000·1		–
23	0.000001	.	.
2·1	0.000000	.	.
25	0.000000	.	.
26	0.000000	.	–
27	0.000000	.	–
28	0.000000	.	.
29	8.5E·II	.	–
30	5.7E·12	.	–
31	1.9E·13	.	–

TABLE 0.19

PROBABILITIES PC/J. v): (d, k) =(1, 7): 8-ARY PSK ON JIlonWAY

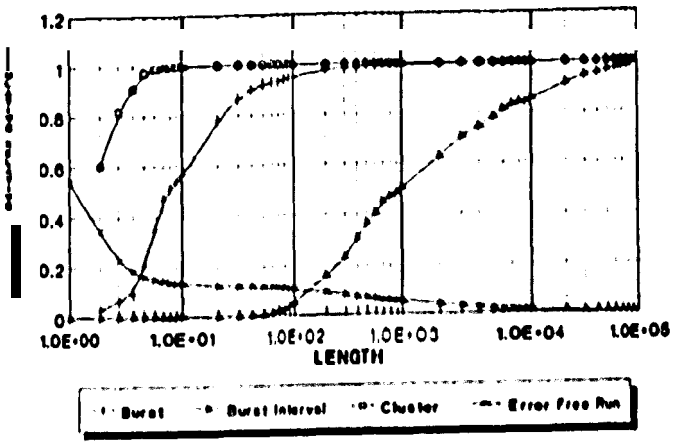


FIGURE D.19

ERROR DISTRIl1tmON: (d, k) =(J, 7); 8·ARY PSK ON JIIOIIWAY

u	P(μ, 31)	P(μ, 15)	P(μ, 7)
0	0.971552	0.9822n	0.98n87
1	0.005319	0.003786	0.003292
2	0.00·1379	0,003070	0.003066
3	0.003598	0,002507	0.002704
4	0.002949	0.002086	0.001901
5	0,002·113	O,OOlnl	0.000931
6	0.001969	0,001497	0.OO02n
7	0.001604	0,001199	0.000037
8	0.00IJ05	0,000860	.
9	0.001061	0,000527	.
10	0.000865	0,000266	.
11	O,O()O707	O.llOOI06	–
12	0.000579	0,000032	.
13	0.000471	O,CIOO007	.
14	O,OO03n	ll,000000	.
15	0.000291	0,000000	–
16	0,000213	.	–
17	0.0001·16	.	–
18	0,000092	.	–
19	0.000053	.	–
20	0,000027	.	.
21	0,000012	.	.
22	0.000005	.	–
23	0,000001	.	.
24	0.000000	.	.
25	0.000000	.	–
26	0.000000	.	–
27	0.000000	.	–
28	0.000000	.	.
29	B.SE·II	.	.
30	5.7E·12	.	.
31	1.8E·13	.	.

TABLE D.20

PROBABILITIES P(μ, *v): U. k)* =(J. 7); 8-ARY PSK ON IJIonWAY

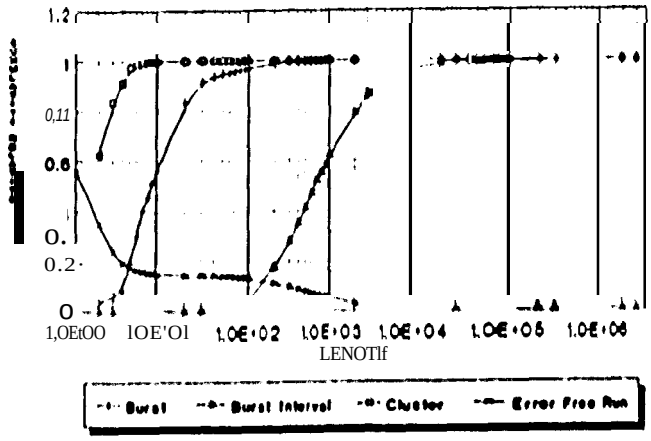


FIGURE D.20

ERROR DISTRIBUTION: *(l, k)* =(I. 7); 8-ARY PSK ON 1II0llWAY

μ	$P(\mu, 31)$	$P(\mu, 15)$	$P(\mu, 7)$
0	0.989064	0.991628	0.992985
1	0.000894	0.000768	0.001080
2	0.000831	0.000735	0.001574
3	0.000n2	0.000756	' 0.001840
4	0.000717	0.000858	0.001485
5	0.000666	0.001016	0.ooono
6	0.000620	0.001138	0.000231
7	0.000582	0.001111	0.000030
8	0.000555	0.000904	-
9	0.000545	0.000596	-
10	0.000554	0.000312	-
11	0.000579	0.000126	-
12	0.000608	0.000038	-
13	0.000621	0.000008	-
14	0.000601	0.000001	-
15	0.000540	0.000000	-
16	0.000443	.	-
17	0.000329	.	-
18	0.0()()219	.	-
19	0.000130	.	-
20	0.000068	.	-
21	0.000032	.	-
22	0.000013	.	-
23	0.000004	.	-
24	0.000001	.	-
25	0.000000	.	-
26	0.000000	.	-
27	0.000000	.	-
28	0.000000	.	-
29	1.8E·1Q	.	-
30	i.is-n	.	-
31	3.4E·]3	.	-

