UNIVERSITY OF ALBERTA

DIVERSITY COMBINING AND EQUALIZATION FOR INDOOR WIRELESS SYSTEMS USING GMSK MODULATION AND TIME DIVISION DUPLEXING

By

Boon Shown Yeh ©

A thesis submitted to the Faculty of Graduate Studies and Research in partial fulfillment of the requirements for the degree of Master of Science.

DEPARTMENT OF ELECTRICAL ENGINEERING

Edmonton, Alberta

Fall 1995

ISBN 0-612-06563-4

UNIVERSITY OF ALBERTA

RELEASE FORM

NAME OF AUTHOR: **Boon Shown Yeh**

TITLE OF THESIS: **Diversity Combining and Equalization for Indoor Wireless Systems Using GMSK Modulation and Time Division Duplexing**

DEGREE: **Master of Science**

YEAR THIS THESIS GRANTED: **1995**

Yeh Boon Shown

LOT 4273, Simpang 381-4
Jalan Maulana, W-8
Kuala Belait, 6680
Brunei

Date: 30 August 1995

UNIVERSITY OF ALBERTA

FACULTY OF GRADUATE STUDIES AND RESEARCH

The undersigned certify that they have read, and recommend to the Faculty of Graduate Studies and Research for acceptance, a thesis entitled **DIVERSITY COMBINING AND EQUALIZATION FOR INDOOR WIRELESS SYSTEMS USING GMSK MODULATION AND TIME DIVISION DUPLEXING** submitted by **Boon Shown Yeh** in partial fulfillment of the requirements for the degree of **Master of Science**.

W. Kr

Dr. Witold A. Krzymien, Supervisor

Paul A Goud

Dr. Paul A. Goud, Co-Supervisor

Colin G. Englefield.

Dr. Colin G. Englefield, Internal Examiner

Dr. Anup Basu, External Examiner

Date: 29 Aug. 1995

Dedication

To my parents, grandfather, sisters and Bee Choo,
for their love, support and encouragement.

In memory of my grandmother.

ABSTRACT

Multipath propagation in a radio channel causes fluctuations in signal envelope known as multipath fading, and the spreading of the received signal. Multipath fading together with intersymbol interference (ISI) resulting from the spreading effect can severely degrade the performance of digital wireless communication systems. This thesis studies the use of antenna diversity and adaptive channel equalization to improve the performance of an indoor wireless system employing GMSK modulation and time division duplexing by mitigating multipath fading and ISI, respectively.

For an indoor wireless system using time division duplexing, the performance gain due to diversity selection can be achieved for both the forward and reverse links by implementing diversity antennas at the base station only, provided that the frame duration is short and the fading rate is low. For radio channel equalization, a non-linear equalizer such as a decision feedback equalizer (DFE) is required to effectively remove the ISI. The combined use of dual antenna diversity and decision feedback equalization is investigated in this study. Four such combined diversity-equalization schemes are considered. They are, a) DFE with diversity selection driven by a received signal strength indication (RSSI) criterion, b) DFE with diversity selection driven by an error rate indication (ERRI) criterion, c) DFE with diversity selection driven by a mean squared error (MSE) criterion and d) the integration of diversity combining and decision feedback equalization through joint-parameter optimization. The performance of these schemes is evaluated through extensive computer simulation, using the Digital European Cordless Telecommunications (DECT) system as a reference. The simulation results have shown that these schemes can greatly enhance the system performance.

Acknowledgments

I would like to express my sincere gratitude to my supervisors, Dr. Witold A. Krzymien and Dr. Paul A. Goud, for introducing me to the field of Wireless Communications, and for guiding, assisting and encouraging me throughout my graduate program. In addition, I thank them for providing me with financial assistance. I am also grateful to the members of my examining committee, Dr. Colin G. Englefield and Dr. Anup Basu, for reviewing this thesis.

My special thanks are also given to my fellow students, especially Anthony Soong and Peng Mok, for many helpful discussions.

Finally, I am indebted to Telecommunications Research Laboratories (TR*Labs*) for providing me with research facilities and financial assistance for my M.Sc. program.

Table of Contents

List of Figures

List of Tables

Abbreviations

ADPCM	Adaptive differential pulse code modulation
AMPS	Advanced Mobile Phone Service
AWGN	Additive white Gaussian noise
BER	Bit error rate
CDF	Cumulative distribution function
CDMA	Code division multiple access
CEC	Commission of the European Communities
CPFSK	Continuous phase frequency shift keying
CRC	Cyclic redundancy check
CT2	Cordless Telephone 2
DCA	Dynamic channel allocation
DCS 1800	Digital Cellular System 1800
DECT	Digital European Cordless Telecommunications
DFE	Decision feedback equalizer
E_b/N_o	Ratio of energy per bit to one-sided noise power spectral density
ECC	Error control coding
ERRI	Error rate indication
ETSI	European Telecommunications Standards Institute
FDD	Frequency division duplex
FDM	Frequency division multiplex
FDMA	Frequency division multiple access
FBF	Feedback filter
FFF	Feedforward filter
FIR	Finite impulse response
FSE	Fractionally spaced equalizer
FSK	Frequency shift keying
GHz	Giga Hertz
GMSK	Gaussian minimum shift keying
GSM	Global System for Mobile communications
IS-54	Interim Standard-54
IS-95	Interim Standard-95
ISDN	Integrated services digital network
ISI	Intersymbol interference
kb/s	kilo-bit per second

LAN	Local area network
LCR	Level crossing rate
LMS	Least mean square
MAP	Maximum *a posteriori* probability
Mb/s	Mega-bit per second
MCS-L1	Mobile Communications Systems L1
MLSE	Maximum-likelihood sequence estimation
MMSE	Minimum mean squared error
MSE	Mean squared error
MSK	Minimum shift keying
NMT	Nordic Mobile Telephone
NRZ	Non-return-to-zero
ns	nano-second
PCN	Personal Communication Network
PDC	Personal Digital Cellular
PDF	Probability density function
PHP	Personal Handy Phone
RF	Radio frequency
RLS	Recursive least squares
rms	Root mean square
RSSI	Received signal strength indication
SNR	Signal to noise ratio
TDD	Time division duplex
TDM	Time division multiplex
TDMA	Time division multiple access
UMTS	Universal Mobile Telecommunication System
WLLAN	Wireless local area network
ZF	Zero-forcing

1. INTRODUCTION

1.1. Wireless Personal Communications

Wireless personal communications, which have gained widespread public acceptance in recent years, continue to experience rapid growth, with new applications emerging continuously. The demand for wireless communications has led to the evolution of application and environment-specific wireless technologies such as cellular radio systems for high mobility vehicular terminals, cordless telephony for portables (including slow moving handsets), and wireless local area networks for high-speed applications. With continuous advances in communication theory and in microelectronics technology, the evolution of wireless systems from analog to digital is also occurring.

One of the old constraints encountered in wireless communications is the limited frequency spectrum availability. In the 1940s, the cellular concept was proposed as a means of increasing the number of communication channels through frequency reuse, by sectoring the coverage area into small cells. Though the concept is old, public cellular systems are less than 20 years old. Research on analog cellular system design initiated in the 1970s led to the realization of first-generation cellular systems including Nordic Mobile Telephone (NMT) in Scandinavian countries, Mobile Communications Systems (MCS-L1) in Japan, and Advanced Mobile Phone Service (AMPS) in the United States. These first-generation cellular systems use analog frequency modulation for speech transmission, frequency shift keying for signaling and frequency division multiple access (FDMA). The analog systems soon failed to accommodate the increased number of subscribers and to support new services such as the transmission of data for fax, e-mail, etc. As a result, second-generation cellular systems were launched in the late 1980s and early 1990s. They are the Pan-European digital cellular system known as Global System for Mobile communications (GSM), the North American Interim Standard-54 (IS-54) and the Japanese Personal Digital Cellular (PDC). The second-generation cellular systems are based on digital modulation schemes and a hybrid of time division and frequency division multiple access (TDMA/FDMA). They offer many advantages over the existing analog systems including a larger system capacity, compatibility with the evolving digital wireline network, flexibility for mixed voice/data communication, encryption for communication privacy, etc. In contrast to its analog counterparts, GSM also supports intercountry roaming within Europe. Recently, Digital Cellular System 1800 (DCS 1800) operating near 1800 MHz has been adopted as a standard for Personal Communication Network (PCN) in Europe [1]; it is a derivative of GSM and is meant for micro or pico-cellular cordless application. In North America, the deployment of Interim Standard-95 (IS-95) systems using code division multiple access (CDMA) as the principle access technique is expected in 1995 [2].

Today, the demand for cordless telephony is also increasing rapidly. Unlike cellular systems, cordless systems are optimized for high user capacity in quasi-static indoor environments. Also, the speech quality of cordless telephony is required to be comparable to that of a wireline telephone, whereas the requirement is less stringent for cellular mobile systems. Similar to cellular radio, the first-generation cordless systems use analog technology. Technological advances stimulated the development of the second-generation digital cordless systems such as Cordless Telephone 2 (CT2) and Digital European Cordless Telecommunications (DECT) in Europe and Personal Handy Phone (PHP) in Japan, respectively [2]. The multiple access duplex operation of these three digital cordless systems is achieved through the combination of frequency division and time division multiple access, and time division duplex (TDD). They are intended not only for home and office use but also for public access through telepoints. Also, in addition to voice services, they also support cordless data transmission.

Together with wireless telephony, wireless data communications have created a new frontier for modern communications. Generally, wireless data communications can be divided into two categories, namely low-speed wide area access and high-speed local area access. The former is known as mobile data whereas the latter is usually recognized as wireless local area networks (WLLANs) [3]. Mobile radio data serve a variety of short-message applications such as paging, whereas WLLANs support high-speed applications such as long file transfers and printing tasks. WLLANs offer many benefits; they include the flexibility of fast reconfiguration or relocation of portable terminals, and the provision of valuable services in areas where wiring or re-wiring may be difficult.

In addition to the quest for more sophisticated services such as multi-media services, and multi-operator and multi-environment operations, there is a desire to establish a more widely-based standard which may allow manufacturers of proprietary equipment to address much larger potential markets. As a result, ambitious research on a third-generation mobile system known as Universal Mobile Telecommunication System (UMTS) has been launched by the Commission of the European Communities (CEC).

1.2. Project Motivation

Besides exploring a new generation of mobile communication technology, there is also a great interest in exploiting the potential of existing standards such as DECT and GSM. The goal is to further enhance the existing systems in order to enable them to support new applications such as wireless video transmission and mobile computing, and possibly co-exist with the next generation mobile communication systems.

DECT is mainly intended to support voice services in a quasi-static environment. For wireless speech telephony, the bit error rate (BER) of the system should be maintained at 10^{-3} or better. However, a BER of at least 10^{-4} is desirable for wireless video transmission due to the high compression video coding algorithm. Without using additional signal processing techniques, the required BER of 10^{-4} can hardly be reached in a dispersive environment. Furthermore, it has been shown that the performance of a DECT system degrades significantly as the time dispersion of the channel increases [4], which may occur in large indoor environments such as warehouses, airport terminals and large factories. Therefore, it is necessary to investigate and incorporate a suitable performance enhancing technique with the current DECT standard such that its range of applications can be broadened.

The following section presents the DECT standard. The impairments of a radio channel which cause performance degradation in wireless communications are discussed in Section 1.4. Section 1.5 introduces some of the techniques that can be used to counteract the channel impairments. Finally, an overview of the research work undertaken in this project is presented in Section 1.6.

1.3. The DECT Standard

The DECT standard, developed by the European Telecommunications Standards Institute (ETSI), is the Pan-European standard for cordless access to a telecommunication network. It is designed as a flexible interface to provide both voice and data services in indoor and outdoor environments with high user densities.

DECT operates in the 1880-1900 MHz band with ten radio carriers spaced at 1.728 MHz. The in-slot data rate is 1.152 Mb/s. It uses a hybrid multiple access scheme of TDMA and FDMA to cope with high user densities, and a time division duplex (TDD) operation. The TDM/TDD architecture offers several advantages over the traditional frequency division multiplexing (FDM) and frequency division duplexing (FDD), which include relatively simple transceiver filter design (as compared to a more complex and expensive duplexing filter required for FDD) and variable-rate data transmission. The digitized speech signal or data are modulated onto a carrier using constant envelope and spectrally efficient Gaussian minimum shift keying (GMSK) with a bandwidth bit interval product (*BT*) of 0.5. A radio carrier is divided in the time domain into TDMA frames. Each TDMA frame with a duration of 10 ms is subdivided into 12 pairs of time-slots. Two associated time-slots in a frame, separated by 5 ms, are used for the forward link (base station to portable) and the reverse link (portable to base station) transmissions in an alternating mode [5]. A DECT time-slot, which has a duration of 480 bit intervals, is shown in Figure 1.1.

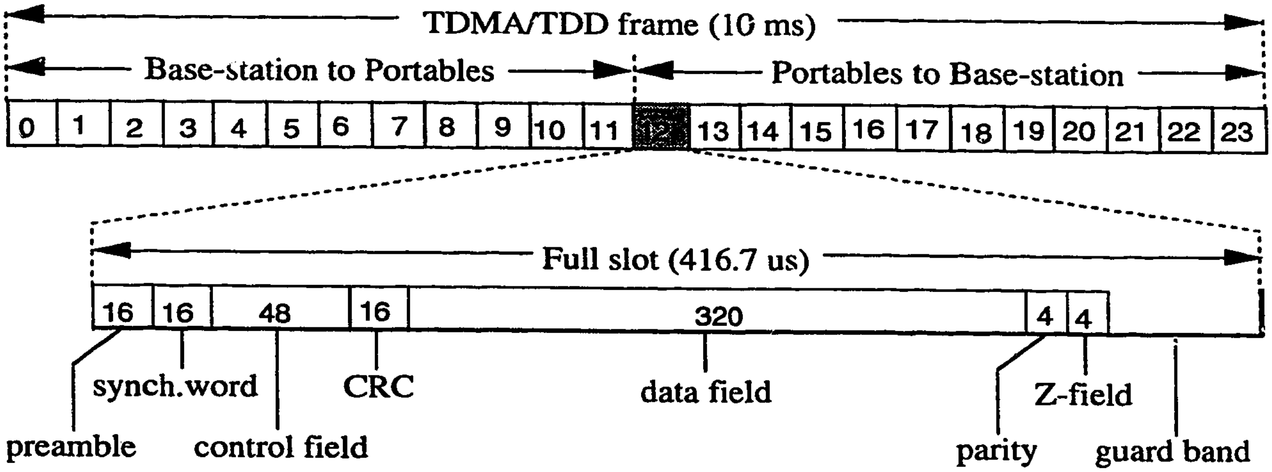

Fig. 1.1. The DECT frame and slot structure.

Each time-slot contains a 32-bit synchronization-field (S-field) for clock and slot synchronization, a 64-bit A-field which carries signaling and control information together with a 16-bit cyclic redundancy check (CRC) code for error detection, a 324-bit B-field with the first 320 bits for user data and the last 4 bits for parity check, a 4-bit Z-field for detecting sliding collisions (a scenario that occurs when portable terminals lose synchronization and transmit at the time intervals originally allocated to other portable terminals), and a guard space of 56 bit intervals [6, 7].

The telephony services use 32 kb/s adaptive differential pulse code modulation (ADPCM) to provide high-quality speech. DECT uses duplex pairs of time-slots to achieve a transmission data rate of 32 kb/s for basic telephony applications. However, it has the capability of providing higher throughputs for integrated services digital network (ISDN) or local area network (LAN) applications by using multiple time-slots, which may include both the forward and reverse link slots, for transmission [8].

With dynamic channel allocation (DCA), the need for frequency planning is obviated in DECT. Based upon availability, a suitable channel is taken from the pool when needed and returned to the pool after a call has been completed. As a result of this adaptive channel reuse, DECT can support a high traffic capacity.

1.4. Radio Propagation Channel

The radio propagation channel contributes to many problems and limitations in wireless communications. One of the main features of a radio channel is multipath propagation, in which energy travels from the transmitter to the receiver through a medium that contains several paths with different time delays and attenuations. This gives rise to a highly complex channel which can severely distort the information it carries. On the other hand, without multipath propagation, wireless communications may become inefficient, if not impossible; it maintains the continuity of a transmission link, which would otherwise be broken, by allowing reflected, diffracted and scattered waves to reach a receiver which is not in a line of sight of the transmitter. Figure 1.2 illustrates the mechanism for multipath propagation in urban areas.

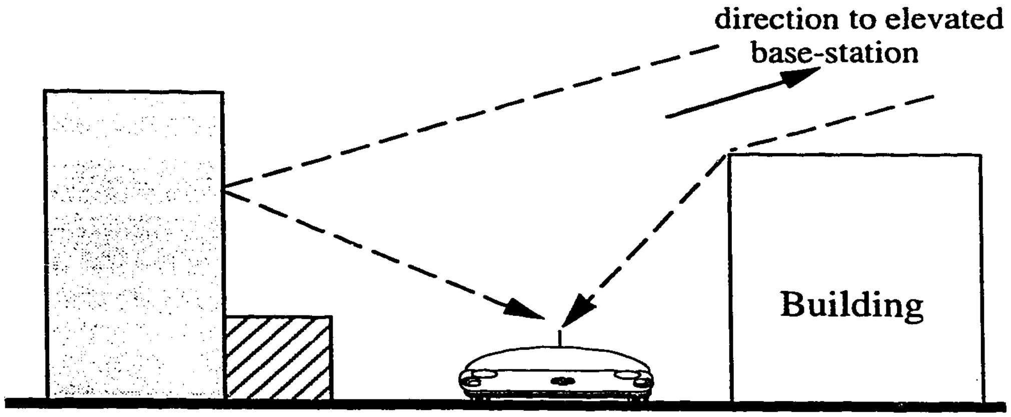

Fig. 1.2. Multipath propagation in urban areas.

The characteristics of a radio channel which have detrimental effects on wireless communications can generally be classified into the following categories:

- *Propagation path loss:* Mobile radio signals are subjected to path loss effects inherent in both atmospheric and terrestrial propagation. Terrestrial losses are affected by the general topography of the coverage area as well as the height of a mobile antenna. Additional losses such as building penetration losses are encountered by radio signals traveling into buildings. Within buildings, attenuation factors for walls, floors and other partitions also contribute to the losses. In general, an indoor channel exhibits a much greater path loss than an outdoor channel. Path loss information is essential in determining the link power budget of a wireless system for a given coverage area.
- *Delay spread:* If the transmission medium contains several paths of different lengths, radio signals traversing through it arrive at a receiver with different delays. This results in the spreading or smearing of signals. In digital transmission, this spreading phenomenon causes each symbol to overlap with adjacent symbols, producing intersymbol interference (ISI) and thus setting a limit on the transmission rate of this digital radio channel. The root mean square (rms) delay spread of a channel is environment dependent. Numerous works quoted in [9] have shown that it may vary from 20 ns in small office buildings to 300 ns in factory environments. For the outdoor mobile radio channel, it is typically in the range of 10 μs to 30 μs. When the rms delay spread of a channel is greater than the

inverse of the signal bandwidth, the channel is said to be time dispersive. In a time dispersive channel, different frequency components of a signal experience uncorrelated attenuation. For this reason, a time dispersive channel is also known as a frequency selective channel.

- *Fading:* There are two types of fading in a mobile radio environment, namely long term fading and short term fading. Long term fading is caused by shadowing effects of large geographical or man-made obstacles located between the transmitting and receiving ends of a communication link. Generally, it follows a log-normal distribution. Short term fading is a result of multipath propagation in which the superposition of component waves leads to either constructive or destructive interference, depending upon the relative phase shifts of the component waves. When a direct path exists in the propagation medium, short term fading can be approximated by a Rician distribution. However, in mobile communications, a line-of-sight path seldom exists and signals normally arrive at the receiver via indirect paths. When a direct path is absent, short term fading can generally be characterized by a Rayleigh distribution. Another important parameter is the rate of short-term fading. It is related to the Doppler effects caused by the motion of the mobile or portable terminals and of scatterers in the surrounding area. Figure 1.3 illustrates the fading phenomenon of a mobile radio channel. Deep fades of 40 dB below the mean signal strength are not uncommon in a typical urban environment.

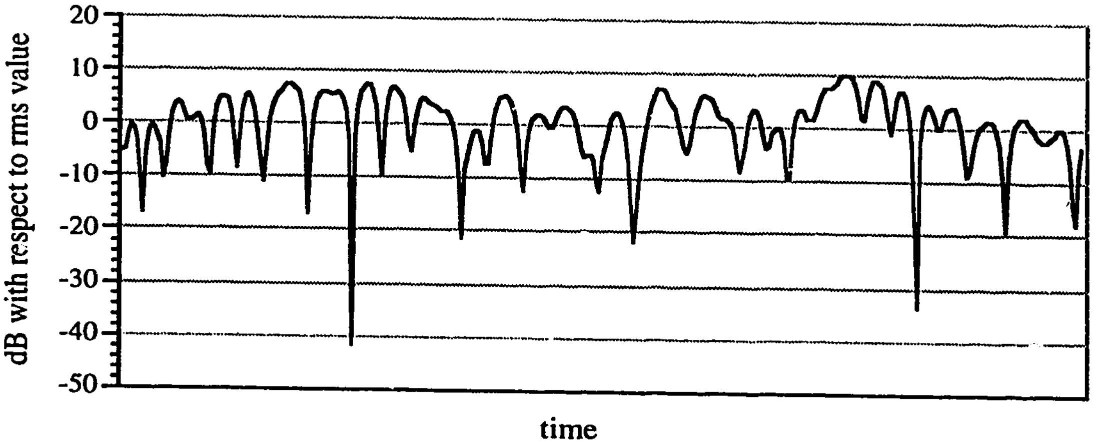

Fig. 1.3. Variation of received signal envelope in a fading channel.

- *Noise:* A radio signal in a given channel is susceptible to many sorts of noise. External noise is the predominant noise source for wireless communications. It includes galactic noise, automobile ignition noise and noise generated by lightning, electric motors, etc. Thermal noise in the receiver itself also contributes to the total noise in the system. The combination of noise from both external and internal sources is generally modeled as additive white Gaussian noise (AWGN). The ratio of the signal strength to the overall noise level is called the signal-to-noise ratio (SNR). It is a basic measure of signal quality.

1.5. Countermeasures to Channel Distortion

Impairments in radio channels can cause severe performance degradation in wireless systems. However, there are different techniques that can be employed to effectively counteract the distortion introduced by multipath propagation. Among them are diversity techniques, channel equalization and error control coding.

1.5.1. Diversity

Diversity is an efficient method for combating fading. To reduce long term fading due to shadowing, two or more antennas located at different sites are used to transmit the same signal. This kind of large scale space-diversity is known as macroscopic diversity. On the other hand, microscopic diversity, which requires two or more antennas or frequencies at the same antenna site (co-located site), is used to counteract short term or multipath fading.

There are six microscopic diversity schemes, namely space, frequency, time, polarization, field component and angle diversity. The basic principle of these schemes is based on the utilization of redundantly available component signals with low correlation among their fading. Space diversity employs two or more antennas which are separated from each other by a certain distance such that each antenna or diversity branch receives an independently fading signal. In frequency diversity, two different frequency carriers are used to transmit the same information. The signals fade independently, provided that the frequency separation between the carriers exceeds the coherence bandwidth by several times [10]. The coherence bandwidth is defined as the frequency separation required to obtain a correlation of 0.5 between the two signals. Similarly, time separation also provides a form of diversity. Time diversity means transmitting identical messages at different instants to achieve uncorrelated fading signals at the receiver. However, it can also be achieved at the receiver end by identifying different multipath components as distinct echoes of the signal separated in time, provided that a reference time exists. This special form of receiver is known as the RAKE receiver [11]. Polarization diversity makes use of the orthogonality of polarized electromagnetic wave components. These polarized waves, with either vertical or horizontal electric fields, are transmitted and received by polarized antennas. Another diversity scheme which exploits the properties of electromagnetic waves is field-component diversity. This form of diversity can be achieved by employing an antenna system which is capable of separating and combining electric and magnetic fields appropriately [12]. In angle diversity, directional antennas can be used to track signal components with different angles of arrival as a result of multipath propagation.

Signals from diversity branches may be processed by different combining techniques. The four main diversity combining techniques are selection diversity, switched diversity, maximum ratio combining and equal-gain combining [12]. In selection diversity, received signals are continuously monitored and the best signal among the diversity branches is selected. This method may require a receiver for each diversity branch. However, only one receiver is needed for switched combining diversity, in which a diversity signal is selected until it falls below a given threshold. Then the receiver is switched to another branch regardless of whether its signal level is above or below the threshold. Both selection diversity and switched diversity do not exploit the full benefit of diversity because they utilize the signal from only one diversity branch at a time. To gain the complete advantage, all diversity signals must be combined vectorially after proper phase and time-delay corrections for multipath distortions. In its simplest form, an equal gain is applied to all the component signals. This is known as equal-gain combining. However, the best combining technique is maximum ratio combining [11]. In this scheme, individual diversity signals are weighted in proportion to their own signal-to-noise ratio

(SNR) and summed to achieve a maximum output SNR.

1.5.2. Channel equalization

The performance of a digital wireless system is often degraded by intersymbol interference (ISI) resulting from the propagation of a signal through a multipath channel, and in some cases also from the modulation scheme itself. To optimize the system performance, a passband or a baseband equalizer may be used at the receiver to mitigate the effect of ISI.

In general, there are two types of equalization technique: linear equalization and non-linear equalization. A linear equalizer may simply be implemented as a finite-duration impulse response (FIR) filter. However, it is not suitable for mobile communications because it is incapable of compensating for severe channel amplitude distortion without significant noise enhancement [13]. In this case, robust non-linear equalizers, such as a decision-feedback equalizer (DFE) can be used to remove the ISI.

To equalize an unknown and time-varying channel, the equalizer is required to be adaptive or self-optimized. Before the transmission of real information data, a sequence of known training symbols is transmitted and its distorted version is received to acquire the information of the channel characteristics. The initial equalizer parameters are derived in this process. After the equalizer is trained, it is required to track the time-varying channel and to continually update its parameters ın a decision-directed mode. Several well-known algorithms, such as the recursive least squares (RLS) algorithm, can be used for equalizer adaptation. For a given algorithm, it is important to determine the length of the training sequence so that the equalizer will converge during the training period. However, as it is also desirable to keep the ratio of training symbols to data symbols low in the TDMA frame, a fast converging algorithm is normally implemented to train the equalizer.

1.5.3. Error control coding

Shannon's channel coding theorem states that it is in principle possible to reconstruct transmitted information with an error probability as small as desired, provided that the data rate over the channel is less than the channel capacity. However, it does not provide an algorithm to search for the appropriate coding scheme for an information source and a channel. Thus, a lot of effort has been devoted in the past four decades to finding "good codes".

Error control coding (ECC) may be used to increase the reliability of digital communication systems by incorporating some form of structured redundancy with the information sequence to facilitate error detection and correction. Historically, there are two types of code, namely block codes and convolutional codes.

In block coding, the encoder receives a k-bit message and adds n-k redundant bits that are algebraically related to the message bits to generate an n-bit code word, where the ratio of k to n is known as the code rate. This requires the encoder to buffer an entire message block before producing the associated code word. Some block codes are meant for both error detection and correction while others are meant solely for error detection.

In convolutional coding, the encoder accepts message bits in a serial manner and generates a sequence of encoded bits at a higher rate. The encoding operation may be considered as the discrete time convolution of the input sequence with the impulse response of the encoder. The encoder consists of a multi-stage shift-register with prescribed connections to modulo-2 adders, and a multiplexer that takes the outputs of the adders to produce a serial output.

The addition of redundant bits to data increases the overall transmission bit-rate and thus increases the bandwidth of the transmitted signal. Bandwidth expansion is undesirable when coding is applied to a bandwidth constrained channel. However, by combining coding and modulation as a single function, the performance gain due to coding can be achieved without expanding the signal bandwidth. This approach is known as coded modulation in which coding is used to impose certain patterns on the transmitted signal such that the minimum Euclidean distance between pairs of coded signals is maximized.

In general, coding and coded modulation may improve system performance. However, the performance gain due to coding may be marginal in a quasi-static indoor channel where long deep fades can occur unless a suitable bit-interleaving scheme with sufficient depth is used.

1.6. Thesis Overview

The objective of this research project is to investigate the application of dual antenna diversity and adaptive channel equalization techniques to an indoor wireless communication system employing GMSK modulation and time division duplexing. The performance of these techniques is evaluated through extensive computer simulations, using DECT as the reference system. The study is presented in the following order.

- Chapter 2 details the modeling of an indoor radio channel and a GMSK transceiver. It includes discussion of the channel characteristics, and of the modulation and demodulation of GMSK signals.
- Chapter 3 reviews the principles of channel equalization and diversity combining. Issues regarding equalizer structure and adaptation, and antenna diversity together with related combining methods are discussed.
- Chapter 4 presents four performance enhancing schemes, which combine dual antenna diversity and decision feedback equalization. *Schemes I, II*, and *III* use decision feedback equalizers with diversity selection driven by a received signal strength indication (RSSI), error rate indication (ERRI) and mean squared error (MSE) criteria, respectively, whereas *Scheme IV* integrates diversity combining and decision feedback equalization into a single unit through joint-parameter optimization. These schemes are designed so that they are applicable to DECT or DECT-type systems.
- Chapter 5 details the simulation models of the DECT baseline system and those associated with the different diversity and equalization techniques. Simulation results are also presented and analyzed in this chapter.
- Finally, Chapter 6 presents conclusions drawn from the results of the research and a discussion of possible future work.

2. CHANNEL MODEL AND GMSK TRANSCEIVER

This chapter is concerned with the modeling of an indoor radio channel and a GMSK transceiver. A general channel model used to emulate a multipath fading channel is defined. Related to the channel model is a Rayleigh fading simulator. The features of the fading simulator are discussed and its performance is verified via computer simulation. Following the description of channel modeling is a discussion of GMSK modulation. A model of a GMSK modulator is outlined. Finally, a simple one-bit differential detector for the demodulation of GMSK signal is discussed.

2.1. Indoor Radio Channel

Multipath propagation in a radio channel results in signal spreading and multipath fading. This distortion severely affects the performance of wireless communication systems. Hence, it is important to characterize the radio channel in detail for successful wireless communication system design.

2.1.1. Modeling of the radio channel

Extensive work on the characterization and modeling of indoor radio propagation channels has been published [14, 15, 16]. Numerous other references can also be found in [9]. A multipath channel can be fully described by its time and space varying impulse response. However, only the time varying factor is considered here. Thus the channel can be conveniently modeled by a baseband complex finite impulse response (FIR) filter as follows:

$$h(t) = \sum_{i=1}^{N} a_i(t) exp\{j\theta_i(t)\}\delta(t-\tau_i) \qquad (2.1)$$

where,

N is the total number of discrete paths in the model,

δ is the Kronecker delta function,

$a_i(t)$ is the time-varying real gain of the i-th path,

τ_i is the propagation delay of the i-th path, and

$\theta_i(t)$ is the time-varying random phase shift associated with the i-th discrete path.

Each discrete path in (2.1) represents a combination of many time-varying propagation paths which cannot be further resolved into separate distinct identities. The vectorial addition of signals from component paths results in a signal with a fluctuating envelope. This fluctuation can be modeled by the variation of the gain of each discrete path, $a_i(t)$, with time. In the absence of a direct path, the envelope variation of the signal is normally characterized as Rayleigh fading. If a direct path exists, it is generally characterized as Rician fading. The probability density functions (PDFs) of Rayleigh and Rician distributions are given by (2.2.a) and (2.2.b), respectively:

$$p_R(r) = \frac{r}{\sigma^2} exp\left\{-\frac{r^2}{2\sigma^2}\right\} \qquad r \geq 0 \qquad (2.2a)$$

$$p_R(r) = \frac{r}{\sigma^2} exp\left\{-\frac{r^2+s^2}{2\sigma^2}\right\} I_o\left(\frac{rs}{\sigma^2}\right) \qquad r \geq 0 \qquad (2.2b)$$

where σ^2 is the variance of the two independent Gaussian variables associated with the distributions, s is the non centrality parameter of the Rician distribution and I_o is the zero-th order modified Bessel function of the first kind. Other distributions, such as Nakagami and Weibull distributions, have also been used in the literature.