TABLE 0.21

PROBABILITIES $P(\mu, \nu)$: PN-SEQUENCE; 4·ARY PSK IN CITY

μ	$P(\mu, 31)$	$P(\mu, 15)$	$P(\mu, 7)$
0	0.965627	0.973622	0.977853
1	0.002699	0.002336	0.003186
2	0.002520	0.002231	0.004688
3	0.002351	0.002264	0.005728
4	0.002193	0.002530	,0.004888
5	0.002045	0.003008	0.002683
6	0.001910	0.003461	0.000852
7	0.001793	0.003532	0.000119
8	0.001704	0.003027	-
9	0.001658	0.002111	-
10	0.001666	0.001169	-
11	0.001730	0.000501	-
12	0.001825	0.000160	-
13	0.001904	0.000035	-
14	0.001907	0.000005	-
15	0.001787	0.000000	-
16	0.001540	.	-
]7	0.001205	.	-
18	0.000849	.	-
19	0.000534	.	-
20	0.000298	.	-
21	0.000147	.	-
22	0.000063	.	-
23	0.000023	.	-
24	0.000007	.	-
25	0.000002	.	-
26	0.000000	.	-
27	0.000000	.	-
28	0.000000	.	-
29	0.000000	.	-
30	8.2E·ll	.	-
31	2.7E·12	.	-

TABLE D.22

PROBABILITIES $P(\mu, \nu)$: PN-SEQUENCE; 4-ARY PSK ON 1lIOIIWAY

μ	$P(\mu, 31)$	$P(\mu, 15)$	$P(\mu, 7)$
0	0.990060	0.992573	0.993958
1	0.000998	0.000852	0.001255
2	0.000910	0.000826	0.001584
3	0.000830	0.000867	0.001533
4	0.000757	0.000956	0.001044
5	0.000693	0.001023	0.000474
6	0.000640	0.000985	0.000131
7	0.000601	0.000814	0.000017
8	0.000578	0.000562	-
9	0.000570	0.000319	-
10	0.000569	0.000147	-
11	0.000561	0.000053	-
12	0.000533	0.000014	-
13	0.000479	0.000002	-
14	0.000400	0.000000	-
15	0.000307	0.000000	-
16	0.000216	.	-
17	0.000137	.	-
18	0.000079	.	-
19	0.000041	.	-
20	0.000019	.	-
21	0.000007	.	-
22	0.000002	.	-
23	0.000000	.	-
24	0.000000	.	-
25	0.000000	.	-
26	0.000000	.	-
27	0.000000	.	-
28	0.000000	.	-
29	2.4E·l1	.	-
30	J.5E·12	.	-
31	4.7E·14	.	-

TABLE D.2J

PROBABILITIES $P(\mu, \nu)$: PN-SEQUENCE; 8-ARY PSK IN CITY

μ	$P(\mu, 31)$	$P(\mu, 15)$	$P(\mu, 7)$
0	0.974153	0.980280	0.983730
1	0.002357	0.002068	0.003243
2	0.002174	0.002050	0.004207
3	0.002005	0.002210	0.004150
4	0.001851	0.002505	0.002884
5	0.001715	0.002742	0.001346
6	0.001606	0.002691	0.000385
7	0.001534	0.002266	0.000052
8	0.001504	0.001597	-
9	0.001513	0.000927	-
10	0.001539	0.000437	-
11	0.001547	0.000163	-
12	0.001498	0.000046	-
13	0.001369	0.000009	-
14	0.001163	0.000001	-
IS	0.000909	0.000000	-
16	0.000650	.	-
17	0.000422	.	-
18	0.000248	.	-
19	0.000131	.	-
20	0.000062	.	-
21	0.000026	.	-
22	0.000009	.	-
23	0.000003	.	-
24	0.000000	.	-
25	0.000000	.	-
26	0.000000	.	-
27	0.000000	.	-
28	0.000000	.	-
29	I.lE·lO	.	-
30	7.0E-12	.	-
31	2.3E·13	.	-

TABLE D.24

PRODAnU.mES $P(\mu, \nu)$: PN-SEQUENCE; 8-ARY PSK ON HIOHWAY

APPENDIXE

PROGRAM FOR MEALYTO MOORE MACHINE CONVERSION AND FSMMINIMIZATION

This appendix contains a program listing written in Turbo C. The program can convert (rom Mealy to Moore machine nnd minimize a given finite-state machine; nil machines cnn be saved and recalled from disk with user definable filenames.