Associated with each unresolvable fading path is a random phase shift, $\theta_i(t)$. A Rayleigh fading signal can be expressed in terms of its quadrature components, a_I and a_Q respectively, which are zero-mean Gaussian random variables. The fading phase of such a signal is defined as

$$\theta = tan^{-1}\left(\frac{a_Q}{a_I}\right) \qquad (2.3)$$

which is a uniformly distributed random process with a PDF given by

$$p_{\Theta}(\theta)=\frac{1}{2\pi}, \qquad 0\leq\theta<2\pi \tag{2.4}$$

In a Rician fading path, the fading signal consists of two components: the direct component, which is an amplitude-scaled version of the transmitted signal, and a diffused Rayleigh-faded component. Therefore, the quadrature components representing the fading signal are non-zero mean Gaussian random variables. Consequently, the random phase of the signal is no longer uniformly distributed and has a PDF defined as [17]

$$p_{\Theta}(\theta)=\frac{exp(-K)}{2\pi}+\frac{\sqrt{K}\,cos\,\theta\, exp\left(-K\, sin^2\theta\right)}{2\sqrt{\pi}}\left(2-erfc\left(\sqrt{K}\,cos\,\theta\right)\right), \qquad |\theta|\leq\pi \tag{2.5}$$

where K is the power ratio of the direct component to the diffused Rayleigh-faded component and $erfc(x)$ is the complementary error function given by

$$erfc(x)=\frac{2}{\sqrt{\pi}}\int_{x}^{\infty}exp\left(-t^2\right)dt$$

Other important channel issues include the rms delay spread, mean excess delay and average power delay profile of a channel.

A radio channel consists of many propagation paths with different time delays associated with their respective path lengths. This results in a time dispersive channel which can be characterized in terms of its mean excess delay (τ_d) and rms delay spread (σ_τ). For a discrete time channel model, the mean excess delay and rms delay spread are defined as

$$\tau_d=\frac{\sum_i(\tau_i P_i)}{\sum_i P_i} \tag{2.6}$$

$$\sigma_\tau = \sqrt{\frac{\sum_i \left(\tau_i{}^2 P_i\right)}{\sum_i P_i} - \tau_d{}^2} \tag{2.7}$$

where P_i and τ_i are the power and time delay of the i-th discrete path, respectively.

The power delay profile of a channel determines the distribution of average power among the resolvable paths. For simplicity, a radio channel is often modeled by two discrete paths having the same average power. However, a channel model consisting of several discrete paths with an exponentially decaying average power appears to be a more realistic model for indoor environments. It has also been shown that, within the same impulse response profile, the multipath components are likely to be correlated. The degree of amplitude correlation between any two paths depends upon their relative time delay. However, this kind of correlation has not yet been well-characterized quantitatively. Thus it is also common practice in channel modeling to assume that the discrete multipath components are uncorrelated. Another related issue which is difficult to characterize is the distribution of the arrival time sequence. The standard Poisson model, which is used to postulate the path arrival time, may be inadequate, since scatterers inside a building causing multipath dispersion are not purely randomly located [9].

A combination of noise from both internal and external sources is modeled as an AWGN voltage, $n(t)$. White Gaussian noise is a stochastic process with a flat power spectral density over the entire frequency range and a probability distribution function defined as

$$p_n(n) = \frac{1}{\sqrt{2\pi}\sigma_n} exp\left\{-\frac{n^2}{2\sigma_n{}^2}\right\} \tag{2.8}$$

where $\sigma_n{}^2$ is the variance of the noise. For narrowband systems, it is convenient to represent the white noise in terms of its quadrature components;

$$n(t) = n_c(t)cos(2\pi f_c t) + n_s(t)sin(2\pi f c t) \tag{2.9}$$

where f_c is the carrier frequency. This description assumes that an ideal bandpass filter, which introduces negligible distortion to the desired signal, is used at the receiver to eliminate only the noise frequency components outside the passband. The noise resulting from ideal-bandpass filtering is called bandpass white noise; it has the power spectral density depicted in Figure 2.1.

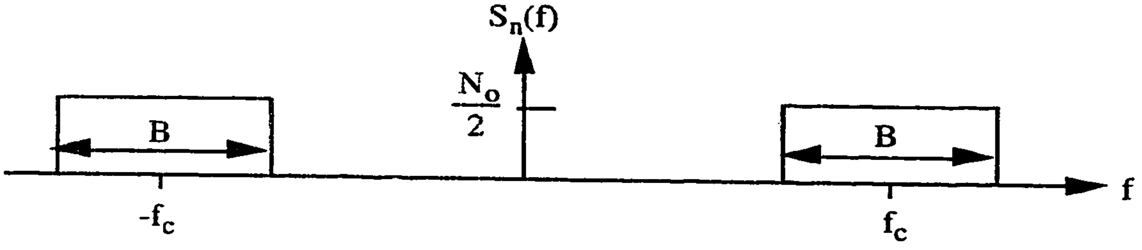

Fig. 2.1. Power spectrum of a bandpass white noise.

An N-path Rayleigh fading channel model based on (2.1) is illustrated in Figure 2.2. In computer simulations, the average gain and time delay of the first path can be arbitrarily set to unity and zero, respectively. Then, the relative time delays and average gains of the remaining paths are adjusted accordingly for a given power delay profile and rms delay spread. Subsequently, fading is introduced by a Rayleigh fader to each of the paths in the profile. Finally, an additive white Gaussian noise is added to the channel output.

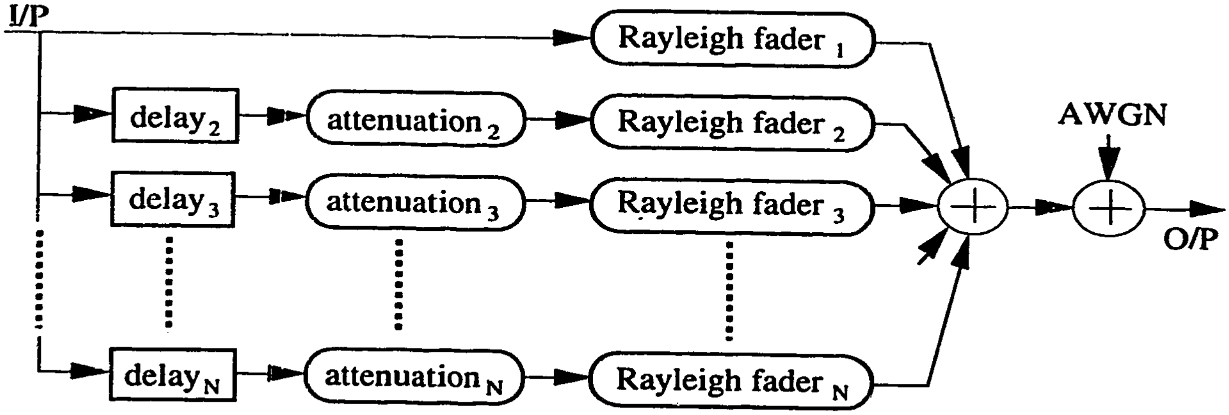

Fig. 2.2. N-path Rayleigh fading channel model.

The first path in the channel model can also be modeled as a Rician fading path to represent the existence of a direct path in the communication link.

2.1.2. Realization of a Rayleigh fader

A Rayleigh fading waveform can be generated by two different approaches. One is the offset oscillator method, which involves the summation of a number of shifted waves to produce two independent Gaussian processes. The other is the Gaussian noise filtering solution in which two independent Gaussian noise generators are required, and the output of each generator is filtered so as to produce a desired power spectrum. However, for computer simulations, the offset oscillator approach is apparently easier to implement than the Gaussian noise filtering method. The realization of the offset oscillator method based on Jakes' analysis is discussed below [18].

The received signal at a mobile is assumed to consist of M horizontally traveling plane waves with uniformly distributed angles of arrival. Furthermore, it is also assumed that the total incoming power is distributed evenly among the M waves and that the receiving antenna is omni-directional. Setting the total incoming power to unity, the equivalent lowpass complex envelope of the received signal, $r(t)$, can be expressed as

$$r(t) = \frac{1}{\sqrt{M}} \sum_{i=1}^{M} exp\{j(\omega_D t \cos\alpha_i + \phi_i)\} \tag{2.10a}$$

$$\omega_D = 2\pi\frac{V}{\lambda}, \qquad \alpha_i = 2\pi\frac{i}{M} \tag{2.10b}$$

where,

ω_D is the maximum Doppler shift in radians per second,

α_i is the angle of arrival of the i-th wave,

ϕ_i is the uniformly distributed random phase-shift of the i-th wave,

V is the speed of the mobile, and

λ is the wavelength of the carrier frequency.

If M is large enough, the Central Limit Theorem can be used to conclude that the envelope of $r(t)$ is Rayleigh distributed.

Based upon (2.10) and the above assumptions, Jakes realized a Rayleigh fading simulator as shown in Figure 2.3 [18]. It consists of M_o+1 offset oscillators, with the first M_o of which having unity gains and frequencies equal to the Doppler shifts of $\omega_D cos\alpha_i$, for $i = 1,2,...M_o$ and $M_o = (M/2-1)/2$, while the remaining one having a gain and frequency equal to $1/\sqrt{2}$ and ω_D respectively. Connected to the M_o oscillators are amplifiers which have gains equal to $2cos\beta_i$ and $2sin\beta_i$. The oscillator associated with the frequency of ω_D is connected to a pair of amplifiers having gains equal to $2sin\alpha$ and $2cos\alpha$.

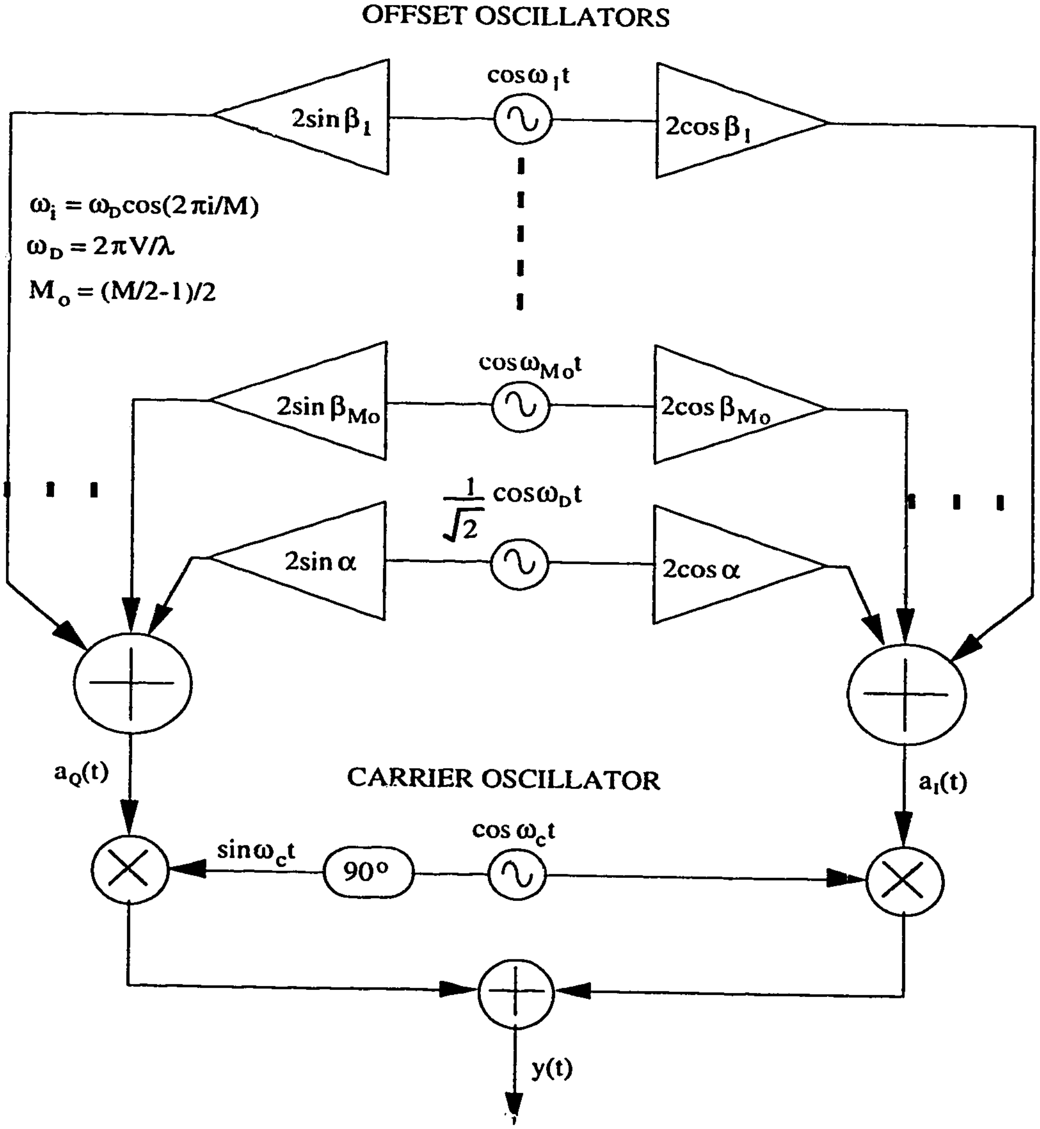

Fig. 2.3. A Rayleigh fading simulator.

The outputs of the individual oscillators, after being multiplied by appropriate gain factors, are summed to produce the in-phase ($a_I(t)$) and quadrature ($a_Q(t)$) components defined as follows:

$$a_I(t) = 2\sum_{i=1}^{M_o} \cos\beta_i \cos\omega_i t + \sqrt{2}\cos\alpha\cos\omega_D t \tag{2.11}$$

$$a_Q(t) = 2\sum_{i=1}^{M_o} \sin\beta_i \cos\omega_i t + \sqrt{2}\sin\alpha\cos\omega_D t \tag{2.12}$$

The two components are approximately Gaussian distributed random processes when M_o is greater than 8. In this case, the envelope of the composite output signal $y(t)$ approximates Rayleigh fading. Also, the random phase of $y(t)$ is uniformly distributed from zero to 2π provided the conditions $<a_I^2> \approx <a_Q^2>$ and $<a_I a_Q> \approx 0$ are satisfied. The notation < > denotes time averaging. It can be shown that by choosing $\alpha = 0$ and $\beta_i = \pi i/(M_o+1)$, $<a_I a_Q> = 0$, $<a_I^2> = M_o$ and $<a_Q^2> = M_o+1$ are obtained.

Jakes has also suggested that this technique may be extended to provide up to M_o independently fading signals by giving the i-th oscillator an additional phase shift of γ_{ij} where j is the index for the independent signals, and γ_{ij} may have the following value.

$$\gamma_{ij} = \frac{2\pi\left(j - 1 + \frac{i}{2}\right)}{M_o + 1}, \qquad i = 1, 2, \ldots, M_o \tag{2.13}$$

However, computer simulation results presented in the next section show that this approach may provide signals with envelope cross-correlations of varying degrees, depending on the amount of additional phase shifts introduced. As is demonstrated in the next section, the simplest way to produce uncorrelated fading signals is to ensure that each fading signal has a different initial time shift.

2.1.3. Performance evaluation of a Rayleigh fader

The performance of the Rayleigh fader shown in Figure 2.3 has been evaluated by computer simulation. A C-program for the Rayleigh simulator has been written. A total of 26 oscillators are used in the fading simulator. In a digital implementation, the oscillators' time progresses in a discrete manner. In other words, the variable t shown in Figure 2.3 is represented by nT_s for $n = 1, 2, \ldots$, where T_S is the sampling interval. The variances of the two random variables, $a_I(nT_S)$ and $a_Q(nT_S)$, are normalized to unity. The carrier frequency f_c is set to zero for baseband simulations. A maximum Doppler shift of 5 Hz is used. Some basic characteristics investigated include the cumulative distribution functions (CDFs) of the envelope and phase of the fading signal $y(t)$, the level crossing rate of the fading envelope, and the RF power spectrum.

The CDFs of the envelope and phase of the fading signal are examined via simulation using the C-program. Two key parameters used in this simulation are the sampling rate and the time duration over which the statistics are calculated. The envelope of a Rayleigh fading signal is quite frequently plagued by deep nulls of relatively short duration. Therefore, the sampling interval should be small enough to ensure that fades of a certain depth are in fact sampled. The time duration over which the signal samples are taken should be sufficiently long in order to obtain reliable estimates of typical fading characteristics. Hence, the sampling rate is set at 10^4 samples per second and the statistics of the fading envelope and phase are computed over a duration of 100 seconds. Figures 2.4 and 2.5 show the comparisons between the simulated CDFs and theoretical CDFs of the phase and envelope of the fading signal, respectively. The simulated results show good agreement with the expected distributions. The random phase follows a uniform distribution with the CDF defined in (2.14), whereas the fading envelope is Rayleigh distributed with the CDF given by (2.15).

$$F_\Phi(\phi) = \frac{1}{2\pi}\phi, \qquad 0 \le \phi < 2\pi \tag{2.14}$$

$$F_R(r) = 1 - exp\left\{-\frac{r^2}{2\sigma^2}\right\} \qquad (2.15)$$

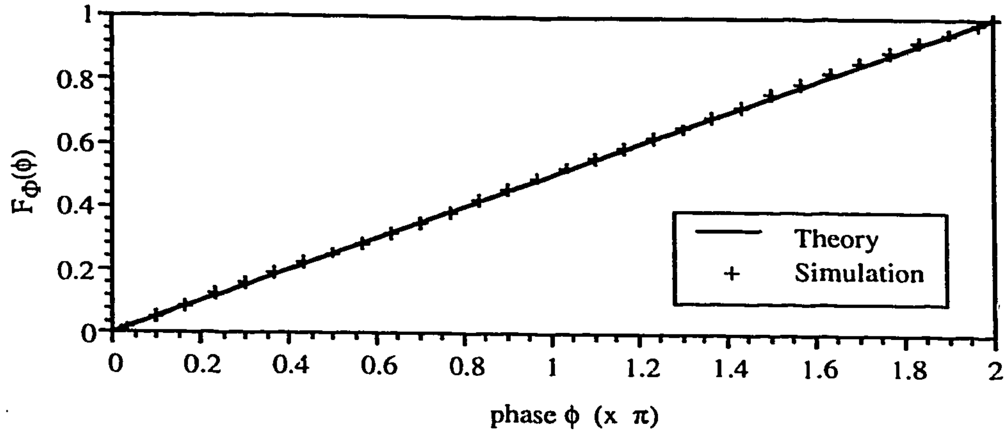

Fig. 2.4. CDF of the phase of a Rayleigh fading signal.

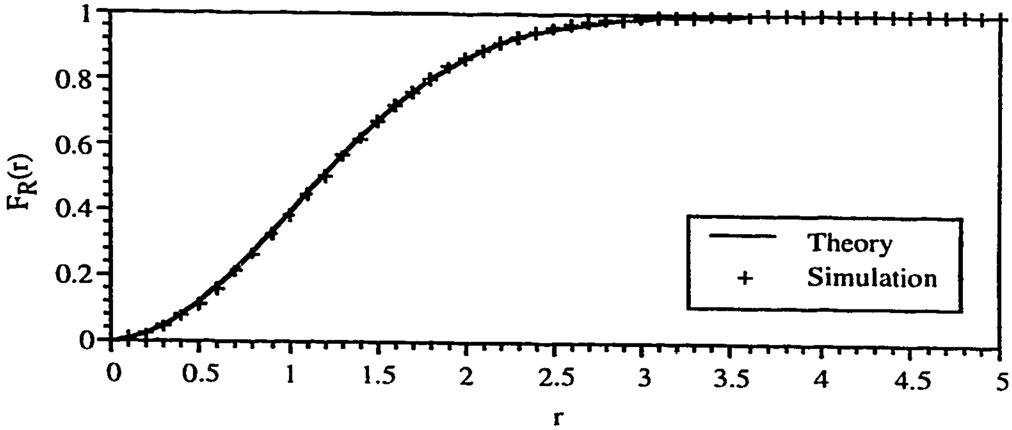

Fig. 2.5. CDF of the envelope of a Rayleigh fading signal.

The Rayleigh CDF defined in (2.15) is a first order statistic which provides information such as the overall percentage of time for which the envelope lies below a specified level. However, it gives no indication of the frequency at which fades of any depth occur. The level crossing rate (LCR), which is a second order statistic, is used to describe the expected rate at which the envelope crosses a specified level in a given direction. It can be shown that the theoretical LCR of a Rayleigh fading envelope is given by [19]

$$N_A = \sqrt{\frac{\pi A^2}{\sigma^2}} f_D \, exp\left\{-\frac{A^2}{2\sigma^2}\right\} \tag{2.16}$$

where,

A is a specified level at which the crossing rate is determined,

$2\sigma^2$ is the average power of the fading signal with σ^2 normalized to unity, and

f_D is the maximum Doppler shift.

A comparison of the LCR of a simulated Rayleigh fading signal with the theoretical LCR is shown in Figure 2.6. The sampling rate of the fading simulator, and the duration over which the simulated LCR is calculated are also set at 10^4 samples per second and 100 seconds, respectively. The result corresponds to a maximum Doppler shift of 5 Hz.

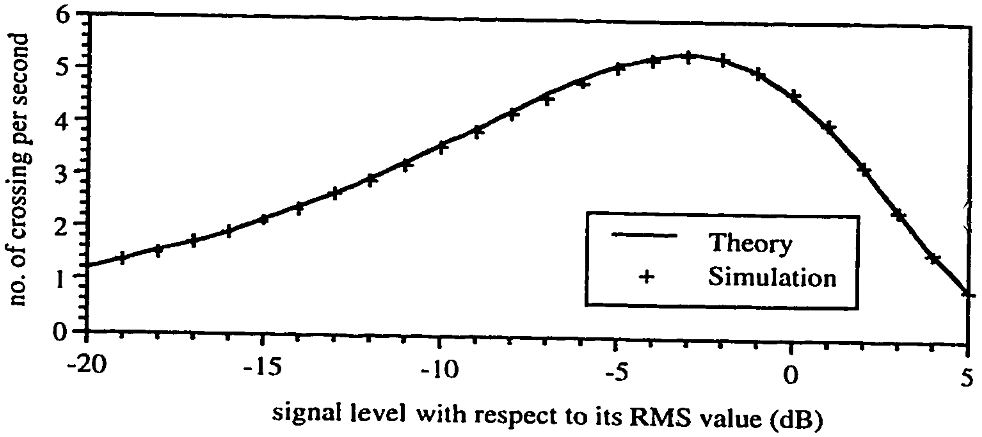

Fig. 2.6. LCR of the envelope of a Rayleigh fading signal.

The RF power spectrum can be obtained as the Fourier transform of the autocorrelation function of the fading signal, expressed in terms of a time delay. In order to compute the RF spectrum, a complex baseband fading signal is first generated with the same C-program in which the sampling rate is set at 20 samples per second. This lower sampling rate allows the generation of a smaller yet sufficient amount of data so as to reduce the computational burden in calculating the RF power spectrum of the signal using MATLAB. The resulting simulated RF power spectrum is shown in Figure 2.7. It resembles the classical Doppler spectrum obtained with an isotropic antenna, which has a sharp cut-off at the maximum Doppler shift of 5 Hz.

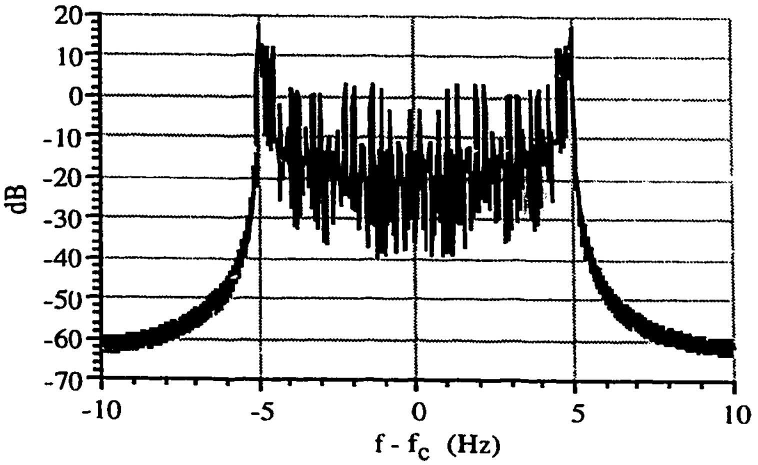

Fig. 2.7. RF power spectrum of a simulated fading signal.

The envelope cross-correlations of fading signals generated by the Rayleigh fader have also been computed. In the study, seven fading signals are generated. The j-th fading signal ($j = 1, 2, ..., 7$) is generated by the fader having its i-th oscillator given an additional phase-shift of γ_{ij}, as defined by (2.13). In the first case, each signal is assigned the same starting time (that is, the time indices of the oscillators for generating these signals are all initialized to the same value). The envelope cross-correlation coefficients are calculated according to [12] as follows:

$$\rho_{mn}(k) = \frac{\sum_{x=1}^{N-k}\left[\frac{e_{m_x} e_{n_{x+k}}}{N-k}\right] - \left(\sum_{x=1}^{N}\left(\frac{e_{m_x}}{N}\right)\right)\left(\sum_{x=1}^{N}\left(\frac{e_{n_x}}{N}\right)\right)}{\sqrt{\sum_{x=1}^{N}\left(\frac{e_{m_x}^2}{N}\right) - \left(\sum_{x=1}^{N}\left(\frac{e_{m_x}}{N}\right)\right)^2}\sqrt{\sum_{x=1}^{N}\left(\frac{e_{n_x}^2}{N}\right) - \left(\sum_{x=1}^{N}\left(\frac{e_{n_x}}{N}\right)\right)^2}} \qquad (2.17)$$

where e_{m_x} and e_{n_x} are the envelope samples of signals m and n , respectively, and N is the number of samples over which each cross-correlation coefficient is calculated. In the computer simulation, the sampling rate and N are set to 100 samples per second and 5×10^5, respectively. Also, only the envelope cross-correlations of signals with zero time shift are considered (i.e. $k = 0$). A few of the results generated are shown in Table 2.1, in which $\rho_{mn}(0)$ is denoted as ρ_{mn}.

$\rho_{12} = 0.9345$	$\rho_{15} = 0.2918$	$\rho_{23} = 0.9343$	$\rho_{26} = 0.2908$	$\rho_{35} = 0.7600$
$\rho_{13} = 0.7611$	$\rho_{16} = 0.1000$	$\rho_{24} = 0.7606$	$\rho_{27} = 0.0992$	$\rho_{36} = 0.5270$
$\rho_{14} = 0.5288$	$\rho_{17} = -0.0097$	$\rho_{25} = 0.5278$	$\rho_{34} = 0.9341$	$\rho_{37} = 0.2902$

Table 2.1. Envelope cross-correlations of signals with the same starting time.

In the second case, the seven signals are regenerated, with each being assigned a different starting time. The envelope cross-correlations of these signals, starting at 10 seconds apart are computed and presented in Table 2.2.

$\rho_{12} = -0.0134$	$\rho_{15} = -0.0321$	$\rho_{23} = -0.0134$	$\rho_{26} = -0.0325$	$\rho_{35} = -0.0047$
$\rho_{13} = -0.0059$	$\rho_{16} = -0.0126$	$\rho_{24} = -0.0050$	$\rho_{27} = -0.0120$	$\rho_{36} = 0.0212$
$\rho_{14} = 0.0210$	$\rho_{17} = 0.0731$	$\rho_{25} = 0.0212$	$\rho_{34} = -0.0132$	$\rho_{37} = -0.0330$

Table 2.2. Envelope cross-correlations of signals with different starting times.

The results presented in Figures 2.4, 2.5 and 2.6 show that the C-program for Jakes' Rayleigh fading simulator is capable of simulating fading signals with the desired statistics. It has also been demonstrated that fading signals, obtained by introducing additional phase shifts according to (2.13) to the oscillators with the same starting time, exhibit different degree of envelope cross-correlation. The degree of cross-correlation depends upon the phase-shifts introduced to the oscillators. However, signals with very low cross-correlations among their envelope fading can always be simulated by assigning different starting times to their corresponding oscillators.

2.2 GMSK Modulation

GMSK is a derivative of a special form of constant envelope, binary continuous phase frequency shift keying (CPFSK) with a modulation index, *h*, of 0.5 and a frequency deviation given by

$$f_d = \frac{f_2 - f_1}{2} = \frac{h}{2T} = \frac{1}{4T} \qquad (2.18)$$

where f_1 and f_2 are the two discrete frequencies of the binary FSK signal representing logic states "1" and "0", respectively, and T is the bit duration. Binary CPFSK with $h = 0.5$ is also known as Minimum Shift Keying (MSK), because it attains the minimum frequency separation ($f_2 - f_1$) required to ensure the orthogonality of signals corresponding to "1" and "0", respectively, over T. The advantages of MSK include its relatively narrow bandwidth and its constant envelope feature which is necessary to avoid problems related to the nonlinearity of RF amplifiers. However, MSK does not satisfy the stringent requirements with respect to the out-of-band radiated power in the adjacent channel, which should be suppressed 60~80 dB below that in the desired channel. Fortunately, the power spectrum of MSK can be made more compact without distorting the signal envelope by introducing a premodulation Gaussian low-pass filter. As a consequence, such a modified MSK is also known as Gaussian MSK or GMSK. The function of the Gaussian filter is to smooth out the phase transitions between pulses in the transmitted signal in order to mitigate spectral broadening. However, by doing so, it inevitably introduces ISI to the digital signal.

2.2.1a. GMSK modulator

The block diagram of a GMSK modulator is given in Figure 2.8.

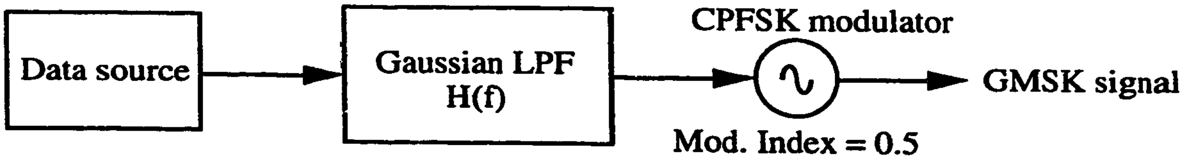

Fig. 2.8. A GMSK transmitter.

The data source generates a series of random non-return-to-zero (NRZ) rectangular pulses,

$$c(t) = \sum_{i=-\infty}^{\infty} a_i \Omega\left(\frac{t-iT}{T}\right) \tag{2.19}$$

where $a_i = \pm 1$ is the data bit, and $\Omega(t/T)$ is a unit rectangular pulse of duration T centered at the origin. These rectangular pulses are then shaped by a premodulation pulse-shaping Gaussian filter which has a transfer function defined by [20]

$$H(f) = A\,exp\left\{-\frac{ln\,2}{2B^2}f^2\right\} \tag{2.20}$$

where B is the 3 dB bandwidth of the filter and A is a constant. The corresponding impulse response is obtained by taking the inverse Fourier transform of (2.20) [21]:

$$h(t) = A\sqrt{\frac{2\pi}{ln\,2}}B\,exp\left\{-\frac{2\pi^2B^2}{ln\,2}t^2\right\} \tag{2.21}$$

The 3 dB bandwidth is adjusted to yield a narrow spectrum for the GMSK signal. However, reducing the value of B to produce a more compact spectrum will simultaneously increase the spreading effect of the pulse shaping filter and thus result in more severe ISI. Both the cross-channel interference due to less stringent bandwidth control, and the reduction in the eye opening of a digital signal caused by ISI can significantly degrade the performance of a system. Therefore, it is necessary to determine the optimum filter bandwidth so that the performance penalty arising from the two contradictory effects can be minimized for a given system.

The constant value A in (2.20) is chosen in such a way that the following condition is satisfied:

$$\int_{-\infty}^{+\infty} g(t)dt = \frac{1}{2} \tag{2.22}$$

where $g(t)$ is the filter response to a unit rectangular pulse of duration T, and is defined as follows:

$$
\begin{aligned}
g(t) &= \Omega\left(\frac{t}{T}\right) * h(t) \\
&= A\sqrt{\frac{2\pi}{ln\,2}}B\int_{t-T/2}^{t+T/2} exp\left\{-\frac{2(\pi Bx)^2}{ln\,2}\right\}dx \qquad (2.23)\\
&= \frac{A}{2}\left\{erf\left(-\sqrt{\frac{2}{ln\,2}}\pi B\left(t-\frac{T}{2}\right)\right)+erf\left(\sqrt{\frac{2}{ln\,2}}\pi B\left(t+\frac{T}{2}\right)\right)\right\}.
\end{aligned}
$$

where the asterisk denotes convolution, and

$$
\begin{aligned}
erf(y) &= \frac{2}{\sqrt{\pi}}\int_0^y exp\left\{-u^2\right\}du \\
erf(y) &= -erf(-y)
\end{aligned}
\qquad (2.24)
$$

After Gaussian shaping, the modulating pulses are used to frequency-modulate a high frequency sinusoidal wave (carrier). The resulting signal at the output of the CPFSK modulator is given by

$$
s(t) = \sqrt{\frac{2E_b}{T}}\,cos\left(2\pi f_c t + \phi(t)\right) \qquad (2.25)
$$

where E_b is the energy per bit and $\phi(t)$ is the phase that carries the transmitted information, which is in turn defined as

$$
\phi(t) = 2\pi h\int_{-\infty}^{\infty}\left(\sum_{i=-\infty}^{\infty} a_i g(x-iT)\right)dx \qquad (2.26)
$$

where h is the modulation index.