The program is based on the matrix representation of finite-state machines, discussed in chapter 3 nnd chapter 5. By entering thc E nnd r matrices for a given machine, the necessary operations can be performed by the program. THC program is also available on a noppy disk at the back of this thesis, with filename FSM.C.

```
#inclllldc<conio.h>
#inc\udc<stdio.h>
#inc\udc<dir.h>
#inc\udc<dos.h>
#inc\lIdc<string.h>
#inc\lIdc<math.h>

#define     TLC     '\xC9'                    I*Dcfinc characters for window.'
#define     TRC     '\xBB'
#define DLC         '\xC8'
#define DRC         '\xBC'
#define JlLNE       '\xCD'
#define VLNE        '\xBA'
#define RCNTR       '\xB9'
#define LCNTR       '\xCC'
#define TCNTR       '\xCB'
#define CNTR        '\xCE'
#define DCNTR       '\xCA'

char (*elements[6]) = {"Create", "Mealy to Moore", "Moore to Mealy", "Reduction",
                "Load", "Save");
char ch =n';
int cur_row, k;

void reversevideofvoid):
void nvideolvoid):
void drawbordcr(int tlx, int tly, int brx, int bry,
            int lycntr, int lxcntr):
void up_arrow(void);
void down_arrow(void);
void arrow(void);
void filcnms(chnr pathname[81]);
void cwindow(int nstate, int k);
void cmalrix(int nstate, double alpha, int outputsl;
void createflnt tx, int ty, inl by):
void mealytvoid):
```

```
void moorefvold):
void reductiontvold):
void loadfint tx, int ty, int by);
void savelvold):
void quitfint tx, int ty, int by);
void selectflnt tx, int ty, int by);
void twindow(int amnt, int tx, int ty, int by);

/*****************************************************************t
void reversevideofvold)

   tcxtattr(llLACK + (WIIITE«4»;


/*****************************************************************,
void nvideotvoid)

   tcxtattr(WIIITE + (DLACK«4»;


/*****************************************************************,
void drawborderfint tlx, int tly, int brx, int bry,
int Iyentr, int lxcntr)
(
   int i;

   for(i =tlx + 1; i <= brx • 1; i++)

        gotoxyfi, tly):
        pUlch(lILNE):
        gotoxyfi, bry):
        putch(IILNE);


   (or(i =tly + I; i <= bry - 1; i++)
   (
        gotoxyltlx, 1):
```

```
        putch(VLNE);
        gotoxytbrx, i);
        plltch(VLNE);


  gotoxyltlx, tly);
  plltch(TLC);
  gotoxyfbrx, tly);
  plltch(TRC);
  gotoxyulx, bry):
  plItch(nLC);
  gotoxyfbrx, bry):
  plItch(nRC);

  if(Jycnlr != 0)
  (
      for(i =tlx + 1; i <= brx- I; i++)
      (
          gotoxyfi, lycntr):
          putch(J ILNE);


      gotoxyttlx, lycntr):
      putch(LCNTR);
      gotoxyfbrx, lycntr):
      pUlch(RCNTR);


  if(Ixcntr != 0)
  (
      for(j =tly + 1: i <= bry • 1: i++)
      (
          gotoxytlxcntr, i);
          putch(VLNE);


      gOloxy(Jxcntr. tly);
```

```
        putch(TCNTR):
        gotoxytlxcntr, lycntr):
        plltch(CNTR):
        gotoxytlxcntr, bry):
        plltch(DCNTR):


/*.....................................................'.....................,
void up_nrrow(void)
(
   cur_row = whcrcyO:

   switchlcur_row)

       case 1 :          nvidcoO:
                         gotoxytl, cur_row);
                         cputslelcmcntslcurrow-Ill;
                         cur_row =6:
                         reversevideof):
                         gotoxytl, cur_row);
                         cpntsfclcmentslcur.row-Il):
                         break;

       case 2 :
       case 3 :
       case 4 :
       case 5 :
       case 6 :          nvidcoO:
                         gOloxy(t. cur_row);
                         cpntsfelcmcntslcur_row- J));
                         reversevideofl:
                         cur_row--:
                         gotoxy(t. cur_row);
                         cputsfclememslcur.row-l ));
                         brenk;
```

```
/*.....................................................................t
void down arrowfvold)

  switchfcur_row =whcrcyO>
  (
      case 6 :          nvidcoO;
                        gotoxyf}, cur.row):
                        CpUls(clcmcnls[curJow.l ));
                        cur_row =1;
                        reverscvideof):   .
                        gotoxyf l, cur_row);
                        cpuls(clcmcnls[cur_row.t ));
                        break;

      case 1 :
      case 2 :
      case 3 :
      case 4 :
      case 5 :          nvideof):
                        gotoxytl , cur_row);
                        cputsfelcmcntslcur.jow-l ]);
                        reversevideof):
                        cur_row++;
                        gotoxytt, cur_row);
                        cputsfelernentslcurrow-I ));
                        break;


/*.....................................................................t
void arrowfvoid)

  switch(ch =getchf))
  (
      case 'P0 :        down_nrrowO;
                        break;
```

```
        case IH' :                  up_nrrowO:
                                     break;


/*****************************************************************,
void filenmstchar pathnameldl])
(
   int count =3, col = 1;
   struct ffblk filcinfo;

   window(31, 9, 74, 18);
       gotoxy(l8, 1);
       printf("%s\\•.MCII          ", pathname):
   if(findfirst('\•.mch", &fileinfo, 0) != 0)

       gotoxyfl , 2):
       cputs("No file's foundll!"):

   else

       gotoxytl , 2);
       printf("%s", filcinfo.ff_name);
       while(findncxt(&fileinfo) = 0 )

           gotoxyfcol, count);
           printf("%s\n", fileinfo.fCnnmc);
           count++;

           if«count = 1J) && (col = 14))
           brenk;

           if(count == 1J)
           (
               count =2;
               col =14;
```

```
/.....................................................  t
void loadlint tx, int ty, int by)
(
   char pathnllme[81 l, ·tbllffcr;
   int tbufsize:
   struet ffblk fileinfo;

   if(gctcwd(pllthnnmc, 80) == NULL)
       perrorf'Error establishing directory!!!!");
   else

       tbutsize =2.(22 • 7 + 0.02 - 7 + 1);
       if((tbuffer = (char *) malloctbufslze)  == NULL)

           cputs("Error allocating buffer!!!!");
           exltfl);


       if(!gettext(7. 7,22, 12, tbufferr)
       (
           cputs("Error saving text!!!l"):
           exitfl);
       )
       drawborderf l, I, 15, 3, 0, 0);
       gotoxy(2, 2);
       printf("%s"•• MCII", pathname):

       whilc«ch =getch(» !=\x1B')
       (
           switchlch)

  case \r' :                filenmsfpathname);
                            break;
```

```
        }


        windowltx, ty, (tx + 15), by);
        if(!puttext(7, 7, 22, 12, tbuffcr))
        cputsf'Error restoring text!!!! "):