2.2.1b. Simulation model of a GMSK modulator

The Gaussian pulse shaping filter with an impulse response defined by (2.21) can be implemented as an N-tap FIR filter with the following impulse response:

$$h[i] = A\sqrt{\frac{2\pi}{ln 2}}B\sum_{i=0}^{N-1} exp\left\{-\frac{2\pi^2 B^2}{ln 2}\left(i-\frac{N-1}{2}\right)^2 T_s^2\right\} \qquad (2.27)$$

where N is an odd integer andT_s is the sampling interval. The filter should be long enough to ensure that the tails of the impulse response that are truncated have negligible amplitudes.

Figure 2.9 illustrates the block diagram of the simulation model of a CPFSK modulator. It consists of a discrete modulo integrator and a real-to-complex converter. The input signal to the modulo integrator is given by

$$b_n = \sum_{i=-\infty}^{\infty} a_i g(nT_s - iT) \qquad (2.28)$$

The resulting output of the integrator is a discrete version of (2.26). Finally, a complex baseband GMSK signal is generated with an amplitude of $\sqrt{2E/T} = 1$.

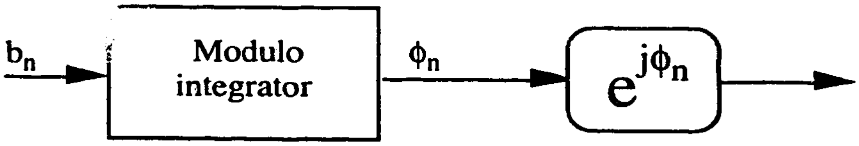

Fig. 2.9. A simulation model of a CPFSK modulator.

The modulo integrator model is based on a numerical integration method using a second order polynomial approximation. According to Simpson's one-third rule, the area bounded by a curve $f(x)$ and the x-axis from $x = x_{i-2}$ to $x = x_i$, as shown in Figure 2.10, is approximated by

$$A_i = \left(\frac{f(x_i) + 4f(x_{i-1}) + f(x_{i-2})}{3} \right) \Delta x \qquad (2.29)$$

where $\Delta x = x_i - x_{i-1} = x_{i-1} - x_{i-2}$. As Δx approaches zero, the result becomes exact.

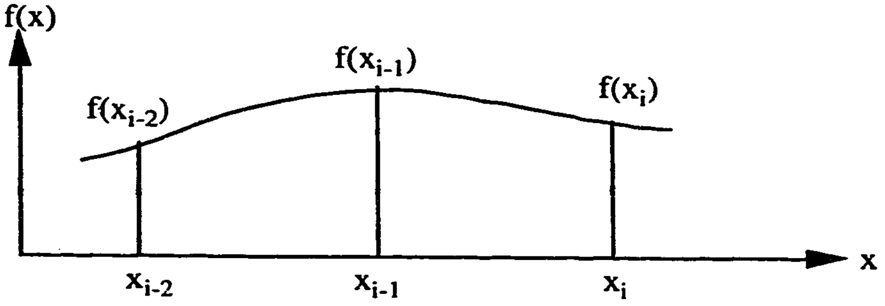

Fig. 2.10. Simpscn's one-third rule for numerical integration.

However, (2.29) cannot be applied directly to model the integrator. This is because, for every input sample $f(x_i)$, the integrator is required to generate a new output by increasing or decreasing its previous output accordingly. By using (2.29), the incremental or decremental value corresponding to the segment bounded by $x = x_{i-2}$ and $x = x_{i-1}$ is calculated twice when the new cumulative output is computed upon the arrival of $f(x_i)$. Hence, a modified version of (2.29), multiplied by a scaling factor of 1/2, is used in modeling the CPFSK modulator, in which the discrete phase of a GMSK signal at time index n is computed according to

$$\phi_n = \left\{ \phi_{n-1} + 2\pi h \left(\frac{b_n + 4b_{n-1} + b_{n-2}}{6} \right) T_s \right\} mod\, 2\pi \qquad (2.30)$$

where h is the modulation index. It should be noted that ϕ_n differs from the actual value, obtained with the original Simpson's one-third rule using the entire sample sequence, by half the amount of phase shift from index n-1 to n. This discrepancy will not cause error accumulation since every phase increment or decrement is calculated twice in the cumulative process and thus will be compensated automatically when the next output is computed. Furthermore, for a relatively small step size (equal to T_s in this case), the difference is negligible. Figure 2.11 depicts a model of the CPFSK modulator. The scaling factors W_1 and W_2 are equal to 4 and $2\pi hT_s/6$, respectively.

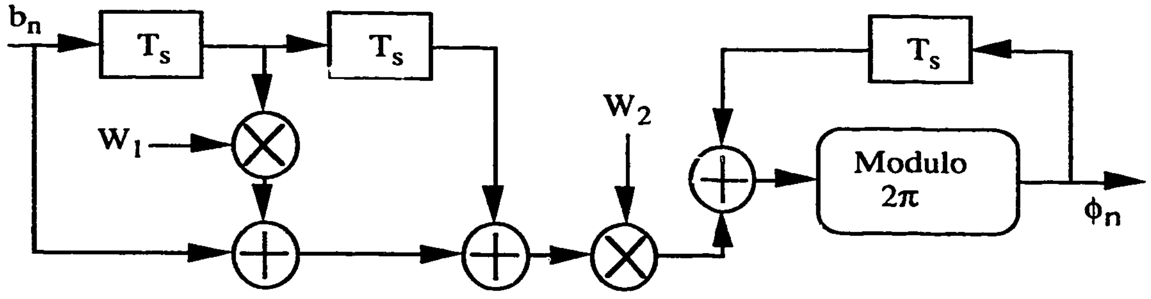

Fig. 2.11. A model of a modulo integrator.

2.2.2a. Demodulator

As described in the previous section, the function of a digital modulator is to map a digital sequence into waveforms that are suitable for transmission over the radio channel. In turn, a demodulator performs the reverse mapping operation, transforming the channel-distorted received waveforms back into the baseband digital signal. There are several demodulation structures which in general can be classified as either coherent or non-coherent demodulations. The former require an accurate recovery of the carrier phase, whereas no such knowledge is required for non-coherent demodulation. Coherent demodulation is known for its superior performance but carrier phase recovery is a complicated process, especially under severe fading conditions and for burst mode transmission in TDMA systems. On the other hand, non-coherent schemes such as limiter-discriminator demodulation and differential detection are relatively simple and effective for recovering data from a signal which has been exposed to severe fading. The performance of a one-bit differential detector is comparable to that of a limiter-discriminator [22]. A two-bit differential detector is superior to a one-bit differential detector in an AWGN channel [23]. However, there is no apparent advantage of ɔsing a two-bit differential detector in a

Rayleigh fading channel since its performance gain over a one-bit differential detector is marginal in such a channel [23]. Therefore, a simple one-bit differential detector has been selected for a basic DECT receiver considered in this thesis.

2.2.2b. One-bit differential detector

A description of a one-bit differential detector is made according to the analysis presented in [23]. For simplicity, it is assumed that a GMSK signal given by (2.25) traverses a flat fading channel and arrives at a receiver shown in Figure 2.12.

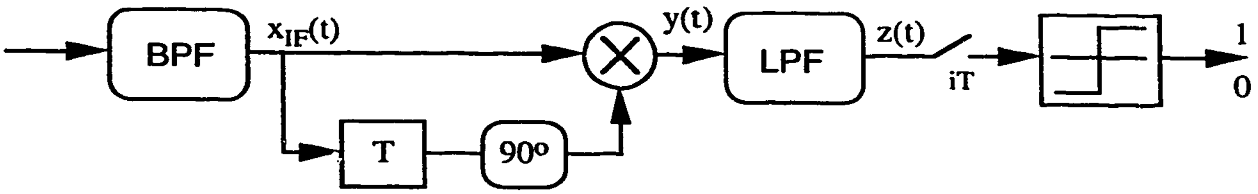

Fig. 2.12. One-bit differential detector.

The predetection bandpass filter, which suppresses the broadband noise and cross-channel interference, is assumed to be of the Gaussian type with the following equivalent lowpass transfer function:

$$H_r(f) = exp\left\{-\frac{2ln2}{B_r^2}f^2\right\} \qquad (2.31)$$

where B_r is the 3 dB bandwidth of the bandpass filter. After bandpass filtering the received signal is given by

$$x_{IF}(t) = \sqrt{\frac{2E_b}{T}}a(t)cos(\omega_c(t)+\phi(t)+\gamma(t))+n(t), \qquad \omega_c = 2\pi f_c \qquad (2.32)$$

where $a(t)$ and $\gamma(t)$ are the envelope and phase distortion, respectively, introduced by the channel and the bandpass filter, and $n(t)$ is the bandpass Gaussian noise. The noise term can be expressed in its narrow band representation

$$n(t) = n_c(t)cos\left(\omega_c(t) + \hat{\phi}(t)\right) + n_s(t)sin\left(\omega_c(t) + \hat{\phi}(t)\right) \qquad (2.33)$$

where $\hat{\phi}(t) = \phi(t) + \gamma(t)$ and $n_c(t)$ and $n_s(t)$ are independent lowpass zero mean Gaussian processes with variance

$$\sigma^2 = N_o \int_{-\infty}^{\infty} \left|H_r(f)\right|^2 df = N_o B_n, \qquad B_n = \sqrt{\frac{\pi}{4 ln\, 2}} B_r \qquad (2.34)$$

$N_o/2$ is the two-sided power spectral density of the channel noise and B_n is the noise equivalent bandwidth of the bandpass filter. By substituting (2.33) into (2.32), $x_{IF}(t)$ can be expressed in its polar form as follows:

$$\begin{aligned} x_{IF}(t) &= Re\left[\left(\sqrt{\frac{2E_b}{T}} a(t) + n_c(t) + jn_s(t)\right) exp\left\{j\left(\omega_c(t) + \hat{\phi}(t)\right)\right\}\right] \\ &= A(t)cos\left(\omega_c(t) + \hat{\phi}(t) + \eta(t)\right) \end{aligned} \qquad (2.35)$$

where,

$$A(t) = \sqrt{\left(\left(\sqrt{\frac{2E_b}{T}} a(t) + n_c(t)\right)^2 + n_s(t)^2\right)}$$

$$\eta(t) = tan^{-1}\left(\frac{n_s(t)}{\sqrt{\frac{2E_b}{T}} a(t) + n_c(t)}\right) \qquad (2.36)$$

As shown in Figure 2.12, the signal $x_{IF}(t)$ is multiplied by a version of itself that is delayed by one bit duration and phase-shifted by +90°. Consequently, the signal at the output of the multiplier becomes

$$y(t) = \frac{A(t)A(t-T)}{2}\left[\begin{array}{l} sin\big(\omega_c T + \hat{\phi}(t) - \hat{\phi}(t-T) + \eta(t) - \eta(t-T)\big) - \\ sin\big(2\omega_c t - \omega_c T + \hat{\phi}(t) + \hat{\phi}(t-T) + \eta(t) + \eta(t-T)\big) \end{array}\right] \tag{2.37}$$

The second harmonic term is then removed by a lowpass filter, resulting in

$$z(t) = \frac{A(t)A(t-T)}{2} sin\big(\omega_c T + \Delta\Phi(T)\big) \tag{2.38}$$

where $\Delta\Phi(T) = \hat{\phi}(t) - \hat{\phi}(t-T) + \eta(t) - \eta(t-T)$ represents the change over one bit duration of the sum of distorted signal phase and noise phase contributed by the additive Gaussian noise. Also, by assuming that the carrier frequency is chosen in such a way that $\omega_c T = 2\pi k$, where k is an integer, (2.38) can be simplified to

$$z(t) = \frac{A(t)A(t-T)}{2} sin\big(\Delta\Phi(T)\big) \tag{2.39}$$

The signal $z(t)$ is then sampled at $t = iT$, $i = 0, 1, 2, \ldots$, etc., and fed into a decision device which decides that a "1" was sent if $z(iT) > 0$, and a "0" otherwise.

The performance sensitivity of a one-bit differential detector to the $B_r T$ product of the receiver Gaussian bandpass filter is also discussed in [23]. It shows that, for one-bit differential detection of GMSK, an optimum value of receiver $B_r T$ product exists for each BT product of the Gaussian pulse shaping filter at the transmitter. For a BT of 0.5, the optimum $B_r T$ value is 1.1.

2.3. Summary

In this chapter, discussions of an indoor radio channel-modeling and GMSK modulation were given. The channel was modeled as a tapped delay line, with a fading signal and an appropriate attenuation being applied to each tap. The statistical characteristics of the fading simulator have been analyzed via computer simulation. It was demonstrated that the fading signal simulated by Jakes' Rayleigh fader exhibited the desired characteristics. In addition, the envelope cross-correlations of fading signals generated by the Rayleigh fader have been investigated. It was shown that a simple way to realize uncorrelated fading signals is to assign different starting times to the corresponding oscillators used to generate the fading signals. A model of GMSK modulator for computer simulation was also presented. Finally, simple non-coherent detection of the GMSK signal using a one-bit differential detector was discussed. This detection scheme did not require the recovery of a carrier phase; a current decision on a received signal was made on the basis of the phase difference between the currently received signal and the one received one symbol interval earlier.

In the following chapter, performance enhancing schemes involving adaptive channel equalization and antenna diversity combining will be introduced.

3. EQUALIZATION AND DIVERSITY COMBINING

Adaptive channel equalization and diversity combining techniques are reviewed in this chapter. The principles of channel equalization, equalizer structures and recursive algorithms for equalizer adaptation are discussed in the first part. The second part describes several diversity techniques and related combining methods.

3.1. Adaptive Equalization

To achieve reliable high-speed data transmission in a band-limited channel, channel equalization is required in order to reduce intersymbol interference (ISI). Before discussing adaptive channel equalization, a review of the mechanism that produces ISI is needed. The following treatment of the topic is based on the presentation in [24].

3.1.1. Intersymbol interference (ISI)

In digital communications, the transmission of symbols over a time dispersive channel results in the spreading and overlap of the transmitted symbols, producing ISI. Furthermore, ISI is also inherent in partial response modulation schemes such as GMSK. The effect of ISI can be readily described for a baseband pulse-amplitude modulation (PAM) system, as shown in Figure 3.1.

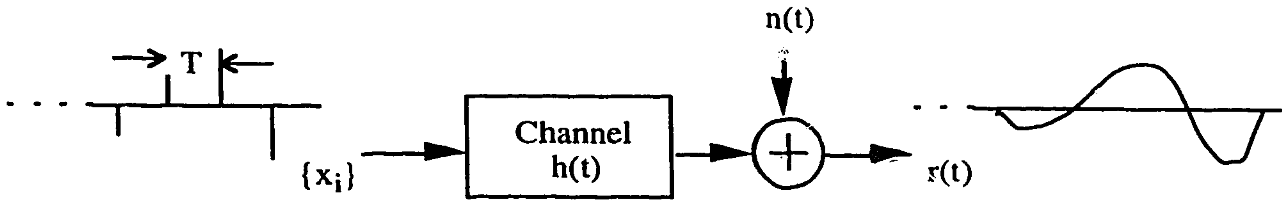

Fig. 3.1. Baseband PAM system model.

In this model, it is convenient to combine the effects of the modulator, the transmitter filter and the transmission medium into a single 'channel' or system impulse response $h(t)$. The received signal can then be expressed as the superposition of the impulse responses of the channel to the individual transmitted symbols plus additive Gaussian noise $n(t)$:

$$r(t) = \sum_i x_i h(t - iT) + n(t) \tag{3.1}$$

where x_i is a transmitted symbol and T is the signaling interval. An example of $h(t)$ is shown in Figure 3.2. The received signal $r(t)$ is then sampled at $kT+t_o$, where t_o accounts for the channel delay and sampler phase. The resulting sampled version of $r(t)$ becomes

$$r(kT + t_o) = x_k h(t_o) + \sum_{i \neq k} x_i h(kT + t_o - iT) + n(kT + t_o) \tag{3.2}$$

The first term on the right hand side of (3.2) is due to the desired symbol, the second term represents the ISI from adjacent symbols, and the last term is the additive noise. The ISI term is zero if and only if $h(kT+t_o) = 0$, for $k \neq 0$.

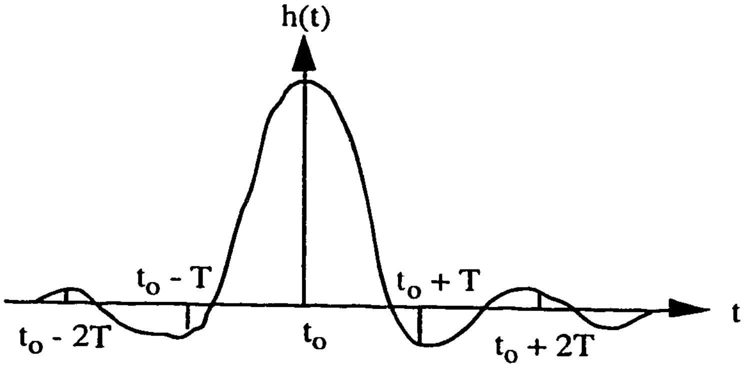

Fig. 3.2. Channel impulse response.

If the channel impulse response is zero at instants $kT+t_o$ for $k \neq 0$, it is said to satisfy the criterion for zero ISI known as Nyquist's first criterion [25]. This corresponds to a channel which has the following folded frequency spectrum after symbol-rate sampling:

$$H_f(f)=\sum_i H\left(f-\frac{i}{T}\right)=constant, \qquad |f|\leq\frac{1}{2T} \tag{3.3}$$

where $H(f)$ is the channel frequency response.

3.1.2. Linear equalization

Traditionally, most of the research on the reduction of ISI was motivated by the desire to transmit high speed data over telephone channels. The main focus was on linear filters, either fixed or manually adjustable, to reduce the ISI introduced by the channel. These linear filters equalize the channel by compensating for amplitude and delay distortions. Thus, they are also known as equalizers. A fixed linear equalizer is intended to compensate an average channel characteristic. As a particular channel characteristic may differ significantly from the average, a fixed equalizer does not provide good performance. On the other hand, a manually adjustable equalizer may be able to compensate for channel distortions to an extent limited by the number of degrees of freedom provided by the available adjustable parameters. However, these filters are not able to cancel time-variable ISI resulting from variations in the channel impulse response with time. This problem was overcome following a major advance in channel equalization in 1964, when an algorithm for automatically adjusting the parameters of an equalizer was published. Today, several algorithms with different complexities and performance are available for automatic adaptive equalization.

The most commonly used linear equalizer for combating ISI is a transversal filter, shown in Figure 3.3. This is a symbol-spaced (T-spaced) transversal equalizer in which the current and $N-1$ past samples of the received signal are multiplied by the equalizer coefficients c_i, $i = 0, 1, ..N-1$, and summed to produce an output:

$$y_k=\sum_{i=0}^{N-1} c_i r(t_o+kT-iT) \tag{3.4}$$

The number of taps, or the span of the equalizer, depends on the maximum duration of the ISI, and the delay introduced by the equalizer is determined by the position of its reference tap. The equalizer coefficients can be optimized using different criteria, such as the zero-forcing (ZF) and mean-squared error (MSE) criteria.

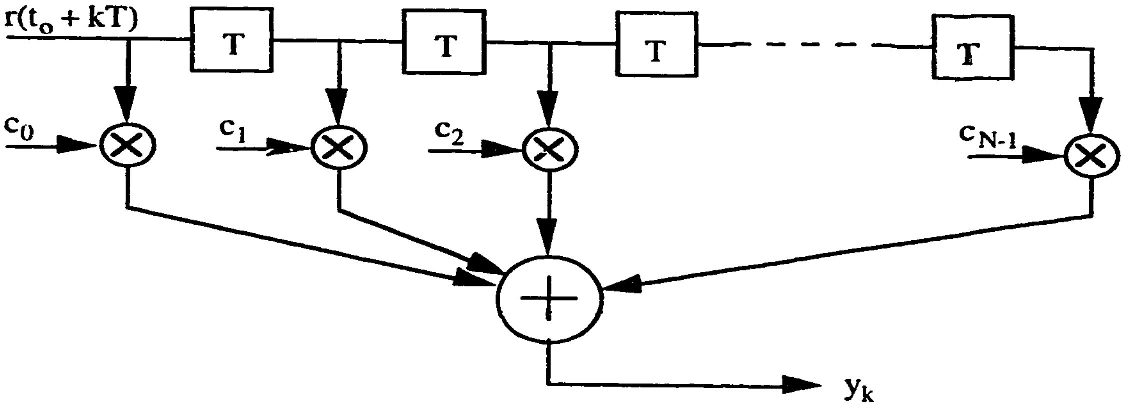

Fig. 3.3. A symbol spaced transversal filter equalizer.

In a zero-forcing equalizer with N symbol-spaced taps, the equalizer coefficients are set so that they will force the composite channel and equalizer impulse response to zero at all but one of the N symbol-spaced instants. If we assume that the equalizer span is of infinite length (i.e., it has an infinite number of T-spaced taps), then the combined frequency response of the channel and the ZF-equalizer would satisfy Nyquist's first criterion for zero ˙SI:

$$C(f)H_f(f) = 1, \qquad |f| \leq \frac{1}{2T} \tag{3.5}$$

where $C(f)$ is the frequency response of the equalizer, which is periodic because of the T tap-spacing, and $H_f(f)$ is the folded frequency response of the channel. This shows that a ZF-equalizer simply inverts the folded frequency response of a channel. A finite length ZF-equalizer approximates this inverse function. Such an inverse operation may have undesirable effects. If the folded channel power spectrum has severe attenuation or spectral nulls, then a ZF-equalizer will inevitably cause excessive noise enhancement at frequencies where the attenuation occurs while trying to synthesize a flat overall response by introducing large gains in the affected regions; this leads to performance degradation. Also, if the binary eye of the input signal is initially closed, there is no guarantee that a finite length ZF-equalizer will minimize the distortion [24].

A MSE criterion provides a more robust approach for adjusting the equalizer coefficients. In contrast to a ZF-equalizer, the coefficients of the least mean-squared equalizer are optimized in such a way that the mean squared error between the desired equalizer output and the actual equalizer output is minimized. In this way, it maximizes the signal to distortion ratio at its output [24].

3.1.3. Fractionally spaced equalizer

Symbol-rate sampling of an input signal before a *T*-spaced equalizer causes spectral overlap, which in turn results in additional band-edge distortion. When the aliasing components add destructively, severe attenuation or spectral nulls may occur in the band-edge regions, leading to noise enhancement for the reason mentioned previously. To circumvent this problem, the spacing between adjacent taps of an equalizer is reduced to a fraction of the symbol interval. Such an equalizer is known as a fractionally spaced equalizer (FSE). In a digital implementation, the tap-spacing is set at *KT/M*, where *K* and *M* are integers and *M*>*K*. Figure 3.4 shows an example of a FSE with tap spacing of *3T/4*.

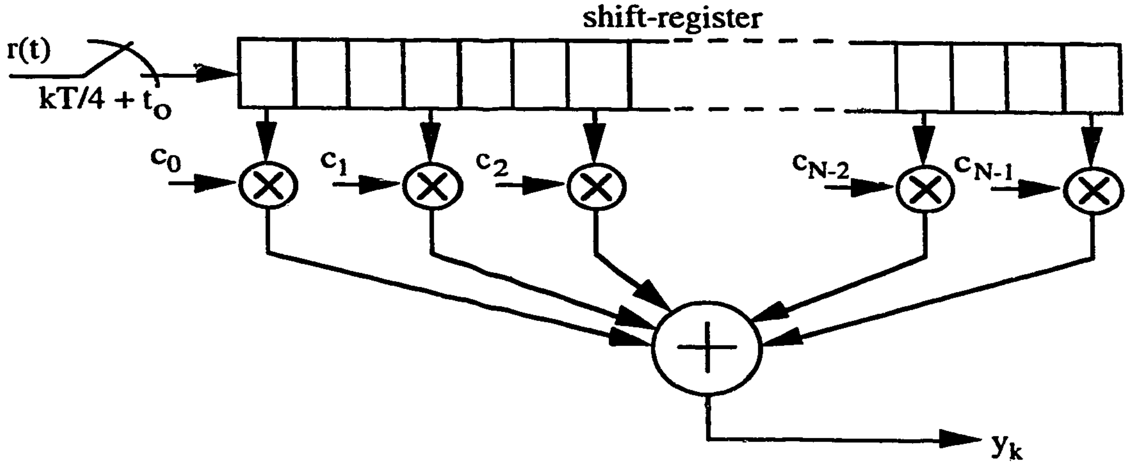

Fig. 3.4. A fractionally spaced transversal equalizer with *3T/4* tap-spacing.

A received signal is sampled and shifted into a FSE shift register at a rate of *M/T* and the signal sample in every *K*-th memory location of the shift-register is multiplied by an appropriate filter coefficient. The output of a FSE, which is produced every *T* interval, is given by

$$y_k = \sum_{i=0}^{N-1} c_i r\left(t_o + kT - i\frac{KT}{M}\right) \qquad (3.6)$$

As a result of the higher sampling rate (since $M \geq 2$) before a FSE, no aliasing of the sampled signal spectrum occurs (assuming that the signal before sampling is bandlimited to *1/T*). Consequently, this allows the FSE to manipulate the signal spectrum at the two band-edge regions independently of one another in such a way that cancellation of aliasing band-edge components can be avoided in the course of symbol-rate sampling of the equalizer output [24].

A related property of a fractionally spaced equalizer is its ability to synthesize any timing delay. As a consequence, its performance becomes insensitive to variation of the phase of a sampler, which corresponds to a variable delay of the signal path. The effect of non-optimal timing phase of a sampler on the performance of an equalizer is explained in the following development.

Let us assume that the front end of a receiver consists of a matched filter with an impulse response $h^*(-t)$, where $h(t)$ is the complex baseband equivalent of the composite transmitter and channel impulse response, and the asterisk denotes complex conjugation. Following the matched filter is a *1/T*-rate sampler, and an infinite-length *T*-spaced transversal filter with a frequency response $C(f)$. The received signal is defined in (3.1). If the transmitted data sequence $\{x_k\}$ is uncorrelated and has unit power, then the signal spectrum at the output of the matched filter is $|H(f)|^2$, where $H(f)$ is the Fourier transform of $h(t)$. After *1/T*-rate sampling, the folded signal spectrum is given by

$$S(f,\tau_o) = \sum_i \left|H\left(f - \frac{i}{T}\right)\right|^2 exp\left\{-j2\pi\left(f - \frac{i}{T}\right)\tau_o\right\} \qquad (3.7)$$

where τ_o is the timing offset from the optimum sampling time. If it is also assumed that $H(f)$ is equal to zero for $|f| > 1/T$, then (3.7) can be simplified to

$$S(f,\tau_o) = \left[|H(f)|^2 + \left| H\left(f - \frac{1}{T}\right)\right|^2 exp\left\{ j2\pi \frac{\tau_o}{T} \right\} \right] exp\{-j2\pi f\tau_o\}, \quad 0 \le f \le \frac{1}{T} \tag{3.8}$$

From (3.8), we observe that when τ_o is equal to zero, $|H(f-1/T)|$ adds constructively to $|H(f)|$, whereas destructive aliasing takes place when τ_o is equal to $T/2$. Particularly, if the power spectrum $|H(f)|^2$ is symmetric about the centre frequency, a spectral null is created in the band-edge ($|f| = 1/2T$) of the sampled signal spectrum when $\tau_o = T/2$. In this case, a T-spaced equalizer cannot manipulate the spectral null into a flat spectrum without significant noise enhancement. On the other hand, when the sampling rate is increased to M/T, (3.7) becomes

$$S'(f,\tau_o) = \sum_i \left| H\left(f - i\frac{M}{T}\right)\right|^2 exp\left\{ -j2\pi\left(f - i\frac{M}{T}\right)\tau_o \right\} \tag{3.9}$$

Given that $M \geq 2$ and $|H(f)| = 0$ for $|f| > 1/T$, (3.9) can be expressed as

$$S'(f,\tau_o) = |H(f)|^2 exp\{-j2\pi f\tau_o\}, \qquad 0 \le f \le \frac{1}{T} \tag{3.10}$$

Equation (3.10) shows that aliasing is avoided at the input to a FSE. It can be shown that an infinite-length transversal FSE is capable of synthesizing the composite characteristics of a matched filter and a infinite-length transversal T-spaced equalizer [24]. This FSE can also explicitly compensate for the timing offset of the sampler at its input by formulating a linear phase adjustment of $exp\{j2\pi f\tau_o\}$. Therefore, the FSE, whose frequency response can be expressed as $C(f)H^*(f)exp\{j2\pi f\tau_o\}$ for $0 \leq f \leq 1/T$, can effectively minimize the MSE of its output after symbol-rate sampling by acting on the pre-aliased signal spectrum and compensating for any timing offset in the process of equalization.

However, linear equalizers generally exhibit poor performance on channels with severe amplitude distortion. Particularly, a linear equalizer is incapable of equalizing a channel containing spectral nulls without introducing excessive noise enhancement to the received signal [13]. Therefore, it is necessary to introduce another class of equalization techniques known as non-linear equalization. Non-linear equalizers are more robust and are capable of handling channels with severe amplitude distortion.

3.1.4. Non-linear equalization

Significant advances in the area of non-linear equalization have been made over the past three decades. They include the development of a symbol-by-symbol detection algorithm based on the maximum *a posteriori* probability (MAP) criterion, a sequence detection algorithm based on maximum-likelihood sequence estimation (MLSE), and decision feedback equalization [13].

The MAP and MLSE algorithms are based on probabilistic formulations. The MAP algorithm is optimum in the sense of minimizing the symbol error rate for symbol-by-symbol detection. On the other hand, the MLSE technique minimizes the probability of symbol error based upon the detection of the whole signal sequence using the computationally efficient Viterbi algorithm. Both the MAP and MLSE algorithms require a knowledge of the channel characteristics as well as of the statistical distribution of the additive noise in order to compute the probabilities for making decisions [13]. The channel characteristic, which is usually unknown and non-stationary, can be approximated by a channel estimator. These probabilistic algorithms are able to achieve superior performance. However, the performance gain is attained at the expense of computational complexity which grows exponentially with the number of interfering paths or the ISI span.

Among the three non-linear equalization techniques mentioned above, decision feedback equalization is the least complex. However, it is inferior to the other two methods. In contrast to the MLSE algorithm which makes use of all the energy in the discrete channel impulse response to maximize the effective SNR, a decision feedback equalizer (DFE) cancels all trailing ISI terms [24]. Besides suffering from a reduced effective SNR, the DFE also suffers from error propagation. Nevertheless, despite its shortcomings, decision feedback equalization has been demonstrated to be a very effective technique for combating ISI in a channel with severe amplitude distortion in its frequency response. On the basis of its ease of implementation, and simple integration with antenna diversity, the DFE has been selected for the applications considered in this thesis.

3.1.5. Decision feedback equalizer

The principle of using previous decisions to remove the distortion of received signals is not new. In the late 1960s, this principle was applied to equalization, which subsequently led to the development of a robust equalization technique known as decision feedback equalization.

A conventional decision feedback equalizer (DFE) consists of a feedforward section and a feedback section, as illustrated in Figure 3.5. The feedforward section is a linear transversal filter known as a feedforward filter (FFF), whose tap spacing τ may either be equal to a symbol interval or a fraction of a symbol interval. The feedback section is a symbol spaced (T-spaced) transversal filter, which is referred to as a feedback filter (FBF). The inputs of the FBF are previously detected symbols. Obviously, the non-linearity of the equalizer results from the inclusion of a hard-decision device. For generality, the tap weights (coefficients) of the filters and the tap inputs may be assumed to be complex.