        gotoxytl , 5);



/*****************************************************************,
void ewindowfint nstate, int k]

    int m, n:
    int fbufsize] 16];
    char .fbuffcr[16];

    fbufsiza[k] =2.«nstnte + 3) - 3).«nstate + 3) • 2);
    if«fbllffer[k] = (char *) mallocffbufsizejkl)) == NULL)
    (
        cputs("Error allocating buffcr!!!!");
        exit(t);
    )
    if(!gettcxt(4, 3. (nstate + 3), (nstate + 3), fbllffcr[k]))
    (
        cputs("Error saving textllll"):
        cxit(t ):
    )
    window(4. 3. (nstate + 3), (nstate + 2»;
    c1rscrO;
    window(4. 3. (nstate + 3), (nsmte + 3»;
    for(m = 1: m <= nstate: m+t)

        for(n = 1: n <= nstate: nt+)

            gotoxytn, m):
            putch(OxOF);
```

```
/*•••••••••••••••••••••••••••••••••••••••••••••••••••••••••••t
void earrowfvold)

   char p;

   switch( P =getch())
   (
        case 'B' : clrserf):





/*                                                                                                    ,
void ematrix(int nstate, double alpha, int outputs)
(
   int i, j;
   char 8_e)em[t6][80][25];

   for(i =1; i <= nstate; i++)

        gotoxyt}, i + 2);
        printf("o/od". 0;



   drnwbordcr<J, 2. (nstate +4), (nstate + 3),0,0);
   window(4, 3, (nstate + 3), Instate + 3));

   for(j = l; j <= nstate; j++)
   (
        forO = 1: i <= nstate: iff)

             gotoxyfi, j);
             putch(OxOF);
```

```
(or(k =0; k <= ((int) alpha - I); k++)
(
    forO = 0; j <= 79; j++)
I
(orO =0; i <= 24; i++)
(
    o_clcm[klij)[i] = '0';


k=O;
whilc(k < (int) alpha)
{
    forO = I: j <= nstate: j++)
    (
        for(i =1; i <= nstate: i++)

        whilc(!(a_clcm[kJ(i-llij-t] >= '0') || !(o_clcm[k][i-llO-l] <= 'T'))

        gotoxyfi, j);
        a_clcm[k](i-JlU-I] =getchef):

        switch(a_clcm[k)[i-JlU-}])
        (
            case 0x1B           k++;
                                ewindowfnstate, k);
                                i =j =I;
                                break;

            case '\x0' :        earrowf):
                                break;
```

```
    k++:
    ewlndowtnstate, k):


/                                                                    t
void createflnt tx, int ty, int by)
(
    int i, sbufsize, tbufsize, nstate, stntcslzl = (0,0), outputs:
    double alpha, inputs:
    char .sbuffer, .tbuffcr, chs;

    sbufsize =2.(75 - 5 + 1).09 - 5 + 0:
    tbefsize =2.(20 - 6 + 1).05 - 12 + 0:
    if«sbllffcr =(char .) malloctsbufslzel) == NULL)
    (
        cputst'Error allocating bufferllll"):
        exit0 );


    if(tbllffer = (char .) mallocltbufsizel) = NULL)
    (
        cpllts("Error allocating buffer!!!!"):
        exittl):


    if(!gettext (5, 5, 75, 19, sbuffcr) ll
         Igcttext(6. 12. 20, 15, tbuffcr»

        cputs(*Error saving textllll"):
        cxit(I );


    window(6, 12.29. 14};
    elrscrO:
    drnwborderO. I, 23, 3, 0, 0):
```

```
ch = 'a';
whllekh != '\r')
{
    nstate = inputs = outputs = 0;
    gotoxy(2.2):
    cputsl"                                    H):
    gotoxy(2. 2):

    eputs("lIow mnny states? H):

    whilc(l(nstnte > 0) III(nstate < 23))
    {
        gotoxy(l9.2);
        for(i = 0: i <= 1; i++)
        {
chs = gcteheO:
            if(!(ehs >= 0,(30) " !(chs < 0,(40))
                gotoxy(l9,2):
            else
                statesli] = (int) chs - 48;
        }
        nstate = 10*statcs[0) + statesltl:
    }
    delay(l00):

    gotoxy(2. 2):
    cputs("                                    H):
    gotoxy(2. 2):
    cputs("lnput symbols? H);

    whilc(l(inputs > 0) II !(inputs < 9))
    {
        chs = gctehcO:
        if(!(chs > 0,(30) II Hchs < 0,(40))
            gotoxy(t7,2);
        inputs = (int) chs • 48;
    }
```

```
        delay(1(0):

        gotoxy(2, 2);
        cputs("
        gotoxy(2, 2);
        cputs("Output symbols? ");

        while(!(outputs > 0) || !(outputs < 9))
        (
            chs = getchef):
            if(!(chs > Ox30) || !(chs < 0x40))
                gotoxy(18, 2);
            outputs = (int) chs- 48;
        )
        dclay( J00);

        gotoxy(2, 2);
        cputs("Enter to continue.");
        eh = getchf):


    clrscr0;

    puttext(6, J2, 20. J5, tbuffer):

    alpha =pow(2.0. inputs);
    window(J' I, BO. 25);
    clrscr0;
    printf("You have to enter %d E and %c matrices.", (int)alpha, 0xE2);
    ematrix(nstate, alpha, outputs);
    window(tx ty, (tx + 15), by);
    gotoxy(J, J);

,..............0.....................................0000/
\'aidquit(int tx, int Iy. int hy)
(
```

```
    windowO. 1.79.25);
    gotoxy(22. 25):
    nvideoO:
    CpUIS("Surc you want 10 exil[N]? H);
    ch =lolower(gctch());
    if(ch == 'y')
        clrscr();
    else

        gotoxy(22. 25);
        clreolf):
        windowttx, ty, (tx + 15), by);
        gotoxytt, cur.jow):


/*****************************************************************,
void selectflnt tx, int ty, int by)
(
    nvideoO;
    switchfcur_row =whercyO)
    (
        case 1 : createttx, ty, by);
                        break;
        case 2 : mealyf):
                        break;
        case 3 : moorcO;
                        break;
        case 4 : reductionf):
                        break;
        case 5 : loadftx, Iy, by);
                        brenk;
        case 6 : save();
                        break;
```

```
void twindow(int arnnt, int tx, int ty, int by)
(
   int i, butsize, oty;
   char .bllffcr;

   oty = ty;
   bulsize =2.(by - ty + O.((lx + 15) - tx + J);
   if((buffcr =(char .) maJloc(bufsizc» == NULL)
   (
      cputsf'Error allocating buffer!!! l"):
      exittl):


   nvideoO:

   forf = 1; i <= arnnt; i++, oly++)
   {
      gotoxyttx, oty);
      cputsfelernentsli-I ]);
   }

   highvideol);
   gOloxy(t6,7);
   cputs("Mcnu");

   gOlOXY(SO, 7);
   cPllts("Window");
   normvidcoO;

   windowux, ty, (tx + 15), by);

   if(!gcItCXI(tx, ty, (tx + 15), by, buffer»
   (
      cputs("Error saving textllll"):
      exitll]:
```

```
  reversevideo0;
  cputslelcmentslul):

  whilelch != 'y')
  (
     ch =getcht):
     switchfch)

        case '\x0'      : DITOWO;       /*Check for arrow keys.'
                                  break:

        case '\r'       : select(tx, ty, by);          /*Check for Enter key.'
                                  break;

        case '\x Ill' : quit(tx, ty, by);
                                  break;



/*''''''''''''''''''''''''''''''''''''''''''''''''''''''''''' ''''';
void mainlvoid)

  clrserf):
  drawborder(5. 5. 75. 19.8,30):
  twindow(6.7. 10. 17);
```

APPENDIXF

GAL LISTINGS

The GAL program listings for the DSP flnite-state machine and the 8254 timcr card are presented in this appendix. As mentioned in Appendix A, the DSP finitc state machine used two GAL 20V8's and the timer card used only one GAL 16V8.

It must be noted that the symbols &, land! indicate, respectively, the logical AND, OR and NOT functions. This listings will also serve as examples for GAL programming.

```
DEVICE     20V8;
TITLE      ADRESS DECODING FOR PC;
NAME       ADR.PLD;
SIGNATURE  ADR_DEC;

/* ADRESS LINE INPUTS.,

PIN        1    =    CLK;
PIN        2    =    A19;
PIN        3    =    AlB;
PIN        4    =    AI?;
PIN        5    =    A16;
PIN        6    =    A15;
PIN        7    =    A14;
PIN        8    =    AI3;
PIN        9    =    AI2;
PIN        10   =    Al1;
PIN        11   =    D0;
PIN        12   =    GND;
PIN        13   =    OE;
PIN        14   =    LOE;
PIN        24   =    VCC;

/* CIIIP SELECT OUTPUTS .,

PIN        15   =    SO;
PIN        16   =    SI;
PIN        17   =    S2;
PIN        18   =    S3;
PIN        19   =    S4;
PIN        20   =    CCSO.
PIN        21   =    CCSl;
PIN        22   =    RSTMS;

/* GENERATING CIIIP SELECTS .,

ISO =A19 & A18 & IAl? & lA16 & Al5 & Al4 & IA13 & IAl2 & IAI1;
```

```
!SI =Al9 & AlB & !AI? & !A16 & AI5 & AI4 & !A13 & !A12 & All;

!S2 =Al9 & AlB & !AI? & !A16 & Al5 & Al4 & !A13 & A12 & !All;

!S3 =AI9 & AlB & !AI? & !A16 & AI5 & AI4 & !A13 & A12 & All;

!S4 =A19 & AlB & tAl? & !A16 & Al5 & A14 & AIJ & !At2 & tAll;

!CCSO =Al9 & AlB & !AI? & !A16 & Al5 & Al4 & !A13 & !A12 & !All
    I Al9 & AI8 & !Al? & IAI6 & Al5 & Al4 & !AIJ & !At2 & All;

tCCSI = A19 & A 18 & !Al? & !A16 & Al5 & Al4 & tA13 & A12 & IAll
    I A19 & AlB & !Al? & tA16 & Al5 & Al4 & tA13 & A12 & All;

/* latch programmed in GAL ./
RSTMS.D =LOE & RSTMS.Q
        IDO& tLOE;
```

```
DEVICE     20V8:
TITLE      ADRESS DECODING FOR TMS;
NAME       TADR.PLD;
SIGNATURE  TMSADEC;

PIN    1    =    A3;
PIN    2    =    A4:
PIN    3    =    AS;
PIN    4    =    A6;
PIN    5    =    A7:
PIN    6    =    AS;
PIN    7    =    A9;
PIN    S    =    AlO;
PIN    9    =    A11;
PIN    10   =    MEN;
PIN    11   =    DEN;
PIN    12   =    GND;
PIN    13   =    WE;
PIN    14   =    RSTMS;
PIN    23   =    CSO;

PIN    15   =    SO;
PIN    16   =    SI;
PIN    17   =    MEMW;
PIN    18   =    EN;
PIN    19   =    POE;
PIN    20   =    RSTMSL;
PIN    21   =    DDlR;
PIN    22   =    N.C;
PIN    24   =    VCC;

ISO  =!All & OMEN I IWE);

lSI  =All & OMEN I IWE);

MEMW = I(IWE & I(IA3 & IA4 & IAS & !A6 & !A? & !A8 & IA9 & IAIO & !AII));
```

```
lPOE  =IDEN & \VE & (!A3 & !A4 & lAS & IA6 & !A? & !AB & IA9 & !AIO & IAll )
      IDEN & !WE & (lA3 & IA4 & lAS & IA6 & IA? & !AS & !A9 & IAIO & fAll):

lEN  =ICSO & fRSTMS:

RSTMSL = !RSTMS;

DDIR =DEN & MEN:
```

```
DEVICE    16V8:
TITLE     ADDRESS DECODING FOR TIMER CARD IN PC;
NAME      IOPORT.PLD
SIGNATURE I/O PORT;

PIN  1   =  A2;
PIN  2   =  A3;
PIN  3   =  A4;
PIN  4   =  AS;
PIN  5   =  A6;
PIN  6   =  A7;
PIN  7   =  AS;
PIN  B   =  A9;
PIN  9   =  AEN;
PIN  10  =  GND;

PIN  12  =  CSO;
PIN  13  =  csr,
PIN  15  =  eS2;
PIN  t4  =  ccs
PIN  20  =  vee,

!CSO  =IAEN & !A9 & IAB& !A7 & A6 & AS& A4 & AJ & A2;

!CSt  =!AEN & !A9 & !AB & !A? & A6 & A5 & A4 & AJ & !A2;

ices  =!AEN & !A9 & !A8& !A? & A6& A5& A4 & AJ & A2
      I !AEN & !A9 & !AB & !A? & A6 & AS & A4 & AJ & IA2
      IIAEN & !A9 & !AB & !A? & A6 & A5 & A4 & IA3 & A2;
```