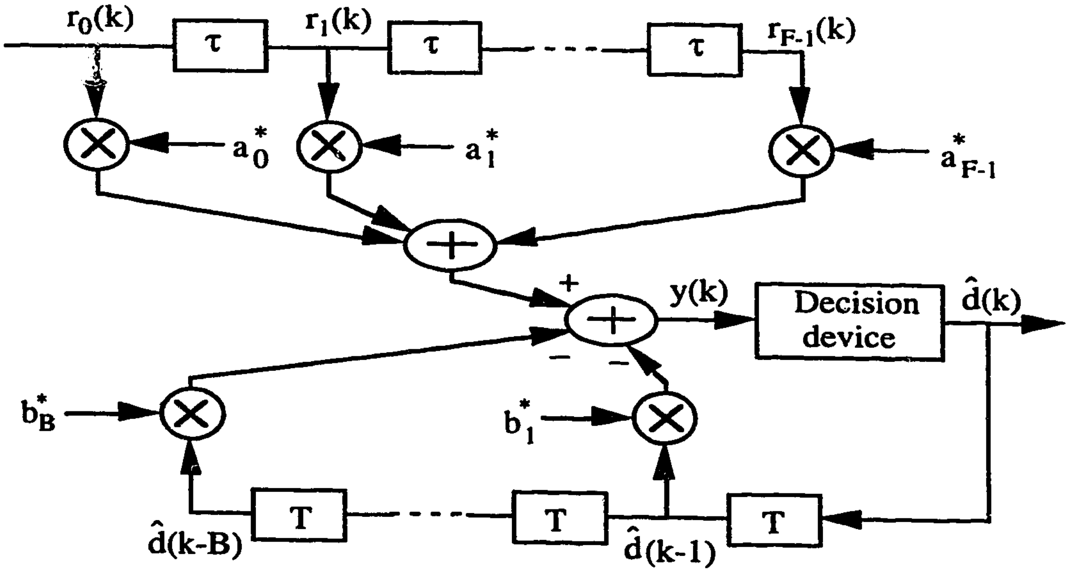

Fig. 3.5. Decision Feedback Equalizer.

The complex lowpass equivalent signal is sampled and fed to the feedforward filter with F tap-coefficients $\{a_i\}$. The feedback filter, with B tap-coefficients $\{b_j\}$, generates a weighted sum of the noise-free past decisions which is then subtracted from the output of the feedforward filter in order to cancel ISI caused by the past detected symbols in the present signaling interval. Consequently, the input to the decision device (output of the equalizer) is given by

$$y(k)=\sum_{i=0}^{F-1} a_i^* r_i(k)-\sum_{i=1}^{B} b_i^* \hat{d}(k-i) \qquad (3.11)$$

where $\{r_0(k), r_1(k), ..., r_{F-1}(k)\}$ are tap inputs of the FFF at time k and $\{\hat{d}(k-1), \hat{d}(k-2),..., \hat{d}(k-B)\}$ are the previously detected symbols residing at the tap inputs of the FBF. The filters' coefficients may be jointly optimized in such a way that the mean squared error (MSE) at the equalizer output is minimized. An optimization algorithm for linear equalization can be utilized in this case by grouping the coefficients of the filters into one single vector and the corresponding tap inputs of the filters into another vector as follows [26]:

$$\boldsymbol{w}=\left[a_0, a_1, ...a_{F-1}, -b_1, -b_2, ..., -b_B\right]^T$$
$$\boldsymbol{u}(k)=\left[r_0(k), r_1(k), ..., r_{F-1}(k), \hat{d}(k-1), \hat{d}(k-2), ..., \hat{d}(k-B)\right]^T$$

where ^{T} denotes matrix transposition. The input to the decision device can now be cast into a familiar form encountered in linear equalization:

$$y(k)=\boldsymbol{w}^H \boldsymbol{u}(k) \qquad (3.12)$$

where ^{H} denotes Hermitian transposition. The vectors $\boldsymbol{w}$ and $\boldsymbol{u}(k)$ are then used by an adaptation algorithm for the joint optimization of the feedforward and feedback filters' coefficients. Equalizer adaptation is discussed in Section 3.1.6.

With the assumption of correct past decisions, it can be shown that the MSE with decision feedback is always smaller than the MSE of a transversal equalizer [27]. However, realistically, occasional spurious decisions do occur. An erroneous decision reduces the margin against noise for future decisions and consequently, the probability of the next decision error is increased. Therefore, there is a concern for severe performance degradation due to indefinite error propagation. Fortunately, the effect of error propagation in a DFE is not catastrophic. It has been demonstrated that, for a finite-length feedback filter with B taps, the actual probability of error at a high signal-to-noise ratio is at most a factor of 2^B higher than the error probability obtained with ideal decision feedback [28].

In conventional DFE operation, precursors or leading samples in a channel impulse response are dealt with strictly by the FFF, whereas postcursors or trailing echoes of the channel, possibly enhanced slightly by the FFF, are handled by the FBF. In this case, the last tap of the FFF is the reference tap of the DFE. However, it has also been proposed to introduce additional taps to the FFF such that the middle tap of the FFF is used as the reference tap [29]. Bingham explains that in the presence of additive noise, there exists a unique MSE solution where the leading taps of such a FFF deal with any precursors while its trailing taps follow the leading taps in such a way as to minimize the output noise; the FBF then deals with both the original postcursors as well as those created by the FFF. Nevertheless, the addition of extra taps to the FFF increases equalizer complexity and also results in a more slowly converging DFE.

3.1.6. Equalizer Adaptation

In a practical situation, an equalizer needs to be adaptive in order to effectively equalize a channel having unknown and time varying characteristics. The optimization of an equalizer can be achieved by using either conventional, recursive adaptation algorithms or adaptive channel estimates. In the conventional approach, either a *stochastic gradient-based algorithm* or a *least squares algorithm* is implemented to adjust the equalizer coefficients directly. For adaptation using channel estimates, the equalizer coefficients satisfying a minimum mean-square-error criterion are computed periodically using the estimated channel impulse response constructed by a channel estimator. It has been shown that equalizer adaptation using channel estimates can be more robust over a moderately rapid fading channel than that using recursive algorithms [30]. However, conventional recursive adaptation methods are considerably simpler to implement. For this reason, they are considered for equalizer adaptation in this thesis.

For an initial adjustment of the equalizer coefficients (tap weights) using a recursive algorithm, a training sequence is usually transmitted to allow the equalizer to acquire information about the channel characteristics before data transmission begins. The equalizer coefficients are adaptively adjusted by making use of the error signal derived from the known training signal $d(k)$ and the corresponding equalizer output $y(k)$, as shown in Figure 3.6. During data transmission, the coefficients are continuously updated by using the decisions at the output of the decision device $\hat{d}(k)$ to determine the error signal for adaptation. Occasional incorrect decisions made during this decision-directed mode have a negligible effect on the performance of a transversal equalizer provided that the average symbol error rate is below 10^{-2} [13]. Incorrect decisions, however, may have a more detrimental effect on the performance of a DFE since, as discussed earlier, the feedback of spurious decisions will lead to error propagation.

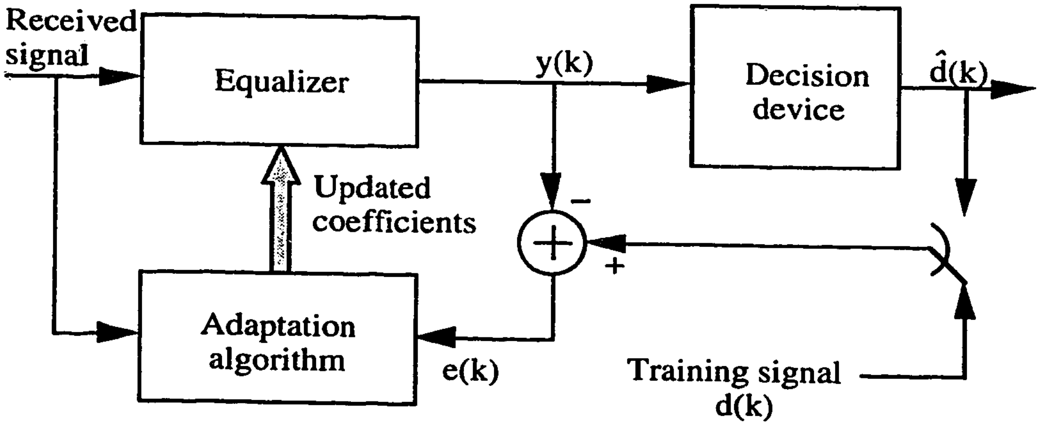

Fig. 3.6. An adaptive equalizer.

3.1.6a. Least mean square algorithm

It can be shown that the mean squared error signal is a quadratic function of the equalizer coefficients. Thus, it can be visualized as a multi-dimensional paraboloid with a minimum point. In other words, there exists an optimum set of equalizer coefficients with which the minimum mean squared error (MMSE) between the actual output and the desired output of the equalizer is attained. According to the *method of steepest descent*, each equalizer coefficient is adjusted in the direction of the gradient of the MSE with respect to that coefficient, that is

$$w_i(k+1) = w_i(k) + \frac{1}{2}\mu\left(-\frac{\partial E\left[\left|e(k)\right|^2\right]}{\partial w_i(k)}\right) \tag{3.13}$$

where $w_i(k)$ is the value of the i-th coefficient at time instant k, μ is the step size parameter, E is the operator denoting ensemble averaging, and $e(k)$ is the error signal given by

$$e(k) = d(k) - \sum_{i=0}^{N-1} w_i^*(k)u_i(k) \tag{3.14}$$

with $d(k)$ designates the desired equalizer output (as shown in Figure 3.6, $d(k)$ is replaced by $\hat{d}(k)$ after the training period), $u_i(k)$ represents the input to the i-th tap of the equalizer at time instant k, and the asterisk denotes complex conjugation. Provided an appropriate step size is used, the equalizer coefficients will converge from arbitrary initial values to the final optimum Wiener solution. However, the computation of the exact gradient of the MSE is impractical. As a consequence, the gradient must be estimated from the available data. This leads to the development of an algorithm which uses a noisy estimate instead of the exact gradient of the MSE. Such an algorithm is known as the *stochastic gradient algorithm*. A well known *stochastic gradient algorithm* is the least-mean-square (LMS) algorithm introduced originally by Widrow. According to the LMS algorithm, the N equalizer coefficients are adaptively adjusted as follows:

$$w_i(k+1) = w_i(k) + \frac{1}{2}\mu\left(-\frac{\partial|e(k)|^2}{\partial w_i(k)}\right), \qquad i = 0,\dots,N-1 \qquad (3.15)$$

Given that the derivative of $|e(k)|^2$ with respect to the complex $w_i(k)$ is equal to $-2u_i(k)e^*(k)$, equation (3.1.5) can be expressed as

$$w_i(k+1) = w_i(k) + \mu u_i(k)e^*(k) \qquad (3.16\ )$$

Even though the set of directions along which the equalizer coefficients are adjusted from one iteration to the next is quite random, the algorithm is still capable of achieving good performance. This is because the recursive nature of the algorithm, in some sense, averages the estimate during the course of adaptation [31]. The step size parameter controls the rate of convergence of the algorithm. A large step size provides rapid convergence but it also results in large fluctuations in the equalizer coefficients during steady-state operation. This fluctuation can be reduced with a smaller step size, but that will result in a slower convergence rate.

Since the step size is the only parameter used to control the rate of adaptation and it is assigned a small value in order to minimize the steady-state mean squared error at the equalizer output, especially in channels which contain spectral nulls, the LMS algorithm converges slowly. If it is possible to introduce an additional parameter in such a way that the step size is first assigned a large value to facilitate a fast convergence and then adjusted adaptively to a smaller value to acquire a small steady-state mean squared error, the performance of the algorithm can be improved. However, the adaptive adjustment of the step size may be difficult if not impractical. The drawbacks of the standard LMS algorithm render it unsuitable for wireless communications since deep spectral fades are likely to occur in the channel and the introduction of a long training sequence may not be practicable. A more quickly converging and more robust *least squares algorithm* is required for such applications.

3.1.6b. Recursive least squares algorithm

The performance index for the *method of least squares* is expressed in terms of a time average instead of an ensemble average. According to this method, the best solution is obtained by minimizing the sum of the squared difference between the estimated results and the corresponding actual results. It can be extended to develop a recursive algorithm for equalizer adaptation whereby equalizer coefficients are updated in a symbol-by-symbol manner. Such an algorithm is known as the recursive least squares (RLS) algorithm.

With the RLS algorithm, the N equalizer coefficients are generated at time k to minimize the sum of all errors as if the same coefficients were used during the observation interval $1 \leq i \leq k$, that is, they minimize the following cost function:

$$\varepsilon(k) = \sum_{i=1}^{k} \left| d(i) - \boldsymbol{w}(k)^H \boldsymbol{u}(i) \right|^2 \qquad (3.17)$$

where $d(i)$ is the desired equalizer output at time instant i, ^{H} indicates Hermitian transposition, $\boldsymbol{u}(i)$ is an equalizer tap-input vector given by

$$\boldsymbol{u}(i) = \left[u_0(i), u_1(i), \ldots.., u_{N-1}(i)\right]^T, \qquad 1 \leq i \leq k \qquad (3.18)$$

and $\boldsymbol{w}(k)$ is the coefficient vector given by

$$\boldsymbol{w}(k) = \left[w_0(k), w_1(k), \ldots.., w_{N-1}(k)\right]^T \qquad (3.19)$$

where ^{T} denotes the matrix transposition.

In order to allow tracking of a time-varying channel, a weighting factor may be introduced to ensure that contributions from data in the distant past are diminished. One such commonly used weighting (or forgetting) factor is the exponential weighting factor. The resulting cost function for the RLS algorithm can be rewritten as

$$\varepsilon(k)=\sum_{i=1}^{k}\lambda^{k-i}\left|d(i)-\boldsymbol{w}^{H}(k)\boldsymbol{u}(i)\right|^{2} \qquad (3.20)$$

where λ is a positive constant close to, but less than, 1.

The desired coefficients for minimizing the cost function are obtained by setting the derivative of (3.17) with respect to $\boldsymbol{w}(k)$ to zero. Thus we have the following solution

$$\boldsymbol{w}(k)=\boldsymbol{A}^{-1}(k)\boldsymbol{B}(k) \qquad (3.21)$$

where the N-by-N correlation matrix $\boldsymbol{A}(k)$ is given by

$$\boldsymbol{A}(k)=\sum_{i=1}^{k}\lambda^{k-i}\boldsymbol{u}(i)\boldsymbol{u}^{H}(i) \qquad (3.22)$$

and the N-by-1 cross-correlation vector $\boldsymbol{B}(k)$ is defined as

$$\boldsymbol{B}(k)=\sum_{i=1}^{k}\lambda^{k-i}\boldsymbol{u}(i)d^{*}(i) \qquad (3.23)$$

It can be shown that both $\boldsymbol{A}(k)$ and $\boldsymbol{B}(k)$ can be recursively updated [31] as follows:

$$\boldsymbol{A}(k)=\lambda\boldsymbol{A}(k-1)+\boldsymbol{u}(k)\boldsymbol{u}^{H}(k) \qquad (3.24)$$

$$\boldsymbol{B}(k)=\lambda\boldsymbol{B}(k-1)+\boldsymbol{u}(k)d^{*}(k) \qquad (3.25)$$

Consequently, by making use of the *matrix inversion lemma* , the following equation for updating $\boldsymbol{w}(k)$ is obtained from (3.21), (3.24) and (3.25):

$$\boldsymbol{w}(k)=\boldsymbol{w}(k-1)+\boldsymbol{K}(k)\alpha^{*}(n) \qquad (3.26)$$

where $K(k)$ is the Kalman gain vector, defined as

$$K(k) = A^{-1}(k)u(k) \qquad (3.27)$$

and $\alpha(k)$ is the *a priori* estimation error given by

$$\alpha(k) = d(k) - w^H(k-1)u(k) \qquad (3.28)$$

The Kalman gain vector $K(k)$ and the inverse correlation matrix $P(k) = A^{-1}(k)$ defined above can be computed recursively according to

$$K(k) = \frac{\lambda^{-1}P(k-1)u(k)}{1 + \lambda^{-1}u^H(k)P(k-1)u(k)} \qquad (3.29)$$

and

$$P(k) = \lambda^{-1}P(k-1) - \lambda^{-1}K(k)u^H(k)P(k-1) \qquad (3.30)$$

To apply the above standard RLS algorithm, it is first required to initialize the inverse correlation matrix $P(k)$. A simple procedure involves the following assignment

$$P(0) = \delta^{-1}I \qquad (3.31)$$

where I is the identity matrix and δ is a small positive constant. The starting values of the equalizer coefficients can all be arbitrarily initialized to zero. After the initialization, equalizer adaptation may begin by sequential computations of equations (3.28), (3.29), (3.30) and (3.26) for each instant of time, $k = 1, 2, \ldots$.

The RLS algorithm typically converges an order of magnitude faster than the simple LMS algorithm [31]. For a T-spaced transversal equalizer containing N taps or a fractionally spaced equalizer of the same length, the standard RLS algorithm will converge in about $2N$ iterations. Furthermore, its rate of convergence is insensitive to the channel characteristics. However, the benefits are achieved at the expense of computational complexity, which is now on the order of N^2, as compared with the LMS algorithm, which has a complexity on the order of N. The increased computational complexity is due to the burden involving matrix multiplication in calculating the Kalman gain vector $K(k)$ and in updating the inverse correlation matrix $P(k)$. In addition, the recursive update procedure for the matrix $P(k)$ given in (3.30) has poor numerical properties; thus, the algorithm becomes sensitive to round-off noise when implemented with finite-precision arithmetic. In order to avoid instability due to the accumulation of round-off errors, binary representations of at least 16 bits are required [32, 33]. In the past, this has posed a problem to the hardware implementation of the algorithm. As a result, a related algorithm with better numerical properties has been derived to avoid the computation of $P(k)$ according to (3.30). The basic principle of this algorithm lies in the square-root factorization of $P(k)$ as $P(k) = S(k)S'(k)$, where $S(k)$ is a lower triangular matrix. The resulting algorithm, known as the square-root RLS algorithm, updates the matrix $S(k)$ directly without computing $P(k)$ explicitly and has a computational complexity on the order of N^2 [34]. Binary word lengths of 10 to 15 bits are required to ensure the stability of the square-root RLS algorithm [32, 33]. However, the requirement for intensive computation may render it computationally inefficient for equalizers with a large number of coefficients. A *fast* RLS algorithm, in which matrix multiplication involved in computing $P(k)$ and $K(k)$ is avoided, becomes more favored for such applications. For this algorithm, a minimum binary word length of 10 bits seems to be necessary [32, 33].

At present, the effects of round-off error accumulation on all RLS algorithms can be greatly reduced in two different ways. The first direct approach is to implement the algorithms with higher precision arithmetic; this is feasible since technological advancement has already made 32-bit DSP chips commercially available. The second method is to allow periodic reinitialization of the algorithms to avoid the accumulation of round-off error with time. In fact, this second method is not even necessary for TDMA systems carrying short-duration time-slots with a training sequence in each time-slot; in this case, complete initialization of the algorithm and training of the equalizer are carried out for the reception of every new time-slot.

A RLS algorithm is a favorable choice for equalizer adaptation in wireless communications because of its fast convergence rate and the insensitivity of its convergence rate to channel distortion. Since the benefits of off-line processing and double precision floating point arithmetic are available in computer simulation, the computational complexity of the algorithm and round-off error accumulation are of lesser concern. Hence, the standard RLS algorithm was implemented in this project for DFE adaptation.

3.2. Diversity Combining

The application of diversity combining to radio systems is an old idea, with the first experiment being reported in 1927 [18]. Diversity techniques require the availability of two or more transmission paths (branches) carrying the same message, and their success depends upon the degree to which signals on these paths are uncorrelated. However, it has been shown that diversity branches with a cross-correlation coefficient as high as 0.7 can still provide a substantial diversity advantage [11, 18]. A proper combination of signals from these transmission paths having approximately the same average signal strength can greatly reduce the severity of fading since the probability of having all branches experience deep fades at the same time is relatively low. Consequently, the reliability of a transmission link is improved.

Diversity paths can be produced in a number of different ways. One can employ different times, different frequencies or spatially separated antennas to achieve independent diversity paths. These techniques are known as frequency, time and space diversity, respectively. In frequency diversity, the desired signal is transmitted at several different carrier frequencies. For frequency separations of more than several times the coherence bandwidth, uncorrelated fading among signals at these frequencies can be achieved. With this method, diversity advantage is gained at the expense of channel bandwidth and hence it is not suitable for narrow band channels. Time diversity, in turn, uses several time instants to transmit the same signal. With successive transmissions separated by a time interval at least as large as the reciprocal of the fading bandwidth (*1/2f_D*, where f_D is the maximum Doppler shift), signals at different time instants will undergo uncorrelated fading [18]. Since the required time separation between independent fades is inversely proportional to the fading bandwidth, this method cannot be applied effectively to slowly fading indoor channels. Space diversity uses two or more antennas for transmission or reception of signals transmitted at the same time and same carrier frequency. The drawback is that the incorporation of multiple antennas within small handsets may not be practical. However, this technique is preferred for the applications considered in this thesis and will be discussed below. Other diversity techniques, which will not be discussed, include angle (directional), field component and polarization diversities [10, 11, 12, 18, 35].

3.2.1. Space diversity

Traditionally, diversity paths are realized by using two or more antennas for the transmission or reception of signals. This is known as space diversity in which the spacing of the antennas is chosen so that the individual signals are uncorrelated in their fading. Assuming that incoming waves have a uniform angular distribution at a mobile unit, the cross-correlation coefficient or the normalized envelope covariance, between the outputs of two diversity antennas, spaced at a distance L, is given by [35]

$$\rho(L) = J_0^2\left(\frac{2\pi L}{\lambda}\right) \tag{3.32}$$

where J_o is the zero-th order Bessel function of the first kind and λ is the wavelength of the carrier. This function is illustrated in Figure 3.7, which shows that the theoretical minimum space separation at which the two diversity branches are totally independent is 0.38λ. In practice, this separation may be larger, which is probably due to the lack of uniform angular distribution of the arrival waves [35]. However, it has also been reported that, for an antenna separation greater than $\lambda/4$, an average envelope cross-correlation of less than 0.1 is obtained for indoor wireless systems operating at about 1.7 GHz [36].

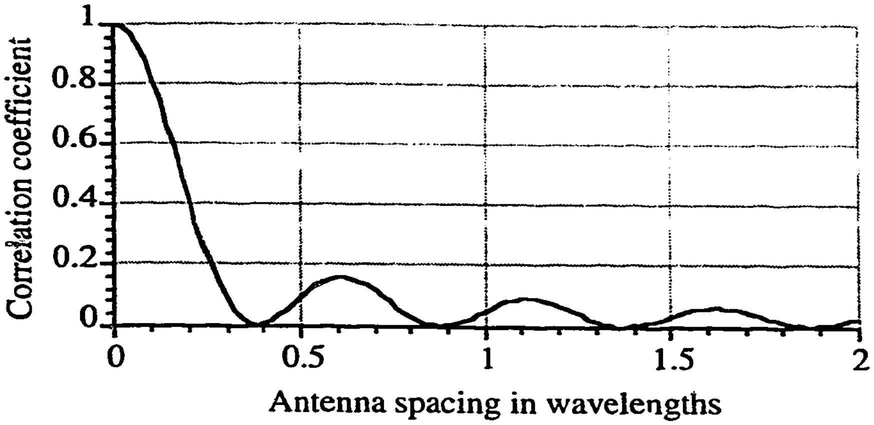

Fig. 3.7. Correlation coefficient between the outputs of two dipole antennas.

At the base station, a larger space-separation may be required to achieve a low correlation between the diversity branches. In mobile communications, signals transmitted from a mobile are scattered mainly near the vicinity of a mobile unit itself, as shown in Figure 3.8. Consequently, the scattered signals arrive at the base station within a confined angular sector (beam-width) instead of from all directions. A larger antenna spacing is required when the beam-width of the incoming signals becomes narrower [35]. In addition, for a given base station antenna spacing, the envelope cross-correlation between the diversity branches increases with antenna height. In other words, a larger antenna spacing is required for higher antennas in order to maintain a low cross-correlation between the diversity branches.

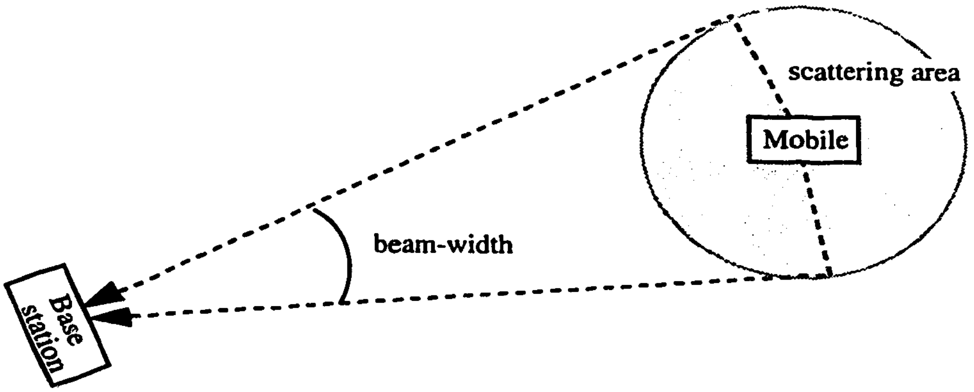

Fig. 3.8. Model of reverse link propagation.

3.2.2. Combining methods

In a diversity system, one or all diversity branches may be used for reception of the transmitted signal. There are basically four types of diversity combining methods: switching, selection, equal-gain combining and maximum ratio combining. In switching or selection diversity, only one of the diversity branches is used at any instant of time, whereas all branches are used in equal-gain and maximum ratio combining.

Diversity switching: This technique can be employed conveniently in a system using two diversity antennas. It requires only one measurement unit to monitor the signal power from a diversity branch used for reception. When the signal power from that branch is below a predetermined threshold, signal reception will be switched to the other diversity branch. The switch will stay in its new position as long as the signal power from a newly switched branch is above the threshold. If the new branch is also below the threshold, the switch may either remain in its new position or return to its previous position. To avoid frequent switching, a low predetermined threshold is used.

Diversity selection: In this scheme, all diversity branches are monitored simultaneously. The selection scheme can be driven by one of the two performance measures, namely received signal strength indication (RSSI) and error rate indication (ERRI). For the RSSI criterion, signal powers from the respective diversity branches are measured at the passband (i.e. at the intermediate frequency). The branch with the strongest power is then selected for demodulation, achieving passband diversity selection. On the other hand, ERRI-driven diversity selection requires a complete receiver for each diversity branch. It is a baseband diversity selection based on error checks of the demodulated signals, which can be achieved by error-detection coding. The performance of the ERRI criterion should be better than that of the RSSI criterion because signal power from a multipath channel does not necessarily reflect the overall quality of the link, as a stronger average power may be due to the existence of stronger trailing echoes, hence more severe ISI. However, the performance gain is achieved at the expense of receiver complexity.

Equal-gain combining: Instead of utilizing only one diversity branches at a time, a greater diversity potential can be exploited by combining signals from all the diversity branches.

Equal gain combining is a simple type of linear diversity combining in which signals from the individual diversity branches are co-phased and summed before detection. However, by having equal contributions from all the branches, it inevitably adds more noise to the resulting signal than could otherwise be obtained by ignoring those branches experiencing a severe fade.

Maximum ratio combining: In maximum ratio combining, individual signals from the diversity branches are weighted in proportion to their respective signal-to-noise power ratios, co-phased and summed. Assuming that the mean noise power is the same for each branch, it can be shown that the resulting output SNR is equal to the sum of individual branch SNRs.

It can be shown that, in terms of the output SNR, maximum ratio combining is the best combining scheme for combating Rayleigh fading with a combining gain of $10\log_{10}M$, where M is the number of diversity branches. Under the same conditions, the combining gains for equal gain combining and diversity selection (RSSI) are $10\log_{10}[1+\pi(M-1)/4]$ and $10\log_{10}\sum_{k=1}^{M} 1/k$, respectively [11, 18].

3.3. Summary

Adaptive channel equalization and diversity combining techniques were discussed. It has been explained how the performance of a simple linear transversal equalizer can be improved by reducing the tap spacing to less than a symbol-interval. Linear equalizers were seen to be inadequate for channels with severe amplitude distortions in their frequency response. Non-linear equalizers such as decision feedback equalizers have better performance for such applications. The coefficients of a DFE can either be calculated directly from the channel impulse response or adaptively adjusted by using a recursive algorithm in such a way that the mean squared error between the equalizer output and the desired output is minimized. For wireless applications, a fast converging recursive least squares algorithm is desired for equalizer adaptation. In addition, the convergence rate of a RLS algorithm is insensitive to channel distortion. In addition to ISI, multipath fading also contributes to severe performance degradation in a wireless system. Antenna space diversity is an efficient technique to combat multipath fading. In practice, an antenna spacing greater than the theoretical minimum of *0.38λ* is generally required to achieve independent diversity branches.

In the following chapter, schemes involving the integrated use of DFE and dual antenna diversity will be introduced.

4. INTEGRATING DECISION FEEDBACK EQUALIZATION WITH DUAL ANTENNA DIVERSITY

As described in Chapter 3, decision feedback equalization and antenna space diversity can effectively counteract intersymbol interference and multipath fading, respectively, in a radio channel. In this chapter, four performance enhancing schemes involving the combined use of these two techniques are discussed. The first three schemes incorporate decision feedback equalization with diversity selection, whereas the fourth scheme integrates decision feedback equalization with diversity combining through joint-parameter optimization.

4.1. Decision Feedback Equalization with Diversity Selection

A DFE can significantly improve the performance of a time dispersive radio link by reducing ISI. However, due to multipath fading, the power of the received signal can be reduced drastically when a deep fade is encountered, causing the desired signal to be masked by additive noise. This in turn, renders the DFE useless since it cannot identify the desired signal. To circumvent the problem, space diversity is used to increase the input signal-to-noise ratio to the DFE. In this section, three schemes involving dual antenna diversity selection and DFE, designed to improve DECT or DECT-type links, are presented. These three schemes differ in terms of the criterion used for antenna selection.

4.1.1. *Scheme I:* DFE with RSSI-driven diversity selection

Scheme I is the simplest of all the three schemes involving DFE with diversity selection, and it is well suited for DECT. DECT uses time division duplexing (TDD), with reverse link and forward link transmissions alternating at 5 ms intervals (half a frame duration) at the same carrier frequency. Consequently, the slowly fading indoor radio channel may be considered reciprocal. This allows diversity antennas to be implemented at a base station to acquire diversity gains for both links through diversity reception and transmission. The block diagram of a base station receiver is shown in Figure 4.1. The implementation of two diversity antennas, with a sufficiently large spacing between them to ensure uncorrelated signals from the diversity branches, at the base station is of less concern since constraints on the size of the base station are less severe. Here, the received signal strengths from the two independent antennas are monitored simultaneously. The antenna with the stronger received signal is selected for the forward link transmission, exploiting the reciprocity of the channel. Since the transmission to a portable occurs about half a frame duration after the reception of a time-slot from the correspondir g portable, a delay of about half a frame duration is incurred when using the selected antenna for transmission. We call this delay an antenna switching delay. The antenna with the stronger received signal is also used to receive the signal from the corresponding portable in the following frame. Thus, an antenna switching delay of about one frame duration (10 ms) is encountered in the reverse link transmission. This delay is not likely to cause noticeable performance degradation since the indoor channel is quasi-stationary. The selected signal is fed to an adaptive passband DFE.

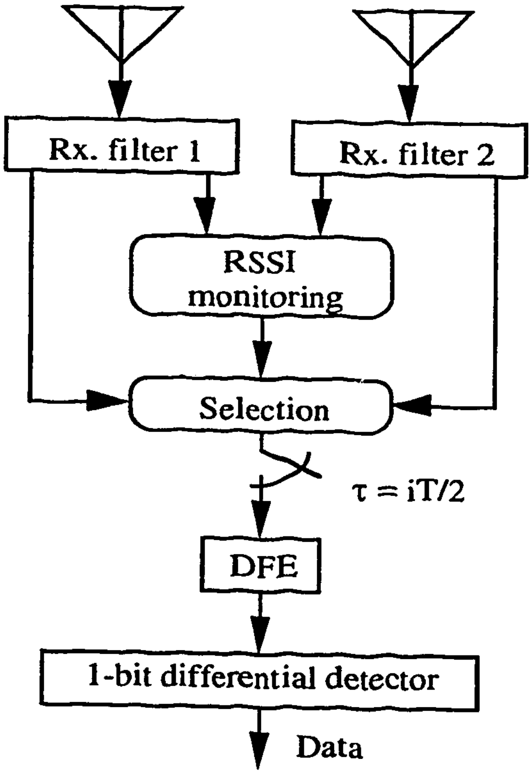

Fig. 4.1. DFE with RSSI-driven diversity selection.