APPENDIXG

PC TO TMSPROGRAM MEMORY TRANSFER PROGRAM

This appendix contains a program listing in Turbo C. which enabled the PC to test the TMS program memory and to transfer precompiled program memory to the static RAM common to both processors.

```
#include <stdlib.h>
#includc<dos.h>
#includc<stdio.h>
#includc<string.h>
#includc<ctype.h>
#includc<conio.h>

void test_mem();
void rcad_mpoO:
int ctoi( int nib):

/*****************************************************************,
void testrnemf)

  int i =0;
  unsigned scgO =OxccOO. seg! = OxcdOO. off;
  char x =Ox80. y = Ox?f. xt, yt;

  pokebl OxccOO. OxO, Oxl );
  clrscrO;

  for( off =0; off <= 0xfff; of(++ )
  (
    pokebl segu, off. x );
    pokebl seg1. off. Y):
    xt =peekb( segf), off );
    yt=peekbl seg1. off );
    if( xt != x II yt != y )
      i++;
  J
  if( i >0 )
    printf("Program memory of TMS has %d errorsll", i );
  else
    puts( "Program memory of TMS is OKIl");
```

```
/*****************************************************************,
void rcnd_mpoO
(
 int len, i, j =0, k =0, I, m=0, n, off;
 unsigned scgO =OxccOO, scg1 =0xcd00;
 int dtn(4096), In, hn;
 char infilcnnmc[lO), ans =a';
 char extil] = ".MPO";
 FILE .infilc;

 printf( "\n\nEnter TMS object file [•.MPO): " );
 gets( infilennmc );

 len =strlcn( infilennme );
 for( i =0; i <= len; i++ )
 (
  if( infilcnnmc[i] != ',' )
    j++;
 )
 if( j !=len)
  strcat( infilcnamc, exti );

 if« infile =fopcn( infilennmc, "rb" »== NULL)
 (
  printf( "\n\n\"%s\"", infilename );
  perrorf "" );
  exitll):
 )

 whilc« dtn[k) =getcl infile» != EOF )
  k++;

pokebl OxccOO, 0x0, Oxl );

if( dtn[O) == 0x4b )
 n =13;
```

```
fore off= 0,1 = n: I <= k: I += 5 )
(
 switch( dtn[l] )
 (
  cnse Ox39 :
              off = ctoi( dta[l+4] ) + 16*cloi( dto[J+3] )
                  + 256*ctoi( dta[l+2] ) + 4096*ctoi( dtn[l+ 1] );
              brenk;

  case Ox42 :
              In =ctoil dlo[l+4]) + 16*cloi( 01011+3] );
              hn =ctoi(dla[J+2» + 16*Cloi(dln[J+t]);
              pokeb( seg0, off, In );
              pokeb( segl, off, hn );
          off++;
              brenk;

  case Ox46 :
          1= m +=77;
              break;

  case Ox3a :
              fclosc( infilc );
              printf( "\nConverted file has %d bytes.", k );
          printf'( "\n\nStart TMS (YIN): ");
              while( !kbhitO )
              (
                gotoxyl 18, 8 );
                ans =toupper( getchcO );
                if( ans = 'Y' )
                (
                  pokeb! OxccOO, 0, 0 );
                  prinlf( "\n\nT'MS cxecutlng file \"%s\".", infilennmc);
                  cxit(l);
                )
                if( ans == 'N')
                (
```

```
                    print(( "\n\nProgram terminated without starting TMS.- );
                    exittl]:
                  }
                )
   )
  )
)

/*****************************************************************,
int ctol! int Inib )

 int i, j;

 switch( Inib )

  case Ox30 : i = 0;
            break;
  case Ox31 : i = I;
            break;
  case Ox32 : i =2;
            break;
  case Ox33 : i =3;
            break;
  case Ox34 : i =4;
            break;
  case Ox35 : i = 5;
            break;
  case Ox36 : i =6;
            break;
  case Ox37 : i = 7;
            break;
  case Ox38 : i = 8;
            break;
  case 0:<39 : i = 9;
            brenk;
  case Ox41 : i =10;
            brenk;
```