The passband DFE, which is similar to the one shown in Figure 3.5, consists of a *T/2*-spaced feedforward transversal filter (FFF) and a *T*-spaced feedback transversal filter, where *T* is the bit interval. The last tap of the FFF is the reference tap of the DFE. The equalizer coefficients are adaptively adjusted using the standard RLS algorithm. Issues regarding the length of the filters as well as equalizer training will be addressed in Chapter 5. The output of the DFE is then passed to a one-bit differential detector, where the transmitted data is retrieved.

The portable, benefitting from the diversity transmission at the base station, does not require diversity antennas for reception of signals from the base station; it requires only one DFE to achieve a performance comparable to that of the reverse link.

4.1.2. *Scheme II:* DFE with ERRI-driven diversity selection

In *Scheme I*, diversity selection is driven solely by signal powers received from the two diversity branches at the base station. However, the received signal power may not necessarily reflect the quality of the link because a larger received power may be due to stronger trailing multipaths, which the DFE tries to cancel. In such a case, the recoverable signal power will be smaller than that of the interfering paths. A better alternative, which is used in *Scheme II*, selects the appropriate diversity branch according to an error rate indication.

Figure 4.2 shows the block diagram of a base station receiver employing *Scheme II*. In order to determine the link quality using an error rate indication, it requires two separate receivers, one for each diversity branch, to process and detect the signal at the base station. Similar to *Scheme I*, diversity antennas are implemented only at the base station to achieve diversity reception and transmission. Only one antenna and one DFE are required at the portable to obtain a comparable performance on the forward link. Also, the DFEs used in *Scheme II* are identical to those used in *Scheme I*. In practice, the 16-bit cyclic redundancy check (CRC) code in the DECT time-slot is used for error detection and the selection criterion can then be based on monitoring CRC error checks in both branches. A cyclic code is very effective for the detection of transmission errors. An *l*-bit cyclic code is capable of detecting any error burst of length *l* or less, and its probability of detecting error bursts of length l+1 or greater is at least $1\text{-}2^{-(l-1)}$ [37]. Therefore, the error-detection probability of a 16-bit CRC code is at least 0.999969. However, the CRC code is not actually implemented in this work, and error checks for *Scheme II* are achieved by comparing the detected bits with the known transmitted bits used in the simulations. In other words, the error-detection probability is assumed to be 1. For computer simulations, the selection scheme works in such a way that if there are one or more data-bit errors in one branch and no errors in the other branch, then the branch with no error is selected; if there are one or more errors in both branches or there is no error in either branch, then the branch selected in the previous frame continues to be used. Hence, it is necessary to store a full time-slot of demodulated data from each diversity branch before branch selection is effected for each reverse link transmission. As a result, there is no antenna switching delay for the reverse link transmission. The antenna of the selected branch is also used for the forward link transmission, in which an inherent antenna switching delay of about half a frame duration (5 ms) is incurred.

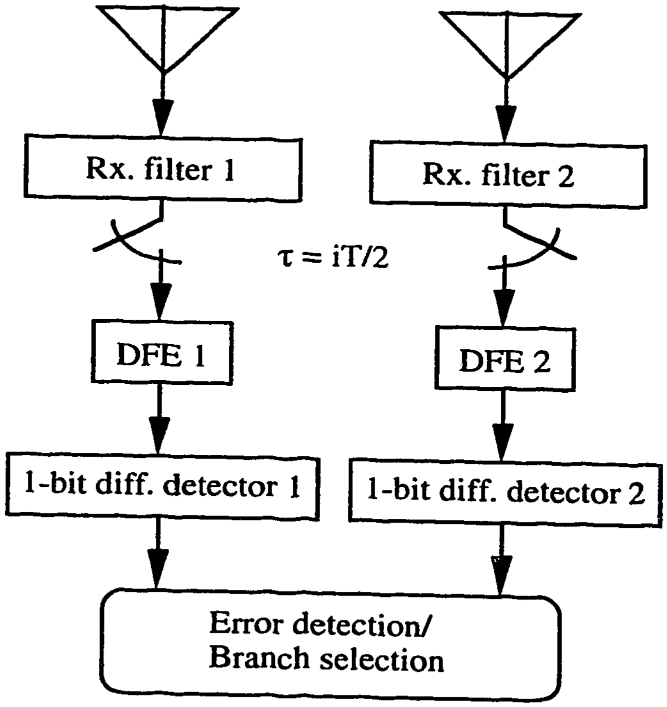

Fig. 4.2. DFE with ERRI-driven diversity selection.

4.1.3. *Scheme III:* DFE with MSE-driven diversity selection

In adaptive decision feedback equalization, the error signal, which is the difference between the input and the output of the decision device of a DFE, is used by a RLS algorithm to adjust the equalizer coefficients. The mean square of this error signal, to a certain extent, reflects the quality of the transmission link, and it is used as a parameter for diversity selection in *Scheme III.*

Similar to *Schemes I* and *II*, *Scheme III* requires diversity antennas to be implemented only at the base station to acquire diversity gains for both forward and reverse links by exploiting the reciprocity of the channel. Figure 4.3 shows the block diagram of a base station receiver using *Scheme III.*

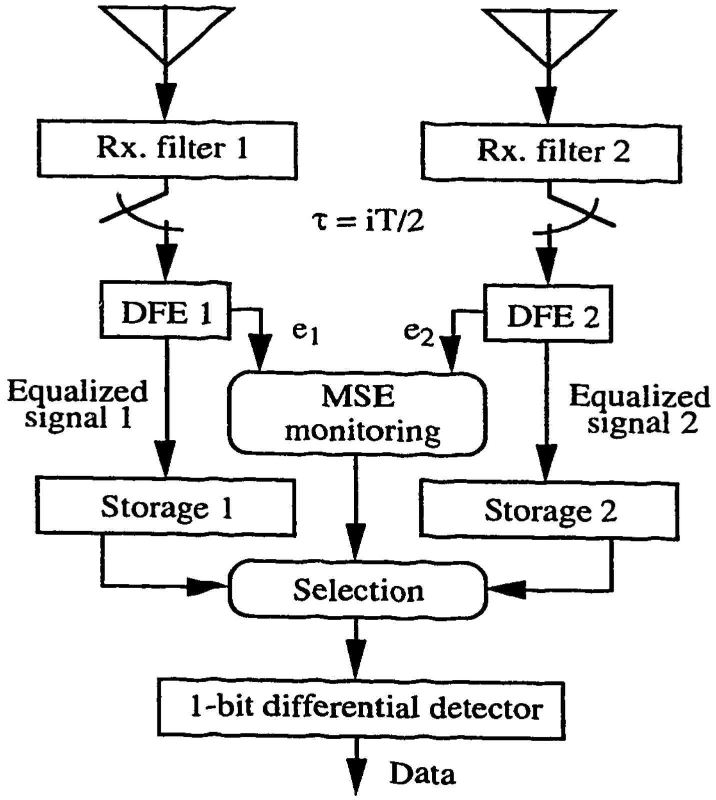

Fig. 4.3. DFE with MSE-driven diversity selection.

A DFE is also required for each diversity branch at the base station. Unlike *Scheme II*, it requires only one differential detector, and its diversity branch selection is based on monitoring the mean squared errors (MSEs) from the two DFEs. For each reverse link transmission, the equalized signal from each branch is temporarily stored, while error signals from individual DFEs are averaged over the entire data sequence of each received time-slot. The equalized signal from the branch with the lower mean squared error is then selected for detection. Consequently, there is no antenna switching delay for the reverse link transmission. The antenna of the selected branch is also used for transmission to the corresponding portable, in which case an antenna switching delay of about half a frame duration is encountered.

4.2. Decision Feedback Equalization with Diversity Combining

In the previous three diversity/equalization schemes, the transmitted data are retrieved only from one of the two diversity branches. As mentioned in the previous chapter, diversity gain can be further increased by combining signals from all the diversity branches. This section discusses a scheme which integrates decision feedback equalization with dual antenna diversity combining through joint-parameter optimization.

4.2.1. *Scheme IV:* Diversity combining and DFE with joint parameter optimization

The block diagram of a receiver using *Scheme IV* is shown in Figure 4.4. Unlike the previous three schemes, *Scheme IV* requires diversity antennas to be implemented at the base station as well as at the portable. Equalization is achieved by using two *T/2*-spaced feedforward transversal filters (FFFs), one for each diversity branch, and a single *T*-spaced feedback transversal filter (FBF). Outputs of the FFFs are added to realize diversity combining, and the FBF's output is then subtracted from the sum. The coefficients of all three filters are jointly optimized using a single RLS algorithm by concatenating the coefficients of the filters into one single vector and the corresponding tap inputs of the filters into another vector as follows:

$$\begin{aligned} w &= \left[a_{1,0}, a_{1,1}, \ldots a_{1,F-1}, \quad a_{2,0}, a_{2,1}, \ldots a_{2,F-1}, \quad -b_1, -b_2, \ldots, -b_B\right]^T \\ u(k) &= \left[r_{1,0}(k), r_{1,1}(k), \ldots, r_{1,F-1}(k), \quad r_{2,0}(k), r_{2,1}(k), \ldots, r_{2,F-1}(k), \right. \\ &\qquad \left. \hat{d}(k-1), \hat{d}(k-2), \ldots, \hat{d}(k-B)\right]^T \end{aligned} \tag{4.1}$$

where $a_{i,j}$ is the j-th tap coefficient of the FFF of length F in branch i, b_j is the j-th tap coefficient of the FBF of length B, $r_{i,j}(k)$ is the j-th tap input of the FFF in branch i and $\hat{d}(k-j)$ is the j-th tap input of the FBF at time k. The input to the decision device can then be expressed as

$$y(k) = w^H u(k) \tag{4.2}$$

By jointly optimizing the two feedforward filters, co-phasing of the signals from the two diversity branches is inherently effected. Presumably, the algorithm can also adjust the filters' coefficients automatically in such a way that maximum ratio combining is synthesized by minimizing the contribution from the branch experiencing deep fades. Theoretically, it is an optimum dual-antenna diversity combining and equalization scheme [38]. However, the improvement is gained at the expense of increased complexity at the portable receiver.

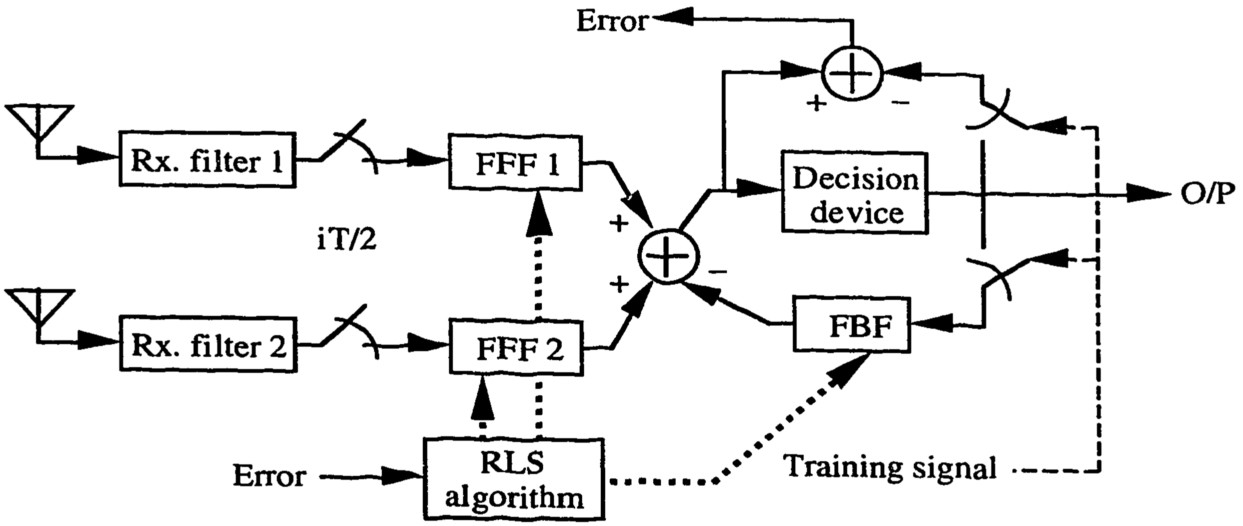

Fig. 4.4. Diversity Combining and DFE with joint-parameter optimization.

4.3. Summary

Four schemes involving dual antenna diversity and decision feedback equalization, which are intended to enhance the performance of DECT or DECT-type systems, have been discussed in this chapter. *Schemes I, II* and *III* feature the combination of DFE with diversity selection, whereas *Scheme IV* integrates diversity combining and decision feedback equalization into a single entity through joint-parameter optimization.

For *Schemes I, II* and *III*, diversity antennas are implemented only at the base station to obtain diversity gain for both the forward and reverse links, with diversity selection driven by RSSI, ERRI and MSE criteria, respectively. *Scheme I* incurs an antenna switching delay of about one frame duration for the reverse link transmission, whereas no such delay is encountered by *Schemes II* and *III*. Since branch selection is determined during the reverse link transmission, and the antenna of the selected branch is then used for transmission to the corresponding portable, all three schemes incur an antenna switching delay of about half a frame duration for the forward link transmission.

Instead of selecting a signal from the better branch, *Scheme IV* combines and equalizes the signals from both diversity branches. Thus, diversity antennas should also be implemented at the portable. As a result, this scheme may only be suitable for larger mobile terminals which can accommodate two diversity antennas.

The performance of these four schemes has been evaluated through extensive computer simulations, and the simulation results are presented in Chapter 5.

5. SIMULATIONS AND RESULTS

For complex systems such as wireless communication systems, performance analysis using pure analytical means can be extremely difficult and abstract, if not impossible. Computer simulation, which is a relatively simple process, has become indispensable for the development and performance evaluation of communication systems prior to costly and time consuming hardware implementation. This chapter is concerned with the performance evaluation by computer simulation of the DECT baseline system and the related performance enhancing techniques discussed in the previous chapters.

A commonly used performance measure for a digital communication system is its average bit error rate (BER). In this project, Monte Carlo simulation is used to estimate the BER, because of its relative simplicity. This method requires no assumptions about the input processes and analysis of the system [39]. Comparing the output sequence with the known input sequence at some relative delay allows the BER of the simulated system to be calculated empirically. The estimated BER, $\hat{p}$, is the ratio of the total number of bit errors observed, n, to the total number of bits processed, N. As N approaches infinity, the estimator $\hat{p}$ tends to a normal distribution with mean p and variance $p(1-p)/N$. It has also been shown [39] that, if one hundred bit errors are counted, this distribution produces a 95% confidence interval of about $(1.25\hat{p}\ ,\ 0.8\hat{p})$, implying that the estimated BER is within 25% of the actual BER 95% of the time. However, the minimum number of bit errors should not be the only parameter considered to obtain a reliable estimated BER of a wireless system in the presence of slow fading and AWGN. It is also important to run sufficiently long simulation sequences to ensure that several deep fades are encountered in the process of estimating the average BER.

In this project, the maximum Doppler shift of a Rayleigh fading channel is set at 5 Hz and the channel characteristics are assumed to be constant within one time-slot duration. Also, only one time-slot in each DECT frame is considered and a minimum of 25,000 frames are used in each simulation process. This corresponds to successive transmissions, at time intervals of 10 ms (one DECT frame duration), from a portable to a base station (or vice versa) over a time duration of 250 seconds in a real situation. The number of time-slots encountering fades of any level can be found through computer simulation by generating the Rayleigh fading signal samples (corresponding to the Rayleigh fading carrier) at intervals of 10 ms. The simulation results for the minimum, maximum and average number of time-slots encountering fades of various depths for a sequence of 25,000 time-slots transmitted, obtained over 2000 consecutive trials of simulation, are summarized in Table 5.1. The fading levels are defined relative to the rms value of the Rayleigh fading envelope.

Fading level, *R* (dB)	Min. number of slots below *R*	Max. number of slots below *R*	Avg. number of slots below *R*
-10	2032	2588	2292
-15	585	876	746
-20	175	313	238
-30	9	42	24

Table 5.1. Number of slots in fades of various depths for 25,000 slots transmitted.

Executing such a long sequence in each simulation process using multi-purpose software packages, such as the Signal Processing Worksystem (SPW™) and Block Oriented System Simulator (BOSS™), on a DEC-3000 workstation may require excessively long simulation run times, due to their large overheads. Therefore, customized C programs have been designed for such purposes in this project.

Section 5.1 details the simulation process for the DECT baseline system, including various channel models. The simulation of the DECT system employing diversity selection is given in Section 5.2, whereas that of the system using a DFE only is presented in Section 5.3. The performance of the four combined diversity-equalization schemes outlined in Chapter 4 is simulated in Section 5.4. In each of these sections, respective simulation results are discussed and compared. Finally, a summary is presented in Section 5.5.

5.1. Simulation and Performance Evaluation of the Baseline System

It is a common practice to decompose a simulation system model into a number of functional blocks. Each functional block is coded as a separate C function, which forms the basic building block for more complicated system models involving different performance enhancing schemes. The descriptions of different functional blocks are given, followed by discussions of the simulation results.

5.1.1a. Baseline system

Figure 5.1 shows the complex baseband equivalent baseline simulation model.

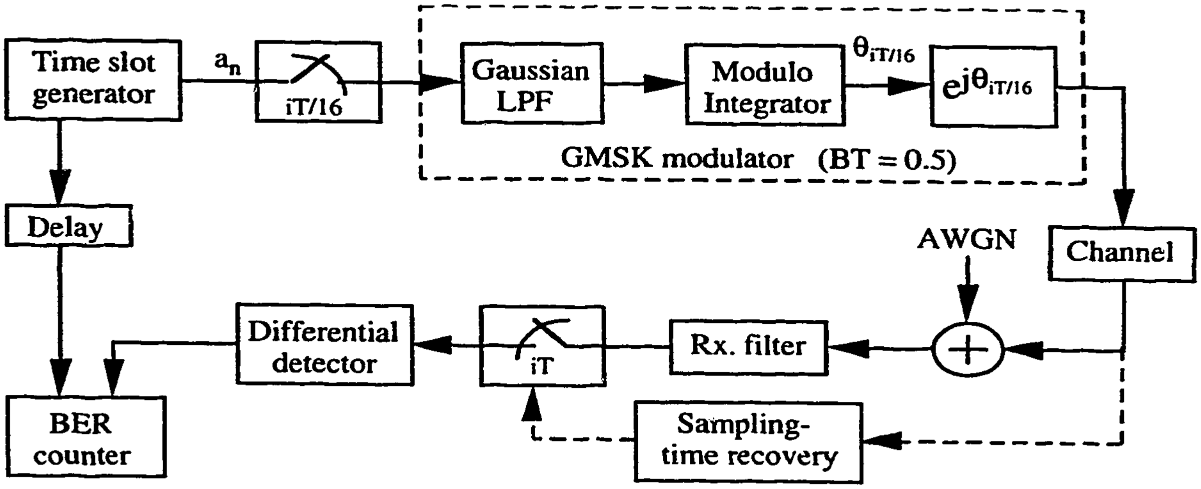

Fig. 5.1. A simulation model of the baseline system.

A time-slot generated at the transmitter resembles the DECT time-slot except that it does not contain the control-field and the CRC check-bits. The data bits in each time-slot are generated by a random number generator. The NRZ sequence from the time-slot generator is then fed to a sampler which oversamples each bit by sixteen times. As discussed in Chapter 2, this small sampling interval is necessary for the precise and easy modeling of a GMSK modulator. In addition, it also facilitates flexible adjustment of the rms delay spread of a channel, which is modeled by a tapped delay line with tap spacings equal to integer multiples of the sampling interval. The transmission bit rate specified in the DECT standard is 1.152 Mb/s This corresponds to a bit interval T of 868.05555··· ns. This irrational bit interval could be a source of a cumulative truncation error. Hence, it was decided to use a bit interval of 800 ns so that the sample interval, T_s, is exactly 50 ns. A bit interval of 868 ns could also be used, which would result in a sampling interval of 54.25 ns. The sample sequence is then fed to a GMSK modulator consisting of a Gaussian lowpass filter with BT of 0.5, a modulo integrator and a real-to-complex converter. A detailed description of the GMSK modulator model has been presented in Chapter 2. The Gaussian pulse-shaping filter is modeled as a FIR filter with the impulse response given by equation (2.27). A total of 51 taps are used in the FIR filter to ensure that the tails of the impulse response that are truncated have negligible amplitudes. The pulse-shaped sequence of samples is then applied to the modulo integrator, which is modeled according to a numerical integration method using a second order polynomial approximation as described in Chapter 2. A model of the modulo integrator is depicted in Figure 2.11. Output of the modulo integrator is the information carrying phase. The real-to-complex converter then transforms this signal phase into a complex baseband equivalent GMSK signal. The signal is then passed through a channel. The different channel models used in this project will be discussed later. In the case of a slowly fading channel, the characteristics of the channel are assumed to be constant within one time-slot interval. Zero-mean additive white Gaussian noise (AWGN), generated by a Gaussian random number generator, is added to the channel output. The noise voltage is adjusted to obtain a desired average signal-to-noise power ratio (SNR) at the output of the receiver filter. In digital communications, the performance measure in terms of BER is usually evaluated against E_b/N_o instead of against the average SNR, where E_b is the energy per bit and N_o is the one-sided noise power spectral density. The average received SNR can be shown to be directly related to E_b/N_o. The transmitted GMSK signal, with a constant amplitude of $\sqrt{2E_b/T}$, has an average power of E_b/T. For the channel with an average gain of unity, the average power of the distorted GMSK signal at its output is also E_b/T. The average SNR at the output of a receiver filter can then be expressed :

$$\mathrm{SNR} = \frac{E_b/T}{B_n N_o}$$

where B_n is the noise equivalent bandwidth of the receiver filter. The receiver filter used in the simulation is a Gaussian filter with an equivalent lowpass transfer function given by equation (2.31). Its optimum normalized 3 dB bandwidth, B_rT, is 1.1 [23]. Hence, given that the noise equivalent bandwidth of a Gaussian filter is $1.06B_r$, the average SNR can be approximated as

$$\text{SNR} \approx \frac{E_b}{1.1N_o}$$

For simplicity, the received signal is passed to a bit-rate sampler before feeding it to a one-bit differential detector, and the optimum sampling time is derived from the noiseless channel output. In actual system implementation, bit-rate sampling only occurs after the one-bit differential detector, and the related timing is recovered from the output of the detector. It is necessary to derive the optimum sampling time at the beginning of every new time-slot received because the instantaneous delay of a frequency selective fading channel, and consequently the optimum sampling phase, varies from slot to slot when the channel is non-stationary. In the simulation, the sampling time is recovered from the distorted but noiseless GMSK signal by tracking the phase of the signal associated with the clock-recovery bit sequence located at the beginning of every time-slot, as shown in Figure 5.2. Once the optimum sampling time is obtained, the received signal is sampled at T-intervals. The transmitted data are retrieved through one-bit differential detection. The retrieved data are then compared with the known transmitted data to determine the BER of the system.

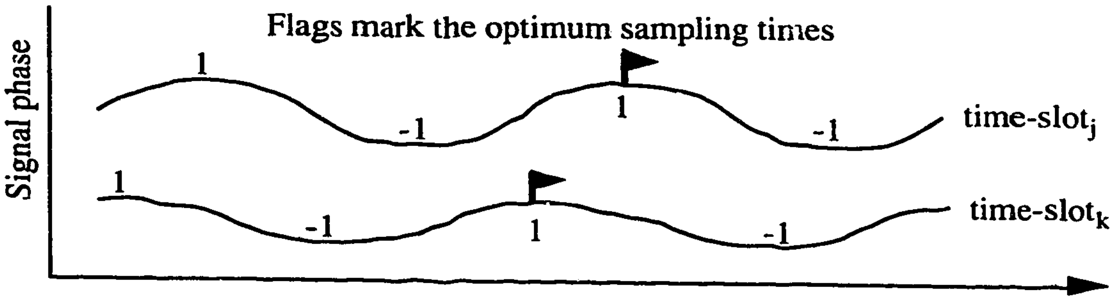

Fig. 5.2. Signal phase representing the clock-recovery bit-sequence with shifted optimum sampling times at different channel conditions.

5.1.1b. Channel models

The characteristics of radio channels and the related modeling issues have been discussed in Chapter 2. Several channel models are considered in this project; they include

i. AWGN channel,

ii. Rayleigh flat-fading channel, and

iii. Frequency selective fading channels.

An AWGN channel model simply consists of a single unity gain, non-fading path to whose output white Gaussian noise is added. On the other hand, the Rayleigh flat-fading channel comprises a single Rayleigh-faded path. The fading characteristics are emulated by Jakes' fading simulator, as shown in Figure 2.3. As discussed in Chapter 2, a frequency selective channel can be modeled by a tapped delay line, with tap spacings (relative delay) equal to integer multiples of the sampling interval, T_S. For frequency selective fading channels, several models are considered; they are classified as follows, according to the number of discrete paths, power delay profiles and fading characteristics of individual paths.

- *Two-path Rayleigh fading channel:* It consists of two independently Rayleigh fading paths (taps) with equal average powers. Thus, the average rms delay spread (usually referred as rms delay spread for convenience) is half the relative delay between the two paths. As described in Chapter 2, an independently fading signal can be obtained by assigning a different starting time to a Jakes' fader for each path.
- *Six-path Rayleigh (Rician) fading channel:* It is modeled by a tapped delay line with an equal tap-spacing and an average impulse response having an exponentially decaying power profile. The tap spacings (relative delays) are adjusted for different rms delay spreads while the relative average power distribution among the discrete paths remains unchanged, as outlined in Table 5.2. The first path is either Rician or Rayleigh fading and the remaining five paths are Rayleigh fading. The K factor (power ratio of the direct component to the diffused Rayleigh faded component) for the Rician fading path is chosen to be 1.5. Fading among the six paths is uncorrelated. Hence, the average power gain of the channel is the sum of average powers of the individual paths, and it is normalized to unity.

	Taps	1	2	3	4	5	6
rms delay spread	Rel. power (dB)	0	-2.88	-7.77	-11.65	-15.54	-19.42
0.0625T	Rel. delay (T_S)	0	1	2	3	4	5
0.1250T	Rel. delay (T_S)	0	2	4	6	8	10
0.1875T	Rel. delay (T_S)	0	3	6	9	12	15
0.2500T	Rel. delay (T_S)	0	4	8	12	16	20
0.3125T	Rel. delay (T_S)	0	5	10	15	20	25
0.3750T	Rel. delay (T_S)	0	6	12	18	24	30

Table 5.2. Average power distribution and relative delays of a six-path fading channel.

- *Seven-path Rayleigh fading channel:* It resembles an experimentally derived indoor channel model provided by The Joint Technical Committee on wireless access [40]. The power distribution among the different discrete paths and their relative delays are detailed in Table 5.3. The relative average powers of the discrete paths follow the exact values suggested by the Committee. However, the relative time delays of some of the discrete paths are slightly shorter than those recommended. Consequently, the rms delay spread for this simulation model is 0.1663T instead of the suggested value of 0.1896T (T = 800 ns).

Taps	1	2	3	4	5	6	7
Rel. delay (T_S)	0	1	3	4	7	10	14
Rel. power (dB)	-4.6	0	-4.3	-6.5	-3.0	-15.2	-21.7

Table 5.3. Average power distribution and relative delays of a seven-path fading channel.

For this channel model, two cases of Rayleigh fading among the discrete paths are considered. The first case assumes that fading among the seven paths is uncorrelated, whereas the second case considers correlated fading among the discrete paths. Since well-established experimental results are not available, it may be assumed that adjacent paths may have a higher degree of envelope cross-correlation between them than those that are further apart. Various degrees of envelope cross-correlation between discrete paths can be obtained from Jakes' Rayleigh fading simulator by assigning the oscillators of faders for different paths with the same starting time and unique phase shifts according to equation (2.13). With M_o in (2.13) equals 25, the following envelope cross-correlation coefficients between paths m and n, ρ_{mn}, are obtained for the seven-path channel model with correlated fading.

$$\begin{aligned}
&\rho_{12} = \rho_{23} = \rho_{34} = \rho_{45} = \rho_{56} = \rho_{67} = 0.93\\
&\rho_{13} = \rho_{24} = \rho_{35} = \rho_{46} = \rho_{57} = 0.76\\
&\rho_{14} = \rho_{25} = \rho_{36} = \rho_{47} = 0.53\\
&\rho_{15} = \rho_{26} = \rho_{37} = 0.29\\
&\rho_{16} = \rho_{27} = 0.10\\
&\rho_{17} = -0.0097
\end{aligned} \tag{5.1}$$

The average power gain of this channel model is also normalized to unity. It should be noted that, when the discrete paths are not independent, the average gain of the channel is no longer equal to the sum of average powers of the individual paths. The actual gain of such a channel can be evaluated during simulation.

The maximum Doppler shift of the fading channels mentioned above is set at 5 Hz to reflect a slowly fading indoor channel. For a DECT link, this maximum Doppler shift corresponds to a portable speed of 0.8 m/s. These channel models are used in computer simulations to evaluate the performance of the baseline system s well as the corresponding system structures involving various performance enhancing techniques.

5.1.2. Simulation results

The performance of the baseline system in the different channels previously described was simulated.

Figure 5.3 shows the system performance in an AWGN channel. The performance degradation is only due to the noise added to the system. The simulation result agrees with the analytical result presented by Simon and Wang for one-bit-differential detection of the GMSK signal with BT and B_rT equal to 0.5 and 1.1, respectively, [23].

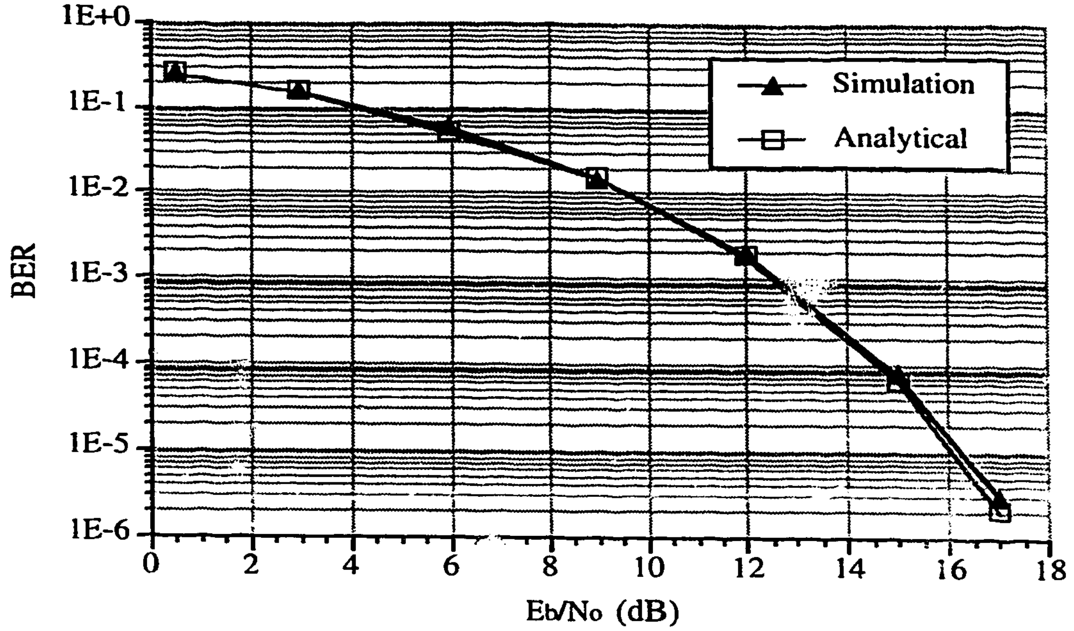

Fig. 5.3. Performance of the baseline system in an AWGN channel.

Figure 5.4 illustrates the system performance in a Rayleigh flat-fading channel. The simulation result again shows good agreement with the analytical result presented in [23].

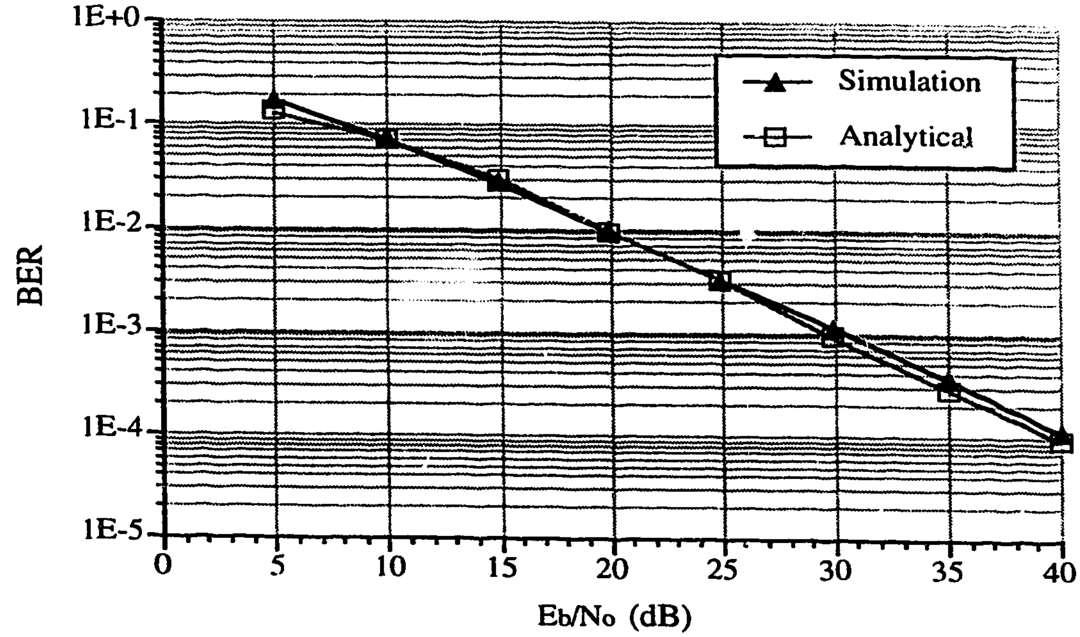

Fig. 5.4. Performance of the baseline system in a Rayleigh flat-fading channel.

The above results demonstrate that Rayleigh fading can cause severe performance degradation. As compared with the system performance in the AWGN channel, an additional 17 dB E_b/N_o is required to attain a BER of 10^{-3}, a performance acceptable for wireless voice telephony, in the Rayleigh flat-fading channel.

The effects of channel dispersion on system performance were investigated using the various frequency-selective fading channel models. The performance of the system in the two-path and six-path Rayleigh fading channels is demonstrated in Figure 5.5. Fading among the discrete paths was assumed to be uncorrelated. Both channel models show good agreement. The system is seen to require an E_b/N_o of about 34 dB to reach the target BER of 10^{-3} at a rms delay spread, σ_τ, of $0.0625T$ (this corresponds to a rms delay spread of 54 ns in a DECT link where $T = 868$ ns). The irreducible error floor due to ISI increases rapidly with σ_τ. The system fails to reach the target BER of 10^{-3} even with an E_b/N_o of 40 dB when σ_τ reaches $0.1250T$.

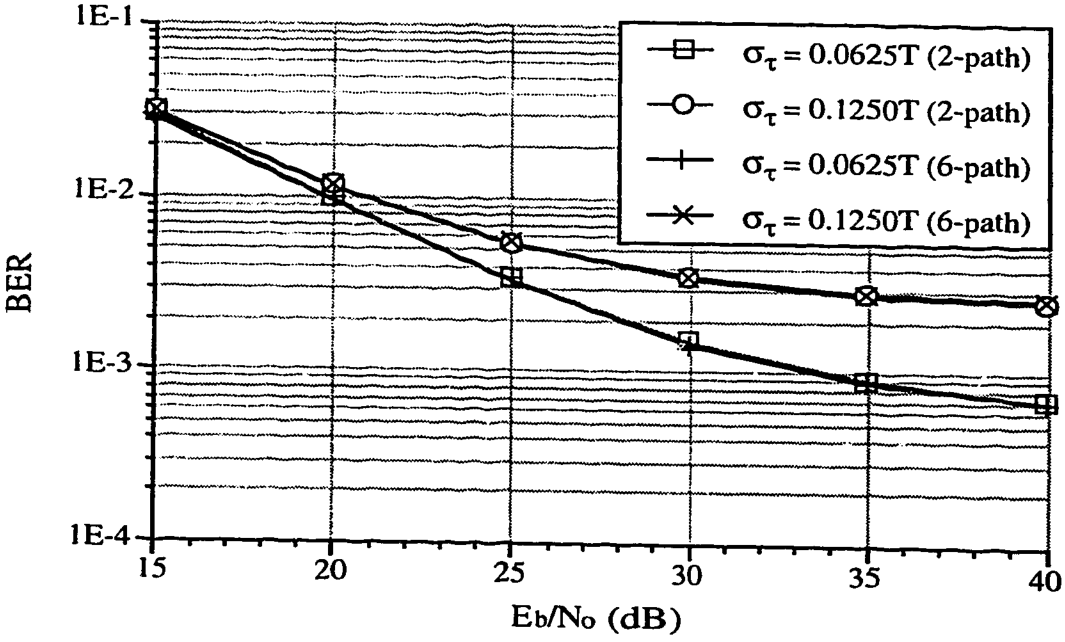

Fig. 5.5. Performance of the baseline system in two-path and six-path Rayleigh fading channels.

The performance of the system, operating in a certain indoor environment where a direct propagation path exists, was also simulated. To emulate such a channel, the six-path

Rician fading channel model, with Rician fading for the first path ($K \doteq 1.5$) and uncorrelated Rayleigh fading for the trailing paths, was used. The simulation results, in comparison with those obtained with the six-path Rayleigh fading channel, are shown in Figure 5.6. For the rms delay spreads considered, the results show that the link performance is better in the presence of a direct path component. In both channel models, the first path is the dominant path with the strongest average power. For the Rayleigh fading channel, the first path, just like the other Rayleigh fading paths, may encounter very deep fades. On the other hand, deep fades of similar magnitudes do not occur in the first path of the Rician fading channel due to the presence of a direct component in addition to the diffused Rayleigh faded component. Consequently, error bursts due to severe deep fades in the dominant path can be avoided in the Rician fading channel. However, channel dispersion still has a detrimental effect on the system. With the Rician fading channel, the system performance also fails to reach a BER of 10^{-3} at a rms delay spread of 0.1250*T*.

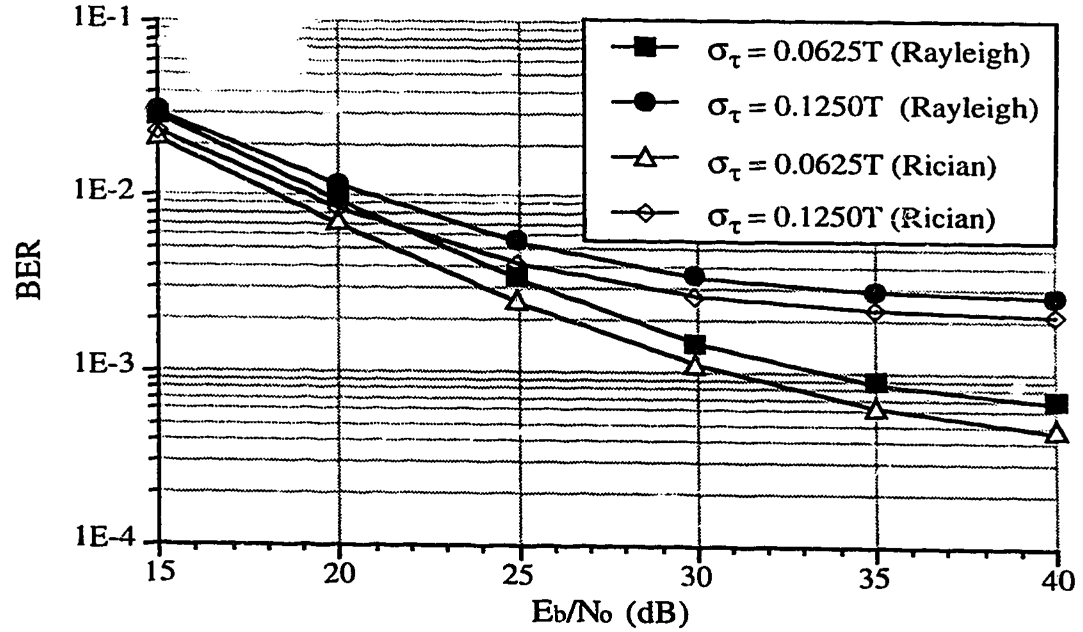

Fig. 5.6. Performance of the baseline system in a six-path Rayleigh (Rician) fading channel.

For the simulation results presented above, fading among discrete paths was assumed to be uncorrelated. However, in reality, certain degrees of envelope cross-correlation between resolvable paths exist. This effect was investigated using the seven-path channel model (with a fixed average rms delay spread of 0.1663T) previously described. For this channel model, discrete paths with uncorrelated and correlated Rayleigh fading, were considered (see (5.1) for envelope cross-correlation coefficients between correlated fading paths). The simulation results presented in Figure 5.7 show that the system performs better in the channel with correlated paths. This behavior may be explained as follows. For the channel with uncorrelated paths, individual paths fade independently and there may exist instants when a few of the discrete paths acquire comparable instantaneous powers, which greatly increases the instantaneous rms delay spread of the channel beyond its ensemble average rms value determined by the given average power delay profile. This mechanism is less likely to occur in the channel with correlated paths since adjacent paths tend to fade together, avoiding a large deviation of the instantaneous rms delay spread of the channel from its average rms value.

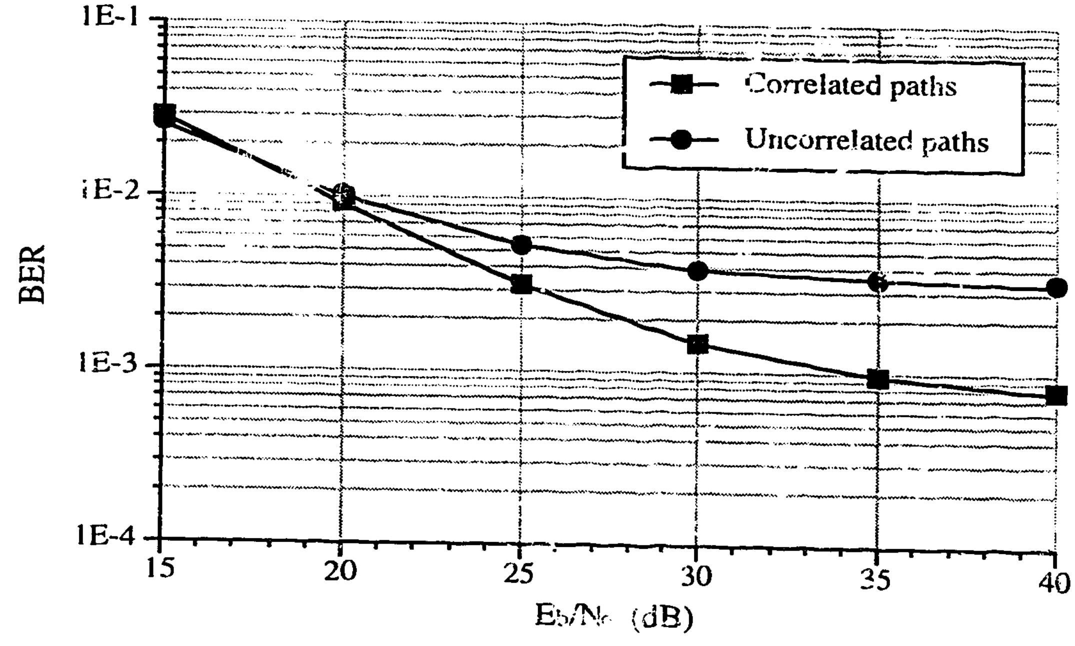

Fig. 5.7. Performance of the baseline system in a seven-path Rayleigh fading channel.

The baseline simulation results presented above have shown that the DECT link can be severely affected by channel fading and dispersion. According to the simulation results obtained with the two-path and six-path Rayleigh (Rician) fading channels, the system fails to reach the required BER even at a rms delay spread of 0.1250T. To enhance the system performance, the application of diversity selection has been considered, and the related simulation models and results are presented in the following section.

5.2. Simulation and Performance Evaluation of the Baseline System Employing Diversity Selection

The application of two diversity selection techniques to the DECT system, driven by different selection criteria, was investigated. For the first method, antenna selection is based on monitoring the received signal strength (RSSI-driven diversity). The second method uses an error rate indication as its selection criterion (ERRI-driven diversity). They are similar to *Schemes I* and *II* described in Chapter 4, except that decision feedback equalization is excluded. The performance of these two diversity schemes was evaluated through computer simulations of the DECT reverse link.

5.2.1. System with RSSI-driven diversity selection

Figure 5.8 shows the reverse link simulation model of the baseline system employing RSSI-driven diversity selection.

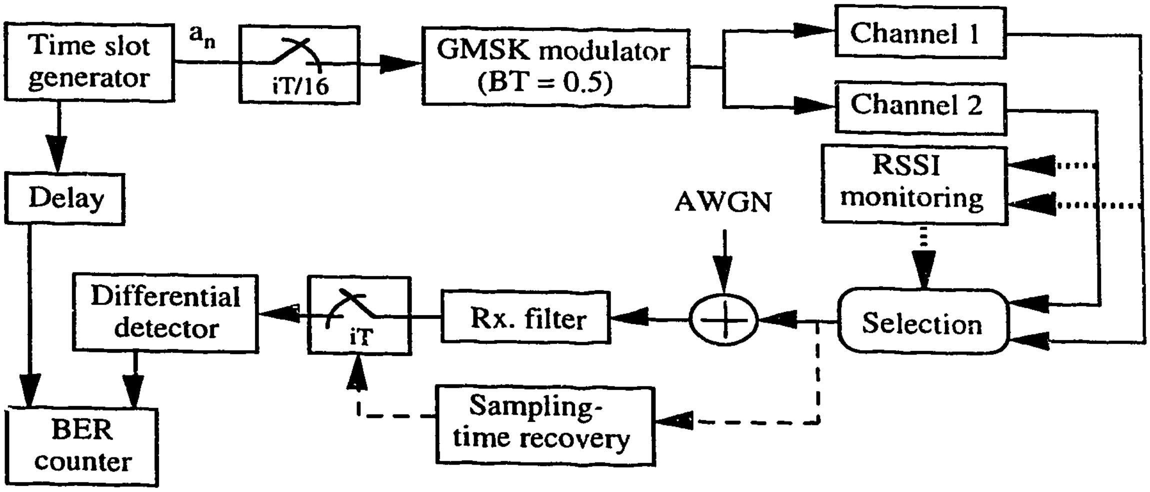

Fig. 5.8. A simulation model of the system employing RSSI-driven diversity selection (reverse link).

The output of the GMSK modulator is fed to two independently fading channels representing the diversity paths in an indoor propagation medium. The channels' characteristics are assumed to be constant within one time-slot interval; the channel parameters are updated once a full time-slot is received, reflecting the change of the channel characteristics over one frame duration between successive transmissions from the same portable. For simplicity, signal strengths are measured at the channels' outputs; in practice, the sum of the average signal and noise powers is measured at the output of each of the receiver filters (one for each diversity branch). The above simplified model is based on the assumption that, for the reception of a full time-slot, the average noise powers on both diversity branches are the same, and thus the difference in the average received powers at the outputs of the receiver filters is only due to the difference in the actual signal powers. In the simulation, the branch with the stronger received signal measured in the previous frame is used for reception of the current time-slot. White Gaussian noise is added to the signal from the selected branch before filtering. The sampling time is recovered from the noiseless signal at the beginning of each time-slot. The received and filtered signal is sampled and passed to a one-bit differential detector. The detected data bits are then compared with the originally transmitted data to determine the BER of the system.

5.2.2. System with ERRI-driven diversity selection

The reverse link simulation model of the baseline system employing ERRI-driven diversity is shown in Figure 5.9. The functional blocks representing the transmitter and the channels are identical to those discussed previously. There are two separate receivers, one for each diversity branch, at the base station. Each receiver branch is identical to the one described in Section 5.1.1a. Bit-error detection on each diversity branch is effected by comparing the respective detected data bits with the known transmitted bits. If there is no bit error in the time-slot received from one of the two branches, then the number of accumulated bit errors in the BER counter remains unchanged, simulating a situation in which the branch with no error is selected for retrieving the transmitted information. If there are bit errors in both branches, then the number of accumulated errors in the BER counter is increased by the number of bit errors occurring in the current time-slot received from the branch selected in the previous frame. This corresponds to the practical situation of bit errors occuring in both branches, for which the branch selected in the previous frame continues to be used for retrieving the transmitted information regardless of the actual number of errors, which is not known.

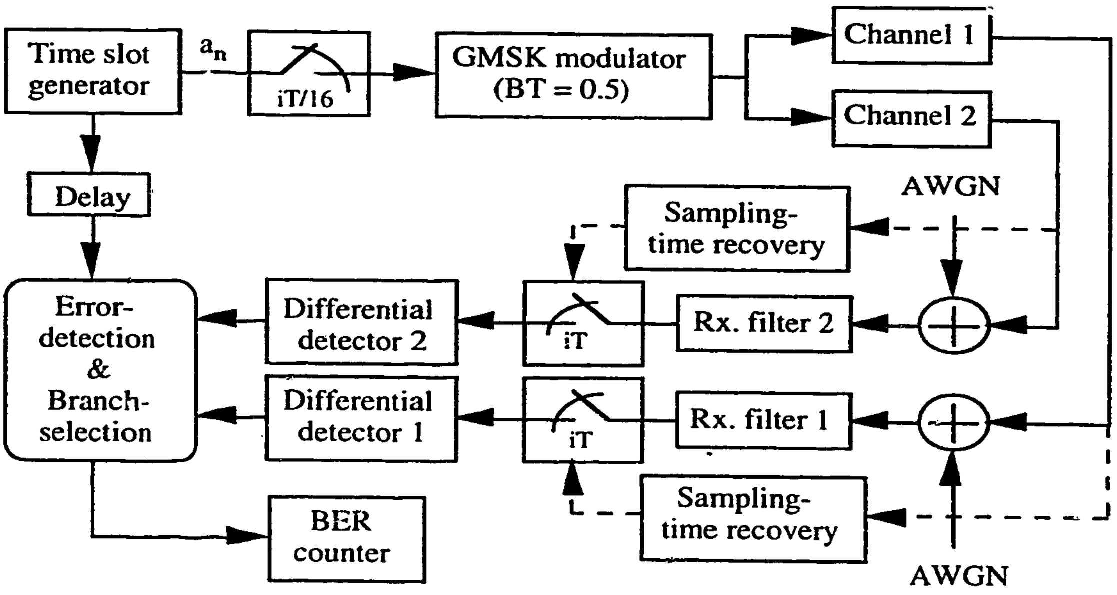

Fig. 5.9. A simulation model of the system employing ERRI-driven diversity selection (reverse link).

5.2.3. Simulation results

The performance improvement of the DECT baseline system as a result of using the two diversity selection techniques is illustrated in the following simulation results.

Figure 5.10 shows the performance gains of the RSSI and ERRI-driven diversity selection schemes, respectively, in the two-path Rayleigh fading channel at a rms delay spread of 0.1250T. The target BER of 10^{-3} at $\sigma_\tau = 0.1250T$, which is not achievable without diversity, can be reached easily with either of the two diversity schemes. As expected, ERRI-driven diversity e.hibits a superior performance over RSSI-driven diversity. This is because the received signal strength does not necessarily reflect the overall link quality; a stronger received signal measured at any instant may be due to the presence of the two strong discrete paths in the channel impulse response, which in turn increases the instantaneous rms delay spread of the link at that instant. The performance of these two diversity techniques was further evaluated using different channel models.

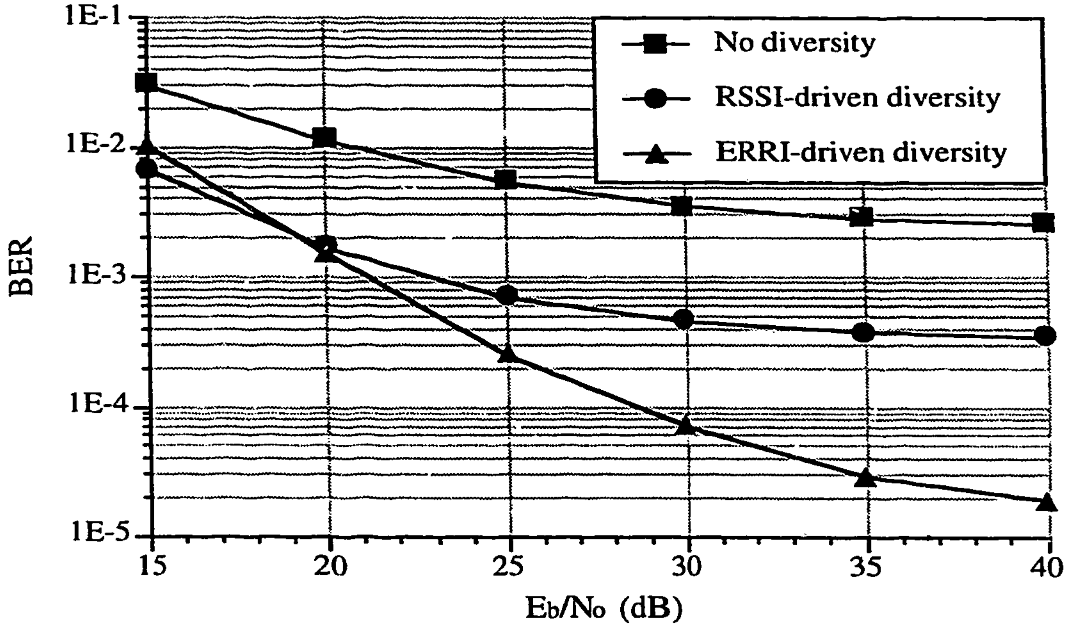

Fig. 5.10. Performance of RSSI-driven and ERRI-driven diversity selection in a two-path Rayleigh fading channel at a rms delay spread of 0.1250*T*.

Figure 5.11 shows the performance of the RSSI-driven diversity scheme in the two-path and six-path Rayleigh fading channels. The results obtained with the six-path Rayleigh fading channel are more pessimistic than those obtained with the two-path channel. In the two-path Rayleigh fading channel, given that the two paths have the same average power, the largest possible instantaneous rms delay spread of the channel corresponds to its ensemble average rms delay spread. In other words, the instantaneous rms delay spread of the selected branch, which has a larger signal power, is at most as large as its ensemble average value. On the other hand, an instantaneous rms delay spread of the six-path Rayleigh fading channel can be larger than the ensemble average rms delay spread when the instantaneous power delay profile of the channel changes in such a way that the relative powers of trailing paths are larger than their respective average values determined by the given exponentially decaying power delay profile. If a stronger diversity branch is due to the presence of strong trailing paths in that branch, then the instantaneous rms delay spread of the selected branch consisting of the six-path channel may likely be larger than that consisting of the two-path channel, which consequently leads to a poor link performance. Nevertheless, the results obtained using the two channel models consistently show that the system with RSSI-driven diversity fails to achieve the required BER of 10^{-3} at a rms delay spread of 0.1875*T*.

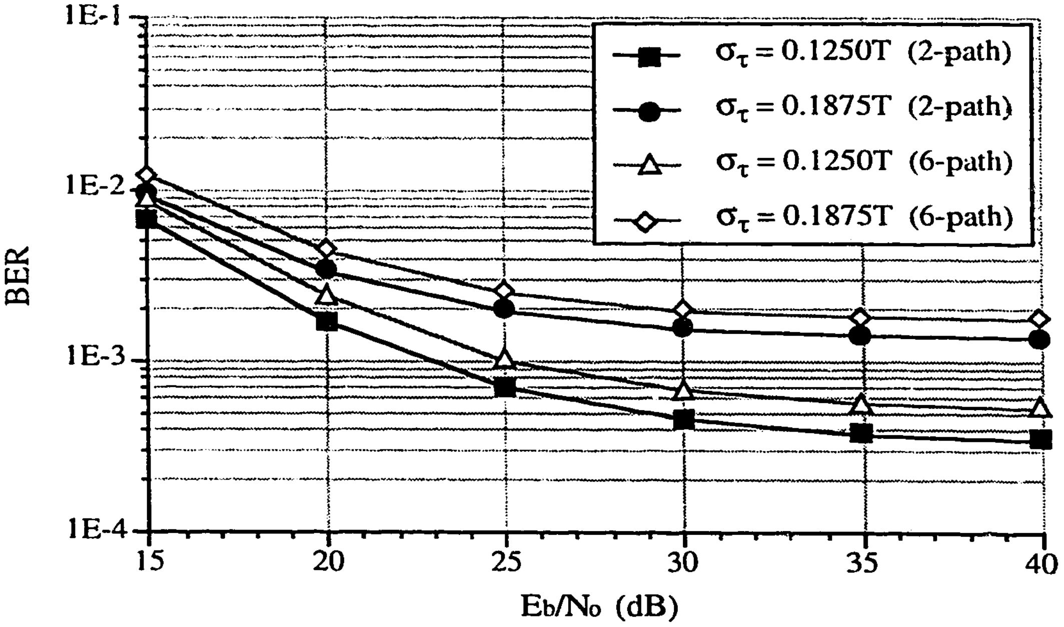

Fig. 5.11. Performance of RSSI-driven diversity selection in two-path and six-path Rayleigh fading channels.

Figure 5.12 shows that, at a small rms delay spread (0.0626*T*), the performance of the system using RSSI-driven diversity is better in the six-path Rician fading channel than that in the six-path Rayleigh fading channel, and vice-versa at a larger rms delay spread (0.1250*T*, 0.1875*T*). A less dispersive channel, just like a flat fading channel, benefits from the presence of a Rician fading path. How the presence of a Rician fading path contributes to worse channel characteristics in the case of a large rms delay spread may be explained using the following extreme scenario, where the trailing paths have large time delays relative to the first path. Suppose the first path of the six-path Rayleigh fading channel encounters a very deep fade while the powers of trailing paths increase by a certain gain factor, then the dominant path simply shifts from the first path to the second path without significantly affecting the rms delay spread of the channel. When a similar scenario occurs in the six-path Rician fading channel, the dominant path may not simply shift from the first path to the second path since the first path has a direct, unfaded component. This may lead to an increase in the instantaneous rms delay spread of the channel, and consequently deteriorate the link quality.

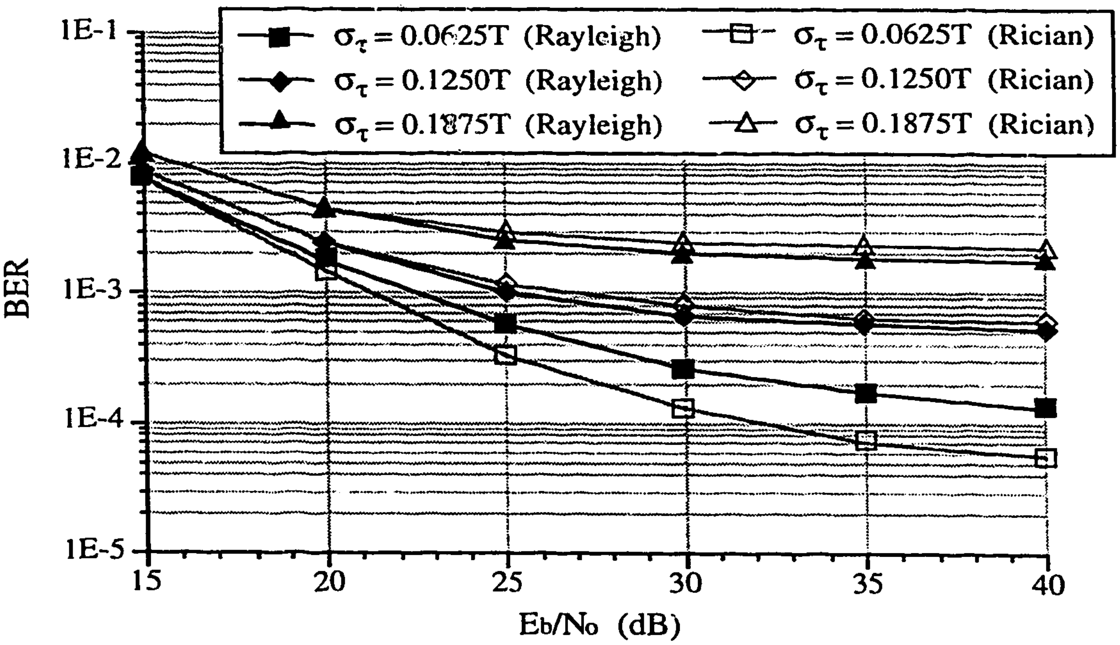

Fig. 5.12. Performance of RSSI-driven diversity selection in a six-path Rayleigh (Rician) fading channel.

The performance of the ERRI-driven diversity scheme has also been evaluated using the different channel models. Figure 5.13 shows that the performance simulated with the two-path Rayleigh fading channel is optimistic as compared to the one obtained with the six-path Rayleigh fading channel. A possible reason to account for this discrepancy is that the diversity branches involving two independent six-path Rayleigh fading channels are more likely to yield a link with a larger instantaneous rms delay spread than those involving two independent two-path Rayleigh fading channels. However, in comparison with Figure 5.11, the results obtained with the two channel models are consistent in showing that ERRI-driven diversity selection is superior to RSSI-driven diversity. This is because ERRI-driven diversity selection helps to counteract both multipath fading and ISI by selecting a branch with a better link quality, in terms of signal power and rms delay spread.

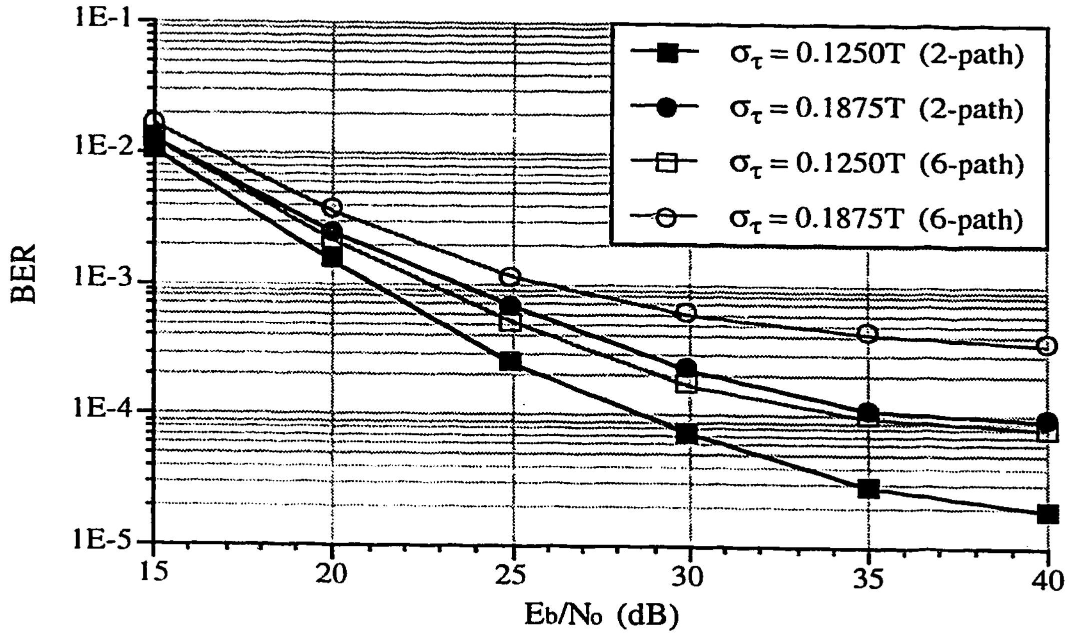

Fig. 5.13. Performance of ERRI-driven diversity selection in six-path and two-path Rayleigh fading channels.

Next, the maximum rms delay spread tolerated by the system employing ERRI-driven diversity was investigated. Figure 5.14 shows that, in the two-path Rayleigh fading channel, the system manages to attain the required BER of 10^{-3} at rms delay spreads up to $0.33T$ (about 286 ns for $T = 868$ ns).