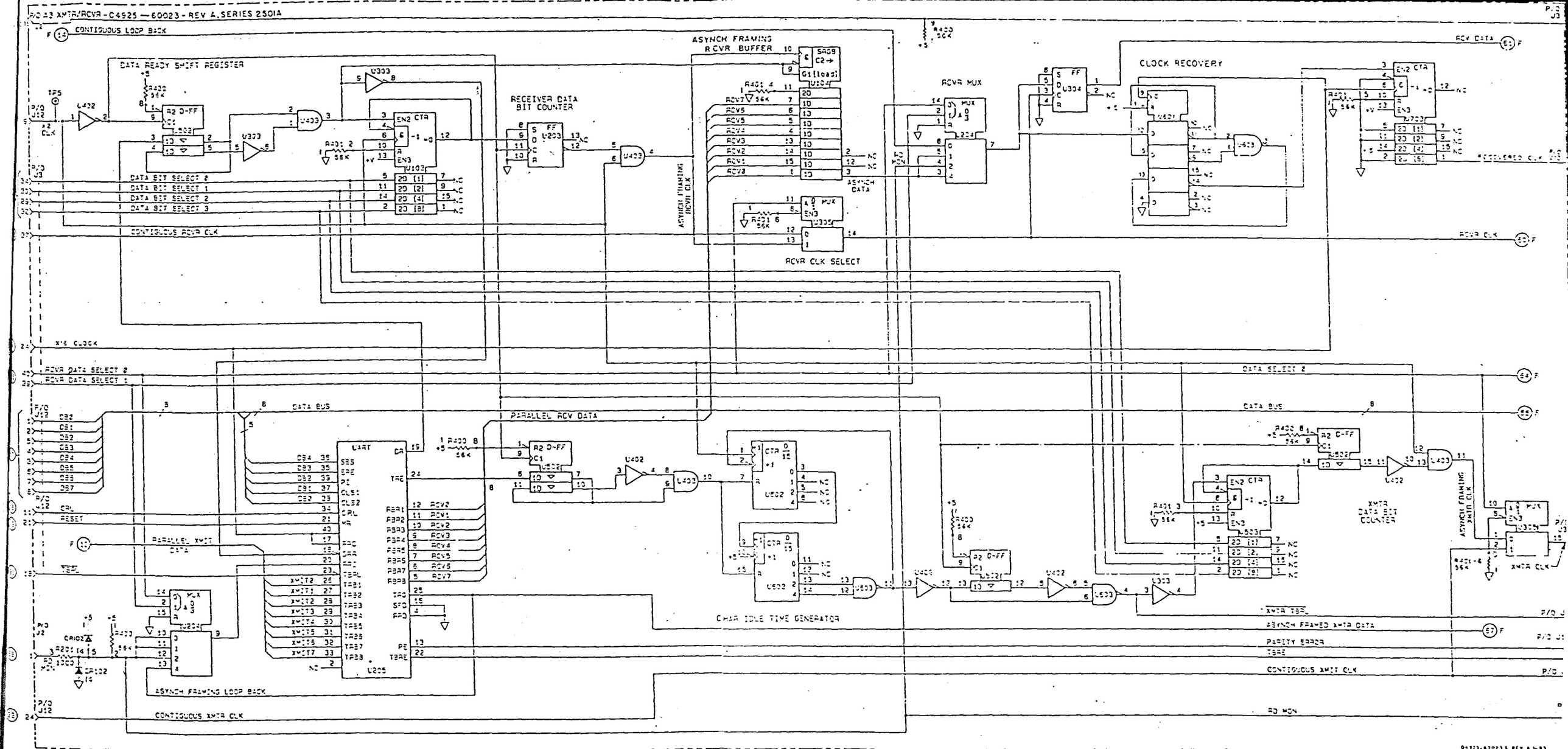
```
    case 0:<42: i =11;
            brenk:
    case 0x43 : i =12;
            brenk:
    case 0x44 : i =13;
            brenk:
    case 0x45 : i =14;
            brenk:
    case 0:<46: i = 15;
            brenk:
  }
  rctum{ i );
}
/.......•.....•.....................•.................•.•...•;
mainO

test_memO: /* void which tests the static RAM */
read_mpoO: /* void which transfers program memory to RAM .,
}
```

APPENDIXH

HP4925BCLOCKRECOVERYCIRCUIT

With the frequency domain results (chapter 6) the liP 492SB bit error rate meter played a very important role in the experimental set up. To ensure complete documentation and repeatability of results, it was decided to include the clock extraction circuit diagram of the UP 49258 BER meter; this appendix contains just that,


P/O A3 XMTR/RCVR-04925 — 60023 - REV A, SERIES 2501A
CONTIGUOUS LOOP BACK
DATA READY SHIFT REGISTER
RECEIVER DATA BIT COUNTER
ASYNCH FRAMING RCVR BUFFER
RCVR MUX
CLOCK RECOVERY
RCV DATA
RECOVERED CLK
DATA BIT SELECT 0
DATA BIT SELECT 1
DATA BIT SELECT 2
DATA BIT SELECT 3
CONTIGUOUS RCVR CLK
ASYNCH FRAMING RCVR CLK
ASYNCH DATA
RCVR CLK SELECT
RCVR CLK
XMT CLOCK
RCVR DATA SELECT 0
RCVR DATA SELECT 1
DATA SELECT 2
DATA BUS
PARALLEL RCV DATA
UART
PARALLEL XMIT DATA
XMTR DATA BIT COUNTER
ASYNCH FRAMING XMTR CLK
XMTR CLK
CHAR IDLE TIME GENERATOR
XMTR TBRL
ASYNCH FRAMED XMTR DATA
PARITY ERROR
TBRE
CONTIGUOUS XMIT CLK
ASYNCH FRAMING LOOP BACK
CONTIGUOUS XMTR CLK
RD MON

REFERENCES

[I] G. Calhoun, "Digital Cellular Radio," Ar1ech House, Inc., Norwood, 1988.

(2) P.A. Siegel, "Recording codes (or digital magnetic storage," IEEE Trnnsactlons on Magnetics, vol. Mng-21, no. 5, September 1985, pp 1344·1349.

[3J K.A. Schouhammcr Immink, "Coding techniques for optical and magnetic recording channels," Prentice Hall Inc, New York, 1990.

(4) \V.I. Fletcher. "Engineering approach to digital design," Prentice Hall, New Jersey, 1980, pp111-118.

[5] J.D. Parsons and J.G. Gardiner, "Mobile communication systems," John Wiley and Sons lnc., New York, 1989.

[6] CCIR: International Radio Consultive Committee. "Recommendations and reports of the CCIR, 1981," report 780·1.

(7) F. Swarts, "Markov characterisation of foding channels," M.Eng thesis, Rand Afrikaans University, June 1991.

ISJ D.R. Oosthuizen, "Markov models for mobile rndio data communication systems," M.Eng thesis, Rand Afrikaans University, December 1990.

19] L.N. Kanal and A.R.K. Sastry, "Models for channels with memory lind their applications to error control," Proceedings of the IEEE, vol 66, no 7, July 1978, pp 724-744.

ltoJ J.G. Proakis, "Digital communications," McGraw-lIi11, London, 1985.