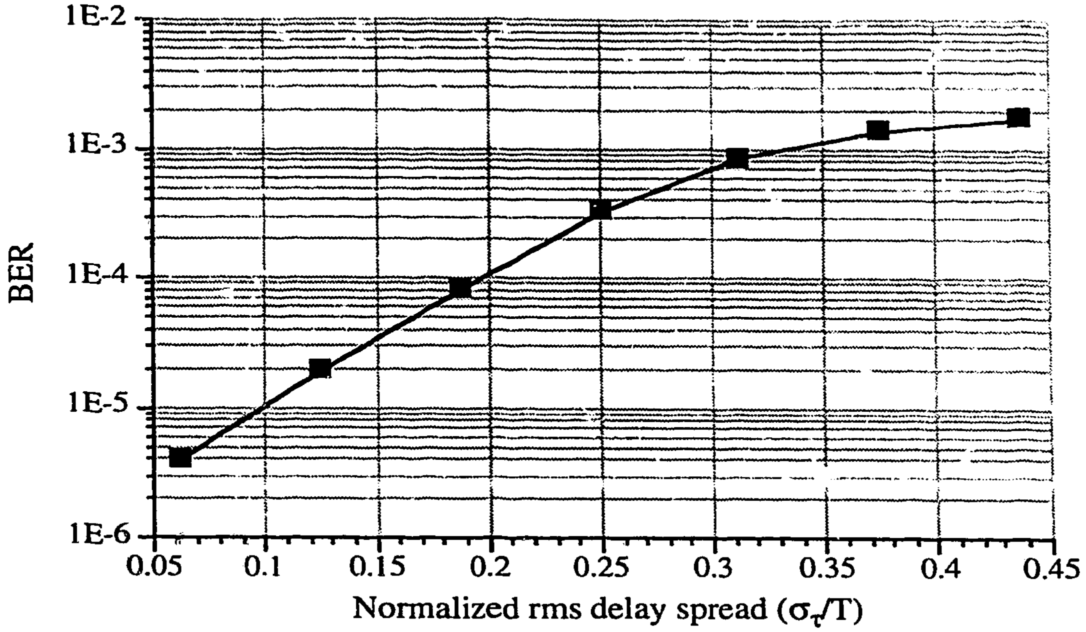

Fig. 5.14. BER vs. normalized channel rms delay spread of the system using ERRI-driven diversity in a two-path Rayleigh fading channel for E_b/N_o = 40 dB.

However, the simulation results presented in Figure 5.14 contradict those obtained using the six-path Rayleigh fading channel; as shown in Figure 5.15, the system with ERRI-driven diversity fails to acquire the required BER in either the six-path Rayleigh or the six-path Rician fading channel even at a rms delay spread of 0.2500*T*. Thc differences occurring in the comparison imply that the simple two-path channel model, which is quite commonly used in the literature, may not be suitable for evaluating the performance of the indoor system using diversity selection. Figure 5.15 also shows that the presence of a direct path (Rician fading) in the fading channel with a large rms delay spread may further degrade the communication link. This is possibly because, as discussed previously while evaluating the performance of RSSI-driven diversity, the presence of a direct unfaded component in the first path of a dispersive fading channel may yield a large instantaneous rms delay spread.

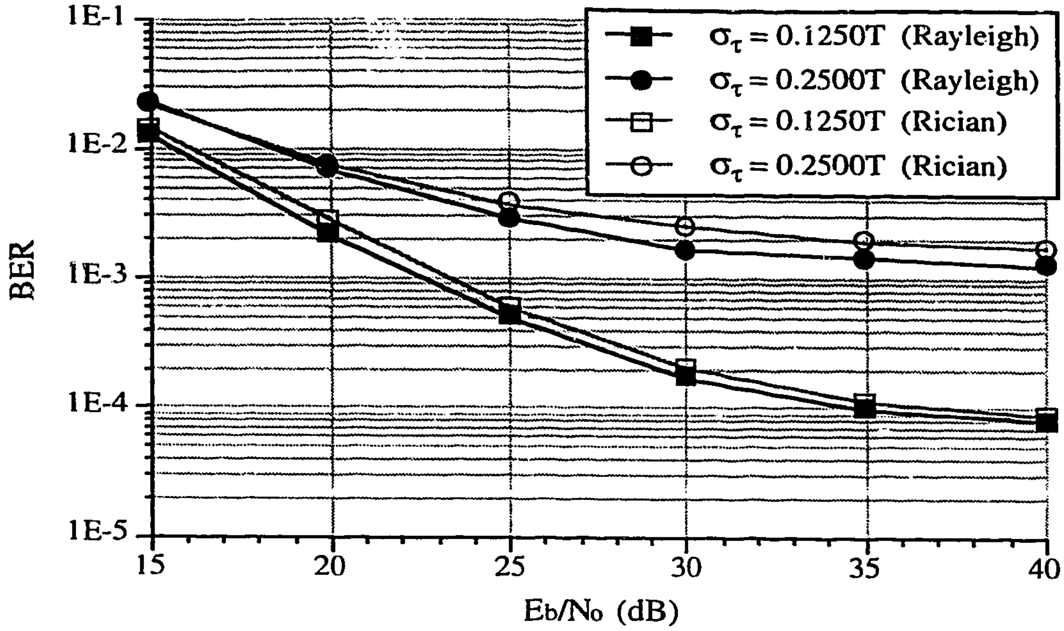

Fig. 5.15. Performance of ERRI-driven diversity selection in a six-path Rayleigh (Rician) fading channel.

Lastly, the performance of the RSSI and ERRI-driven diversity selection schemes was evaluated using the experimentally derived seven-path Rayleigh fading channel with a rms delay spread of 0.1663*T*. The effect of envelope cross-correlation between the discrete paths of the channel on the performance of the diversity schemes was investigated. In this study, the diversity branches were again assumed to be uncorrelated (independent). The simulation results in Figure 5.16 show that both diversity schemes perform significantly better in the channel with correlated discrete paths (see (5.1) for envelope cross-correlation coefficients between correlated fading paths). In comparison with Figures 5.12 and 5.15, it is observed that the performance of these two schemes in the uncorrelated seven-path Rayleigh fading channel is still better than that obtained with the six-path Rayleigh fading channel at a lower rms delay spread of 0.1250*T*. Apparently, the six-path fading channels with an exponentially decaying average power delay profile are the worst channels among the different frequency-selective channel models considered for the baseline system in this thesis.

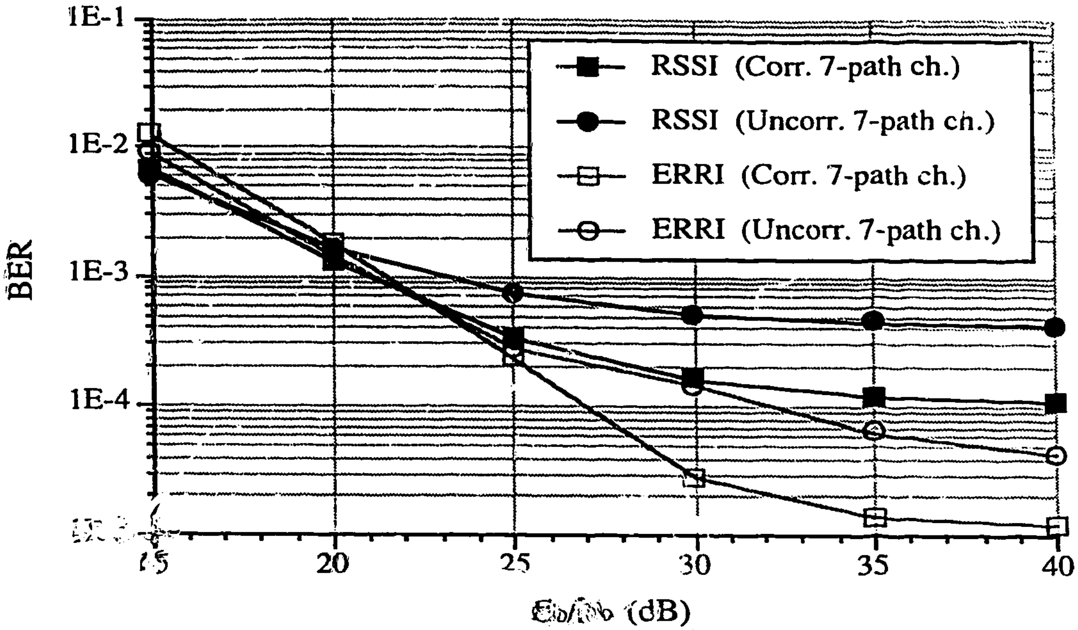

Fig. 5.16. Performance of RSSI-driven and ERRI-driven diversity selection in a seven-path (correlated or uncorrelated) Rayleigh fading channel with independent diversity branches.

The foregoing investigations have shown that channels with different power delay profiles may have noticeable effects on the performance of the system using diversity selection. The diversity branches containing simple two-path channels tend to produce better link performance, whereas those comprising six-path fading channels with uncorrelated discrete paths yield the worst link performance. The investigation also shows that, for the system with or without diversity selection, the channel model consisting of correlated fading paths represents a better transmission link than that consisting of uncorrelated fading paths with the same average power delay profile.

Simulation results obtained with the two-path Rayleigh fading channels and six-path (Rayleigh or Rician) fading channels consistently show that channel dispersion can severely degrade the system performance. According to the more conservative results obtained with the six-path (Rayleigh or Rician) fading channel, even the best diversity selection scheme, driven by an ERRI criterion, fails to acquire the required BER of 10^{-3} as the rms delay spread of the channel increases to 0.2500T (217 ns for T = 868ns). This implies that, in addition to diversity selection, channel equalization is required to further enhance the system performance.

5.3. Simulation and Performance Evaluation of the Baseline System Employing Decision Feedback Equalization

It was decided to use a passband decision feedback equalizer (DFE), as depicted in Figure 3.5, to mitigate ISI occurring in the DECT link.

5.3.1. System with an adaptive decision feedback equalizer

Figure 5.17 shows the model of the baseline system employing a passband DFE.

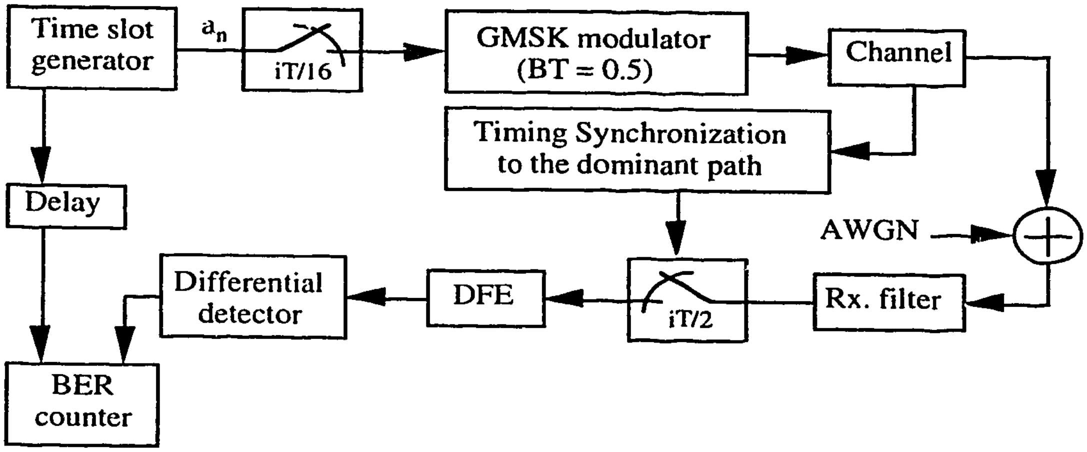

Fig. 5.17. A simulation model of the baseline system with a passband DFE.

The functional blocks of the transmitter are the same as those previously described. Some changes are introduced into the receiver. Instead of recovering the sampling time from the channel output, the sampler at the receiver is now synchronized to the dominant component (path) of the average channel impulse response. In this case, contributions from other discrete paths of the channel may be considered as interferers to be dealt with by the DFE.

The passband DFE consists of a $T/2$-spaced feedforward transversal filter (FFF) and a T-spaced feedback transversal filter (FBF). Two samples of the received signal, taken at $T/2$-intervals apart, are fed to the FFF and an output is generated every T-interval.

Ideally, the DFE removes all the ISI introduced by both the channel and the Gaussian pulse shaping filter at the transmitter, and the resulting equalizer output corresponds to the ISI-free MSK signal. To accomplish such an idealistic task, the DFE may be required to have an infinitely long FFF. However, such a structure cannot be implemented, and thus only finite length FFF and FBF are considered in practice. It was then decided to determine the length of the filters based on the amount of dispersion introduced by the communication link. The constant ISI from GMSK modulation (with $BT = 0.5$) has a duration of less than two bit intervals ($\pm T$) around the main pulse. Since it was also assumed that the excess delay of the channel impulse response was only up to $2T$, a DFE with six $T/2$-spaced taps in its FFF and two T-spaced taps in its FBF might be used to achieve acceptable performance. We call this a (6,2)DFE. The last tap of the FFF is also the reference tap of the DFE. To train the DFE, a known training sequence is transmitted at the beginning of every time-slot; however, no sequence specifically dedicated to equalizer training is included in the standard DECT slot structure, since originally equalization was not considered necessary for the DECT system to operate within its performance specifications. Nevertheless, the 16-bit packet synchronization word in the S-field of the DECT time-slot may be used for training the equalizer. It is defined by [5] as {1, 1, 1, -1, 1, -1, -1, 1, 1, -1, -1, -1, 1, -1, 1, -1} for the forward link transmission, and it is inverted for the reverse link transmission. For the training of the passband DFE, samples of the MSK signal corresponding to the states of the training bits are used. Assuming that the signal phase associated with the first training bit is $\pi/2$, then the phase sequence of the MSK signal corresponding to the above training sequence is $\{\pi/2, \pi, 3\pi/2, \pi, 3\pi/2, \pi, \pi/2, \pi, 3\pi/2, \pi, \pi/2, 0, \pi/2, 0, \pi/2, 0\}$; i.e., the signal phase changes by $\pm\pi/2$ according to the polarity of the binary symbol. Consequently, samples of the in-phase and quadrature components of the MSK signal (cosine and sine of the signal phase, respectively) will either be -1, 0, or +1, depending on the phase of the signal. After the training, the equalizer is switched to the decision feedback mode and its coefficients continue to be adaptively adjusted. The decision device of the DFE has two threshold levels set at ± 0.5. The decision on the equalized signal is made according to the following procedure at every T-interval:

If $a > 0.5$, then $b = 1$,
if $a < -0.5$, then $b = -1$,
else $b = 0$

where a is an in-phase or quadrature component of the equalized signal at the input of the decision device and b is the corresponding component of the MSK signal at its output.

The standard RLS algorithm is used to jointly update the tap-weights of the FFF and FBF once every T-interval. The DFE output is fed to a one-bit differential detector.

5.3.2. Simulation results

Figure 5.18 shows the convergence rate of the (6,2) DFE. The square of the error signal from the DFE was ensemble averaged over 50 trials, and a random bit sequence was used to train the equalizer in each trial. The six-path channel, with the channel characteristics being kept constant at a rms delay spread of $0.3125T$, was used in the simulation.

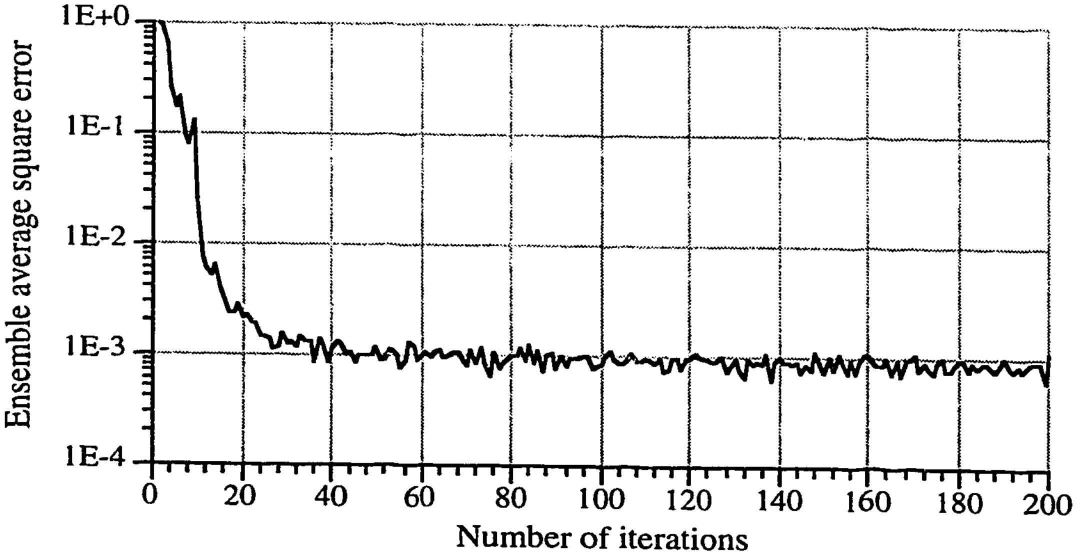

Fig. 5.18. Learning curves of the RLS algorithm for a (6,2)DFE in a six-path static channel at a delay spread of $0.3125T$ and E_b/N_o of 40 dB.

The simulation result shows that the (6,2)DFE does not fully converge after sixteen bits of training. Therefore, using the 16-bit training sequence previously specified may result in a performance penalty.

Figure 5.19 compares the performance of the baseline system, with or without diversity selection, with that of the system using the (6,2)DFE. The six-path Rayleigh fading channel with a rms delay spread of 0.1250T was used in the simulation. The 16-bit packet synchronization word in the DECT slot was used to train the DFE. The results show that, for the range of E_b/N_o considered, the DFE effectively removes the error floor resulting from the ISI. It has also been demonstrated that, at a rms delay spread of 0.1250T (109 ns for T = 868 ns), the DFE has no advantage over ERRI-driven diversity for E_b/N_o less than 35 dB.

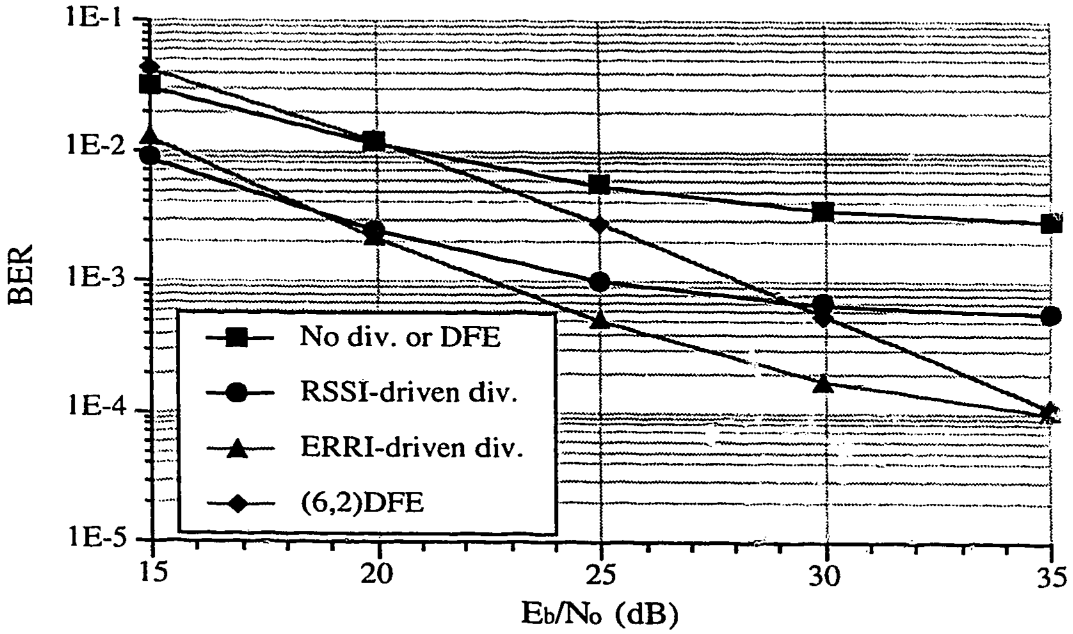

Fig. 5.19. Performance of the baseline system (with or without diversity) and a (6,2)DFE in a six-path Rayleigh fading channel at a rms delay spread of 0.1250T (16-bit training sequence).

The influences of different average power delay profiles of a channel upon the performance of the DFE were also investigated. Figure 5.20 shows that, at a rms delay spread of 0.1250T, the DFE performs equally well in both the two-path and six-path Rayleigh fading channels. However, at a rms delay spread of 0.3750T, the result obtained with the two-path fading channel is more optimistic than that obtained using the six-path channel; about 2 dB more E_b/N_o is required to acquire a BER of 10^{-3} in the six-path channel. Apparently, the presence of more interfering (trailing) paths may give rise to a more adverse channel.

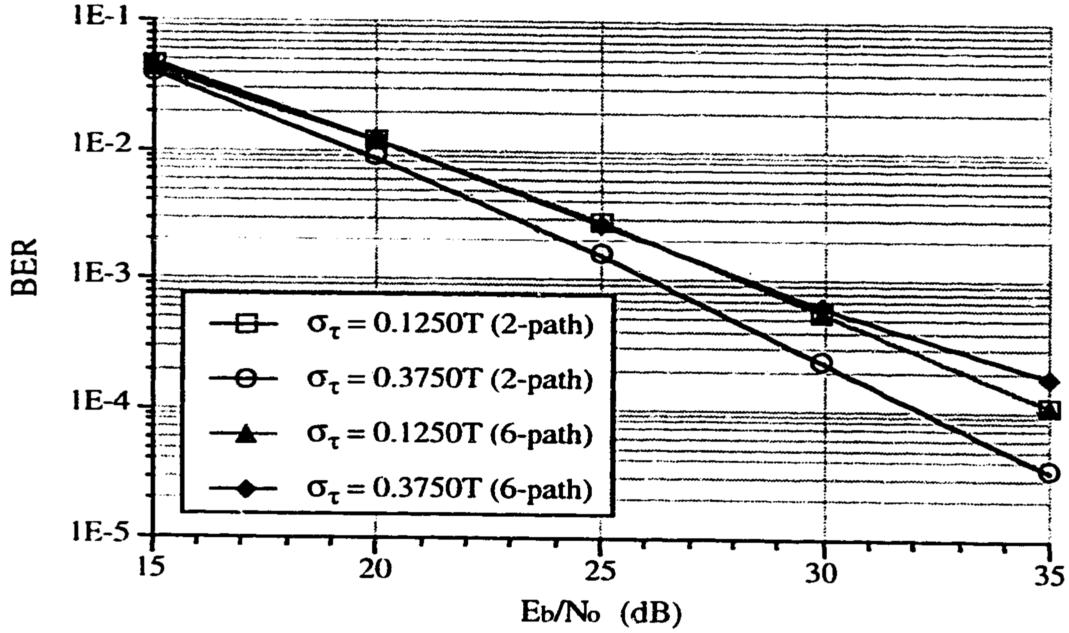

Fig. 5.20. Performance of a (6,2)DFE in two-path and six-path Rayleigh fading channels (16-bit training sequence).

As indicated in Figure 5.18, the 16-bit training sequence is insufficient to allow for full convergence of the RLS algorithm used in the (6,2)DFE. To further exploit the DFE, a longer training sequence was used. Figure 5.21 shows the performance gain of the DFE in the six-path Rayleigh fading channel when the number of training bit is increased from sixteen to thirty-two; about 2.5 dB gain is achieved at a BER of 10^{-3} for a rms delay spread of 0.3750*T*. The thirty-two bit training sequence was formed by appending sixteen random bits to the original 16-bit training sequence. It is illustrated in Figure 5.22 that, for E_b/N_o less than 30 dB, only a marginal gain is achieved by further increasing the number of training bits to forty-eight. The inconsistent results obtained at E_b/N_o of 35 dB, as shown in Figure 5.22, are likely due to larger variances of the estimated BERs at this point since relatively fewer bit errors were counted (350 bit errors were counted at E_b/N_o = 35 dB as compared to more than 2000 bit errors counted at E_b/N_o = 30 dB)

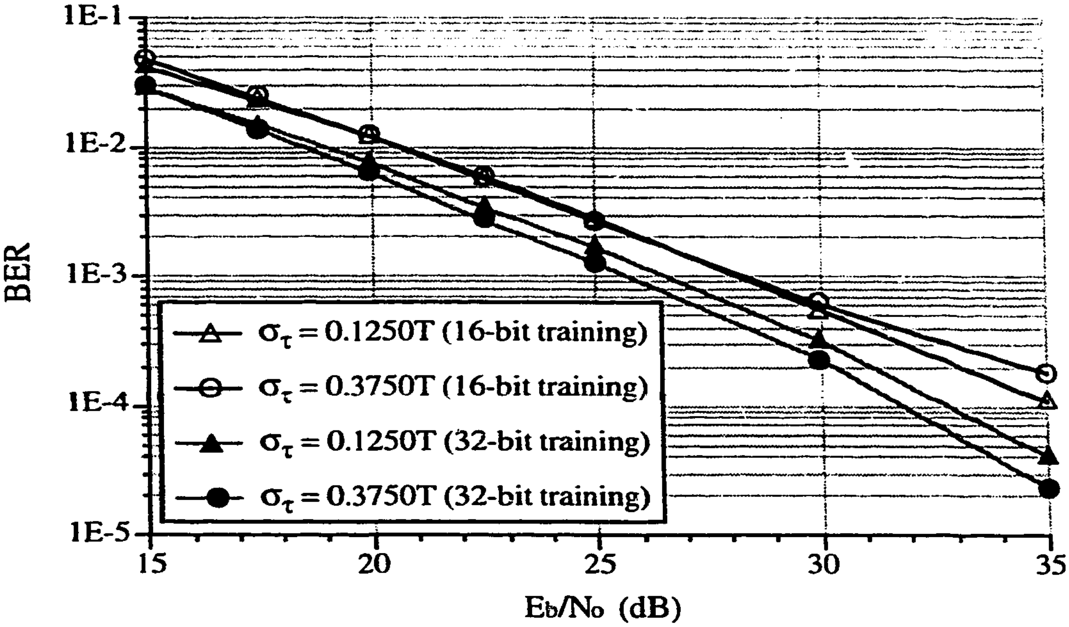

Fig. 5.21. Performance of a (6,2)DFE, with 16-bit and 32-bit training sequences, in a six-path Rayleigh fading channel.

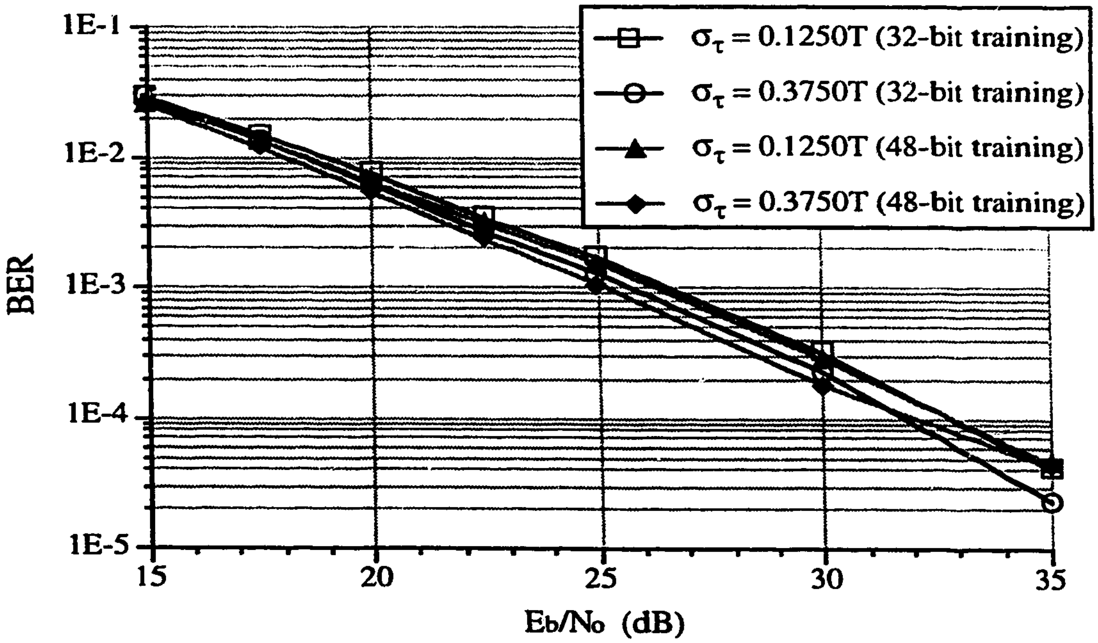

Fig. 5.22. Performance of a (6,2)DFE, with 32-bit and 48-bit training sequences, in a six-path Rayleigh fading channel.

Next, the effect on the system performance due to an increase in the number of taps in the FFF and FBF of the DFE was investigated using the six-path Rayleigh fading channel. The performance of the (8,2)DFE (8-tap *T*/2-spaced FFF and 2-tap *T*-spaced FBF) and (8,3)DFE (8-tap *T*/2-spaced FFF and 3-tap *T*-spaced FBF) is compared with that of the (6,2)DFE in Figure 5.23. A forty-eight bits training sequence was used for equalizer training. The simulation results indicate that there is no apparent advantage of using the DFEs with longer filters in the channel with a rms delay spread of 0.1250*T*.

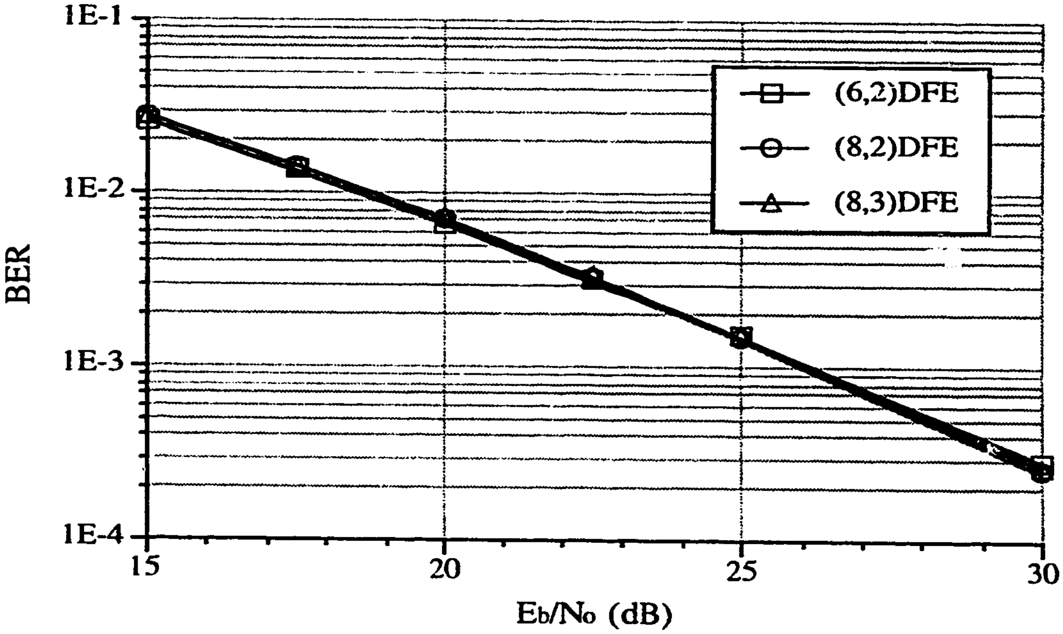

Fig. 5.23. Performance of a (6,2)DFE, (8,2)DFE and (8,3)DFE in a six-path Rayleigh fading channel at a rms delay spread of 0.1250*T* (48-bit training sequence).

The performance of the three DFEs was re-evaluated at a larger rms delay spread of 0.3750*T* using the same six-path Rayleigh fading channel structure. Figure 5.24 shows that an additional gain of about 1 dB is achieved at a BER of 10^{-3} by the (8,2)DFE. Increasing the number of taps in the FBF does not further improve the performance of the DFE significantly. However, it should be emphasized that all three DFEs were trained by the same forty-eight bit training sequence, and the (8,3)DFE has the slowest rate of convergence among the three. So, there might be a possibility that the (8,3)DFE did not fully converge before equalization began.