11 t] W.C.Y. Lee, "Mobile communication engineering," McGraw-lIi11, New York, 1982.

112J A.N. Kruger, "Introduction of the first cellular mobile radiotelephone system in the Republic of South Africa," Elckiro, Technical journal of the Department of Posts and Telecommunications, vol I, no 2, 1986.

113] A. Gill, "Introduction to the theory of finite-state machines," McGraw-lIiII, New York, J962.

114] F.C. Hennle, "Finite-state models for logical machines," John Wiley and Sons Inc., New York, 1968.

115] C.V. Freiman and A.D. Wyner, "Optimal block codes for noiseless input restricted channels," Information and Control, vol. 7, 1964, pp 398-415.

116] W.H. Kautz, "Fibonacci codes for synchronization control," IEEE Transactions on Information Theory, volIT-J 1,1965, pp 284-292.

117] A. Gabor, "Adaptive Coding for self-clocking recording," IEEE Transactions on Electronic Computers, vol EC-J6, 1967, pp 866-868.

118] D.T. Tang and L.R. Bahl. "Block codes for a class of constrained noiseless channels," Infonnation nnd Control, vol 17, 1970, pp 436-461.

[19] P.A. Franaszek, "Sequence-stare encoding for digital transmisslon," Bell Systems Technical Journal, vol 47, January 1968, pp 143-157.:

120] K. Norris and D.S. Bloomberg, "Channel capacity of charge constrained runlength limited codes," IEEE Transactions on Magnetics, vol MAO-I?, no. 6, November 1981, pp 3452-3455.

121] C.E. Shannon, "A mathematical theory of communlcauoa," Bell Systems Technical Journal, vol 27, July 1948, PI) 379-423.

[22] R.L. Adler, D. Coppersmith and M. Hassner, "Algorithms for sliding block eedes, An upplication of symbolic dynamics to information theory," IEEE Transactions on Information Theory, vol rl'-29, JanuDry 1983, pp 5-22.

12.1] L. Hong and P. Ostapo, "Coxles for self-clocking transmission," IBM Technical Journal of Research and Development, vol 20, July 1975, pp 358-365.

[24] K.V. Cauermole, "Prlnciples of digital line coxling," International Journal on Electronics, vol 55, July 1983, pp 3-33.

1251 J.M. Griffiths, "Binary code suitable for line transmission," Electronics Letters, vol 5, 1969, pp 79-81.

126J E.V. Khrishnamunhy, "lniroductory theory of computer science," John Wiley and Sons lnc., New York, 1985.

127] G.L Cariolaro et al, "AnolY5is of codes and spectra calculations," International Journal of Electronics, vol 55, no I, t983, pp 33-79.

[28] L. Borha, II.C. Ferreira, I. Broere, "New multilevel line codes," IEEE Global Telecommunication Conference, December 2-5, 1990, pp 501.4.1-501.4.6.

129] A.M. Patel, "ImproVed encoder and decoder for a byte-oriented rate 8/9 (0, J) code," InM Technical Disclosure Dulletin, vol 28, 1985. pp 1938.

130] Hewleu Packard, "Operating Ind service manulli for the HP 4925B bit error rate test set And accessories," Manual part no 04925-90021.

[31J F.G. Stremler, -lntmductinl1 In communication systems," Adklison-Wesley, Sydney, 1982.

[32] II.C. Ferreira, "On de free mngnetic recording codes generated by (jnile-state machines," IEEE Transactions on MlIgnclics, vol MAO-19, no 6, November 1981, pp 3452-3455.

[33] J.L La Cicero, DJ. Costello lind I..C. Peach, "Chnracterisucs of Ihe Hedeman II-1, 11-2 and 11-3 codes," IEEE Transactions on Communications, vol COM·29, no 6, June 1981, pp 901-908.

[34] G.V. Jacoby and R. KOSI, "Binary two-thirds rate code with full word look-ahead," IEEE Transnctions on Magnetics, vol MAG-20, September 1984, pp 709-714.

[35] G. van Rensburg and I1.C. Ferreira, "Four new runlength constrnined recording codes," Electronic Letters, vol 24, nO 17, 18 August 1988, pp 1110·1111

[36] CCJR: International Rndio Consultive Committee. "Recommendations and reports of the CCIR, 1986," vol VIII-L, Land Mobile Service, Amateur Service, Amateur SOlcllite Service, Duvrovnik, 1986, report 903.

(37) F. Swarts, II.C. Ferreira and n.R. Oosthuizen, "Renewal channel models for PSK on slowly fading Rayleigh channel," Electronic Letters, vol 25, no 22, October 1989, pp 1514-1415.

[38] S. Tsai, "Markov characterization of the HF channel," IEEE Transactions on Communication Technology, vol COM-17, no I, February 1969, pp 24-32.

[39] RD. Frithchman, "A hinary characterization using partitioned Markov chains," IEEE Transactions on Information Theory, vol IT·13, April 1967, pp221-227.

[40] Lattice Semiconductor Corporntion, "GAL handbook," 1990.

(41) Electronics World + Wireless World, August 1990.

(42) Intel Corporation, "Microcontrollcr handbook," 1990.

[43] L.II. Pollard, "Computer design and architecture," Prentice-Hall International, New Jersey, 1990.

[44] F. Gardener, "Phaselock techniques,", McGraw-Hili. 1966.

[45] RCA Solid State, "CMOS integrated circuits," 1983.

[46] Texas Instruments, "TMS32010 user's guide." 1983.

[47] TEMC 32-2, "Fail-safe telemetry system technical manual," M.L. Engineering Ltd. (Plymouth), UK, Issue 4, January 1987, pp 2-8.

(48) E. Zchnvi and J.K. Wolf, "On runlength codes," IEEE Transactions on Information 111COry, vol 34, no I, January 1988, pp 45-55.