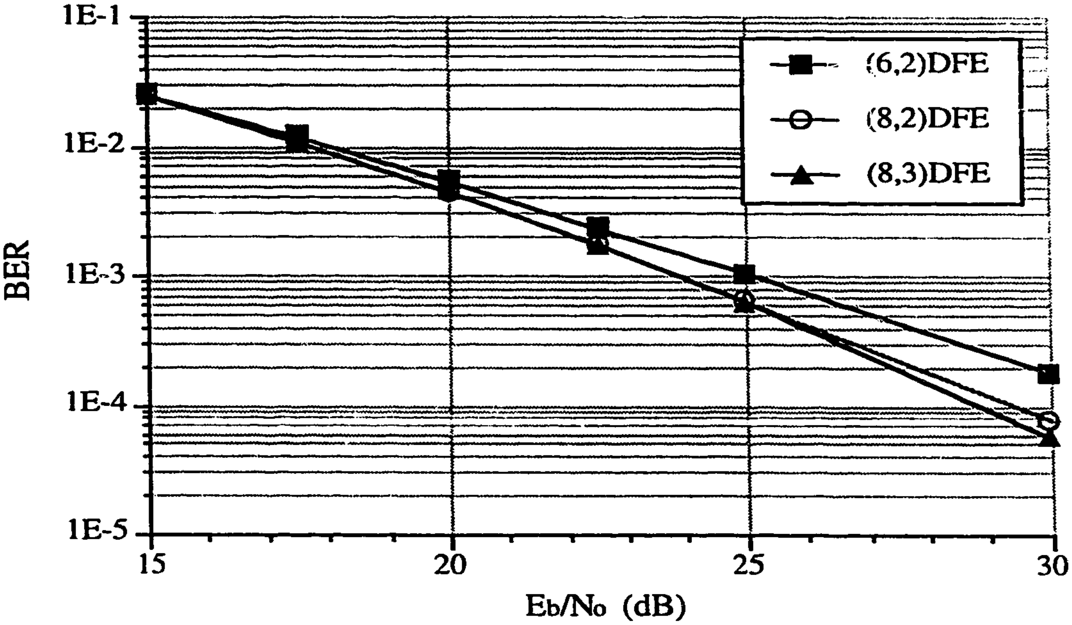

Fig. 5.24. Performance of a (6,2)DFE, (8,2)DFE and (8,3)DFE in a six-path Rayleigh fading channel at a rms delay spread of 0.3750T (48-bit training sequence).

From the previous simulation results, it is observed that the system with DFE performs better in a frequency selective fading channel than the system without DFE in a flat fading channel (compare Figures 5.4 and 5.24). This implies that the DFE is also capable of exploiting the time-diversity inherent in the frequency selective channel. This characteristic becomes more obvious in the following study.

The effect of envelope cross-correlation between discrete paths of the channel on the performance of the (6,2)DFE was investigated using the experimentally derived seven-path Rayleigh fading channel. A 32-bit training sequence was used for equalizer training. The simulation results, as given in Figure 5.25, show that the DFE performs better in the channel with independently fading paths. The DFE requires an additional 1.5 dB E_b/N_o to acquire a BER of 10^{-3} in the channel with correlated fading paths. This is because the DFE is able to make use of the time-diversity inherent in the channel, and the diversity gain is maximum when the discrete paths are uncorrelated.

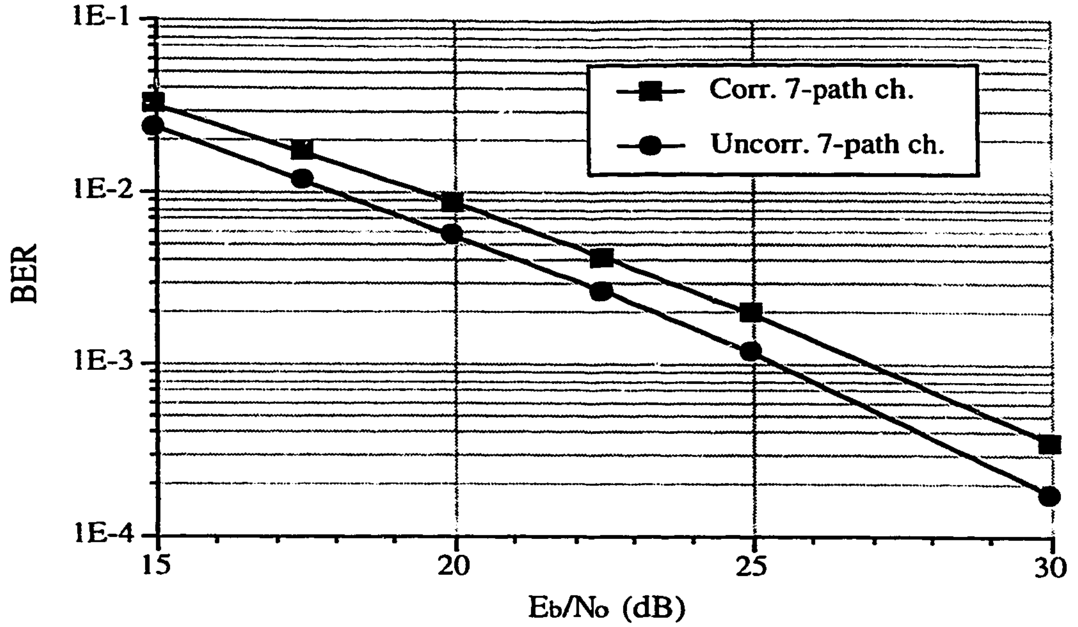

Fig. 5.25. Performance of a (6,2)DFE in a seven-path Rayleigh fading channel (32-bit training sequence).

The simulation results presented in this section have shown that the DFE is effective in counteracting channel dispersion. Provided with sufficient training and an adequate number of taps, the DFE performs better at a larger rms delay spread. Besides mitigating ISI, it is also capable of exploiting the time diversity inherent in the multipath channel.

5.4. Simulation and Performance Evaluation of Combined Diversity-Equalization Schemes

The performance of the following four schemes, which have been detailed in Chapter 4, was simulated for the reverse link of the DECT system.

- *Scheme I*: DFE with RSSI-driven diversity selection
- *Scheme II*: DFE with ERRI-driven diversity selection
- *Scheme III*: DFE with MSE-driven diversity selection
- *Scheme IV*: Integration of diversity combining and DFE through joint-parameter optimization

The structure and function of the DFE for *Schemes I, II,* and *III* have been described and evaluated in Section 5.3. However, the same DFE is not used in *Scheme IV.*

5.4.1. System with combined diversity-equalization schemes

The simulation model of the system employing *Scheme I* is similar to that depicted in Figure 5.8 for the system with RSSI-driven diversity, with the exception that a DFE is introduced at the receiver. In this case, the signal received from the selected diversity branch is sampled by a 2/*T*-rate sampler instead of a 1/*T*-rate sampler, and samples of the received signal are input to a DFE before passing to a one-bit differential detector. As described in Section 5.3.1, the 2/*T*-rate sampler is synchronized to the dominant path of the average channel impulse response.

For *Scheme II*, changes similar to those described above are introduced to the simulation model shown in Figure 5.9 for the system with ERRI-driven diversity. Each of the receiver filters is now followed by a 2/*T*-rate sampler synchronizing to the dominant path of the average channel impulse response, and sampled signals from the two diversity branches are input to two separate DFEs. The equalized signals from the two branches are then detected and error count is effected with the same procedure presented in Section 5.2.2.

The simulation model of the system employing *Scheme III* closely resembles that described for *Scheme II*. However, instead of selecting the appropriate branch based on an ERRI criterion, *Scheme III* utilizes mean square error signals from the DFEs in the two diversity branches for branch selection. In the simulation, the mean square error signals from the DFEs are determined for each time-slot received, and the number of accumulated errors in the BER counter is increased by the number of bit errors occurring in the time-slot received from the branch with a smaller mean square error signal.

Scheme IV integrates diversity combining and decision feedback equalization into a single entity. It has two *T*/2-spaced FFFs, one for each diversity branch, and one *T*-spaced FBF. The simulation model of the system using this scheme is again similar to that using *Scheme II* except that the two DFEs in *Scheme II* are replaced by a single composite diversity-equalization unit depicted in Figure 4.4. That is, instead of feeding to two separate DFEs, samples of the received signals from the diversity branches are fed to the two FFFs of the composite unit. The coefficients of the three filters are jointly optimized by the standard RLS algorithm. Equalizer training and the process of making decisions on the equalized signal follow the exact procedures described for the DFE in Section 5.3.1. The equalized signal from the composite unit is fed to a one-bit differential detector. The detected data are then compared with the known transmitted data to determine the BER of the system.

In all the simulations, diversity branches were assumed to have uncorrelated fading.

5.4.2. Simulation results

The performance gains achieved by the four combined diversity-equalization schemes are demonstrated in the following simulation results.

Figure 5.26 compares the performance of *Scheme I* with that of ERRI-driven diversity selection and a (6,2)DFE in the six-path Rayleigh fading channel with a rms delay spread of 0.1250*T*. The 16-bit packet synchronization word was used for equalizer training. The results show that, by simply incorporating a (6,2)DFE with RSSI-driven diversity selection, *Scheme I* is able to achieve an additional gain of about 5 dB at a BER of 10^{-4} over the (6,2)DFE and ERRI-driven diversity selection.

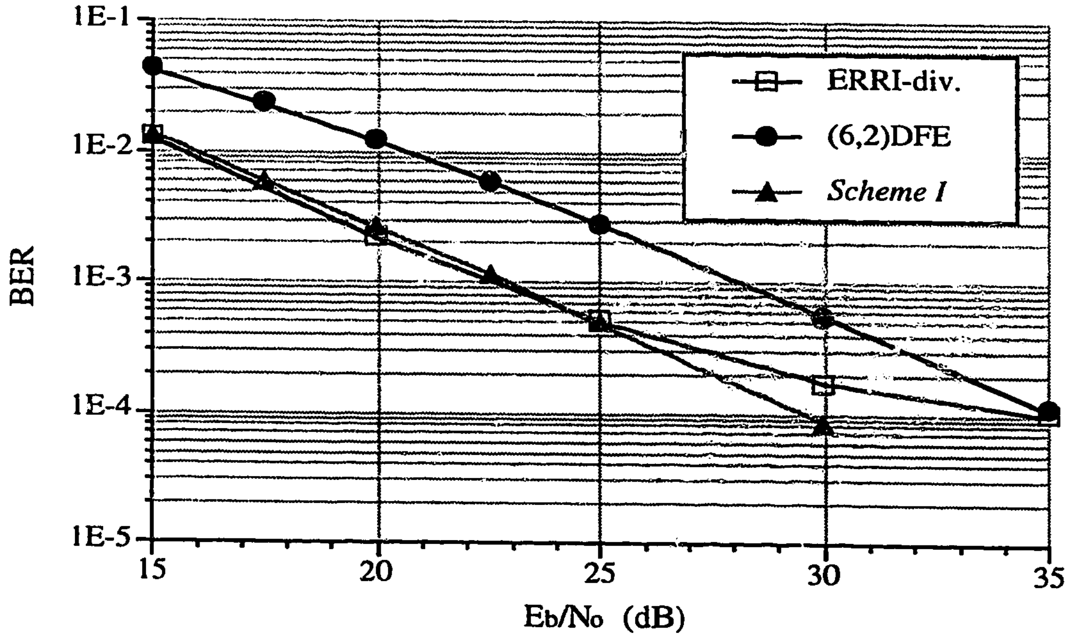

Fig. 5.26. Performance of ERRI-driven diversity selection, (6,2)DFE and *Scheme I* in a six-path Rayleigh fading channel at a rms delay spread of 0.1250*T* (16-bit training sequence).

Figure 5.27 shows the performance of the four schemes in the same six-path Rayleigh fading channel with a rms delay spread of 0.1250*T*. *Schemes I, II* and *III* used (6,2)DFEs, whereas *Scheme IV* used a 2-tap FBF and 4-tap FFFs. Shorter FFFs were used in *Scheme IV* because the 16-bit training sequence used for equalizer training in this case would be far too short for *Scheme IV* if it also used 6-tap FFFs. The results show that, for E_b/N_o greater than 25 dB, *Schemes II* and *III* perform equally well with a gain of about 4 dB over *Scheme I* at a BER of 10^{-4}. An additional gain of about 1 dB can be achieved by *Scheme IV.*

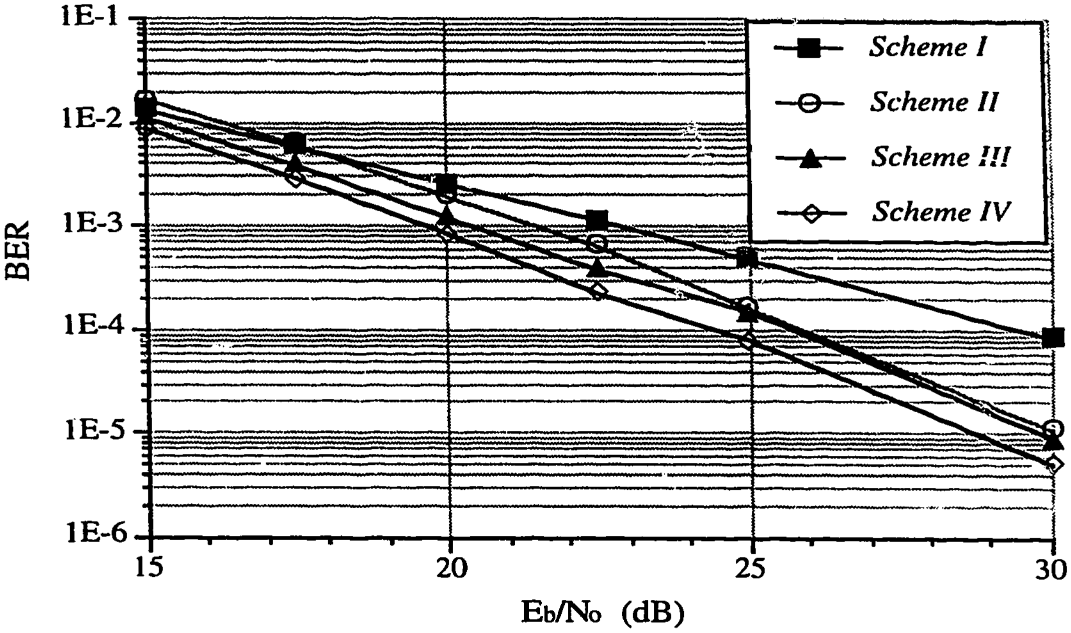

Fig. 5.27. Performance of *Schemes I, II, III* and *IV*, in a six-path Rayleigh fading channel at a rms delay spread of 0.1250*T* (16-bit training sequence).

The performance of the four schemes, with the length of their corresponding filters (FFFs and FBF) remaining unchanged, was also evaluated against different rms delay spreads of the six-path Rayleigh fading channel. The 16-bit training sequence was used and an average E_b/N_o was kept at 25 dB in all cases. The simulation results are shown in Figure 5.28. The performance of *Schemes I, II* and *III* is fairly consistent at the different rms delay spreads considered. *Scheme IV* gives the best performance but its performance gain over *Schemes II* and *III* is not significant, particularly at the channel rms delay spread of 0.3125T. However, it will be shown later that, provided with sufficient equalizer training, the performance of these four schemes can be further improved, with *Scheme IV* being far sunerior to the other three schemes.

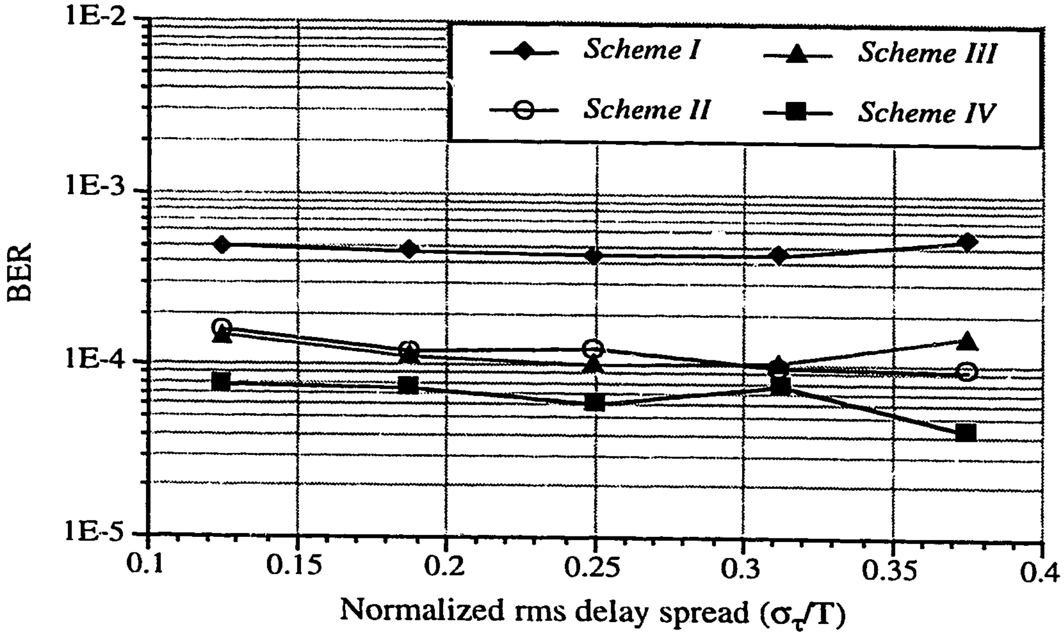

Fig. 5.28. Performance of *Schemes I, II, III* and *IV* in a six-path Rayleigh fading channel with different rms delay spreads at E_b/N_o of 25 dB.

The influences of different average power delay profiles of a channel on the performance of these schemes were also investigated using the two-path and six-path Rayleigh fading channel models. The filter structures (length of FFF and FBF) of these schemes and the training sequence remained the same as before. Figure 5.29 only shows the simulation results for *Schemes I* and *IV*. Similar effects have been observed for *Schemes II* and *III*. The results illustrate that the link performance obtained with the two-path Rayleigh fading channel is always better than that obtained with the six-path Rayleigh fading channel at the same rms delay spread of 0.1250T; both *Schemes I* and *IV* require an additional E_b/N_o of more than 1 dB to attain a BER of 10^{-4} in the six-path Rayleigh fading channel. Apparently, the simple two-path channel structure may tend to provide an over-optimistic link performance. Hence, it was not used in subsequent performance evaluations of the four schemes. Unless otherwise specified, the six-path Rayleigh fading channel was used in the following investigations.

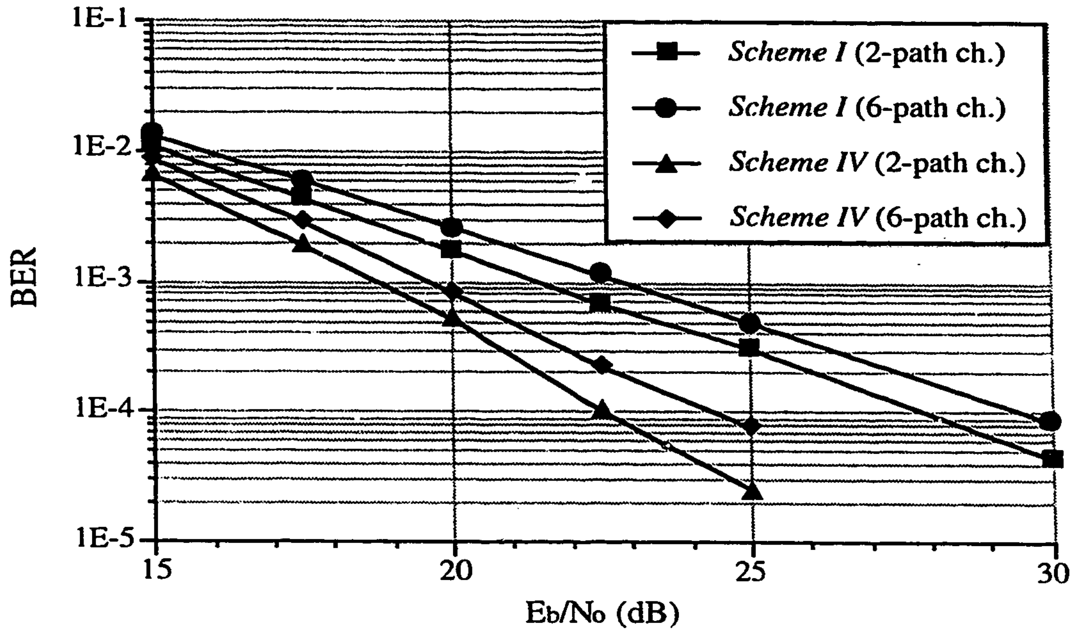

Fig. 5.29. Performance of *Schemes I* and *IV* in two-path and six-path Rayleigh fading channels with a rms delay spread of 0.1250*T* (16-bit training sequence).

Figures 5.30 and 5.31 show the performance of the four schemes in a channel with rms delay spreads of 0.1250*T* and 0.3750*T*, respectively, when a 32-bit training sequence was used. As before, *Scheme IV* had four taps in each of its FFFs and two taps in its FBF and the other three schemes used (6,2)DFEs. In comparison with Figure 5.27, Figure 5.30 shows noticeable improvements in the performance of the four schemes when the number of training bits is increased from sixteen to thirty-two, with the largest improvement of about 3 dB gain acquired for *Scheme IV* at a BER of 10^{-4}. The comparison of Figure 5.30 with Figure 5.31 reveals that the four schemes, similar to the (6,2)DFE, perform better at a larger rms delay spread of 0.3750*T*. In both cases, *Scheme II* tends to perform slightly better than *Scheme III* at a higher E_b/N_o, and vice versa at a lower E_b/N_o. This is because at a lower E_b/N_o where bursty errors are more likely to occur in both diversity branches, *Scheme II* cannot recognize the better branch when errors occur in both branches. In this case, the mean square error signals from the DFEs may give a better assessment of the link quality. As E_b/N_o increases, the chances of having bit errors appear in both branches at the same time are reduced, which then allows *Scheme II* to perform better.

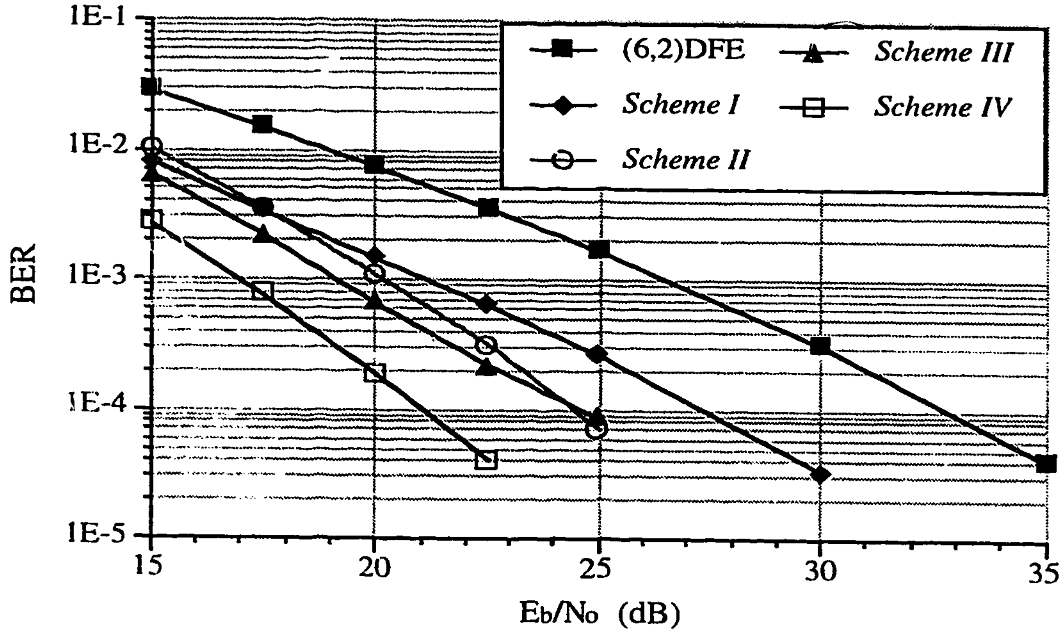

Fig. 5.30. Performance of a (6,2)DFE, *Schemes I, II, III,* and *IV* in a six-path Rayleigh fading channel at a rms delay spread of 0.1250*T* (32-bit training sequence).

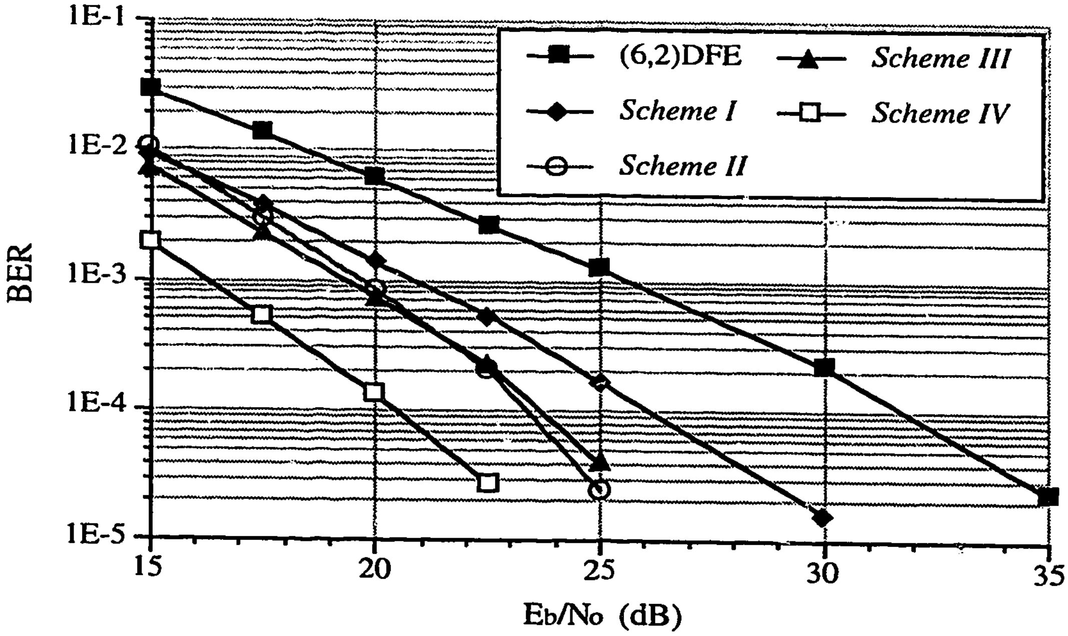

Fig. 5.31. Performance of a (6,2)DFE, *Schemes I, II, III,* and *IV* in a six-path Rayleigh fading channel at a rms delay spread of 0.3750*T* (32-bit training sequence).

Next, the effect of increasing the number of taps in each FFF of *Scheme IV* from four to six was investigated. Figure 5.32 compares the performance of *Scheme IV* using 4-tap FFFs and 6-tap FFFs. In both cases, the same 32-bit training sequence was used. It is observed that, by using six-tap FFFs, *Scheme IV* is able to achieve an additional gain of about 1.5 dB at a BER of 10^{-4} when the rms delay spread of the channel is 0.3750*T*. Also, with 6-tap FFFs, *Scheme IV* has a significant performance gain over the other three schemes (shown in Figure 5.31) at a rms delay spread of 0.3750*T*; it has about 5 dB gain over *Schemes II* and *III* and a 7 dB gain over *Scheme I*. However, at a rms delay spread of 0.1250*T*, *Scheme IV* suffers a slight performance degradation with 6-tap FFFs. This is because, for a small rms delay spread, the gain obtained from longer FFFs is relatively small, and is not enough to offset the penalty of insufficient training provided by the 32-bit training sequence.

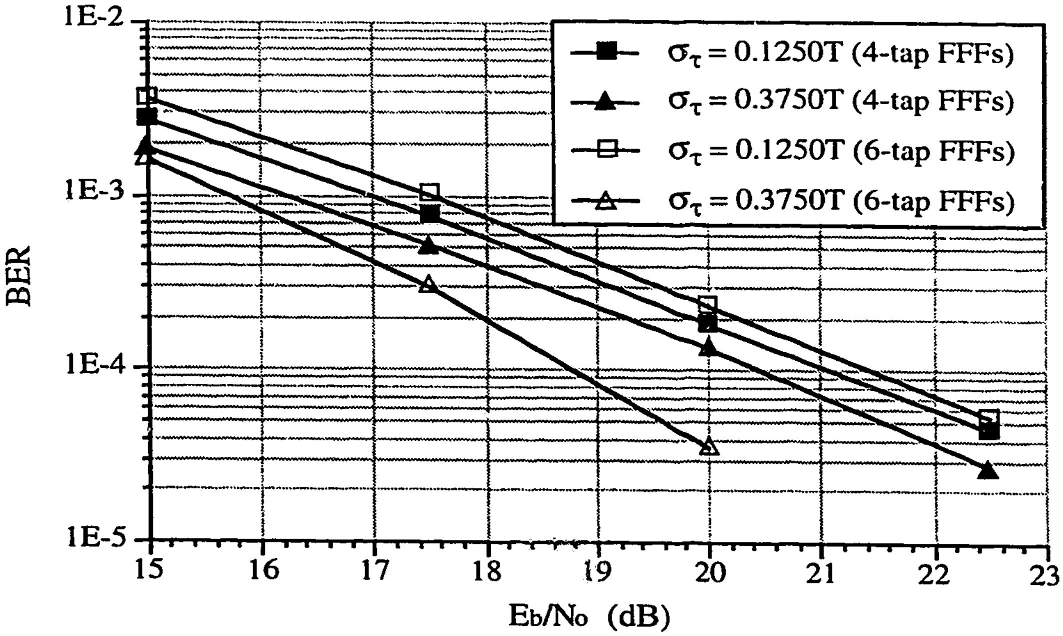

Fig. 5.32. Performance of *Scheme IV* with 4-tap and 6-tap FFFs in a six-path Rayleigh fading channel (32-bit training sequence).

To gain the full advantage of *Scheme IV* with 6-tap FFFs, a longer training sequence was used. Figure 5.33 shows the improvement in its performance when the length of the training sequence is increased from thirty-two to forty-eight bits.

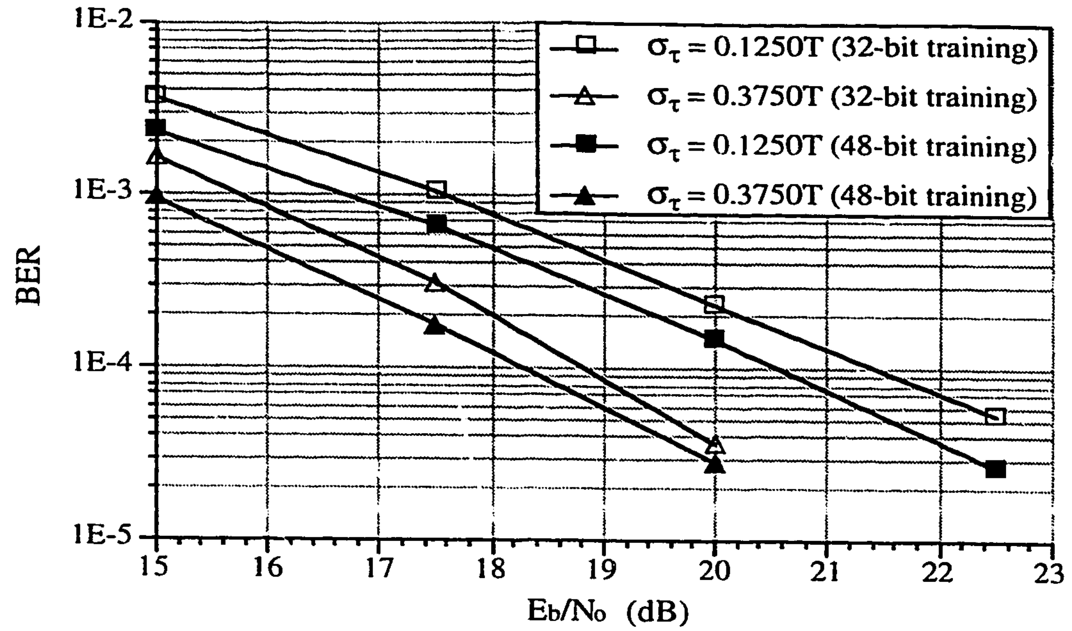

Fig. 5.33. Performance of *Scheme IV* with 6-tap FFFs, using 32-bit and 48-bit training sequences, in a six-path Rayleigh fading channel.

The improvement in the performance of *Scheme IV* with 6-tap FFFs due to an increase in the number of training bits from thirty-two to forty-eight is not as significant as the corresponding improvement obtained by increasing the number of training bits from sixteen to thirty-two (Figures 5.27 and 5.30). It may be inferred that *Scheme IV* with 6-tap FFFs is at least close to full convergence after forty-eight bits of training. For *Schemes I, II* and *III*, increasing the number of training bits from thirty-two to forty-eight may not result in noticeable improvement since, as shown in Figure 5.22, the (6,2)DFE used in these schemes almost converges fully after thirty-two bits of training.

The potential gain of using longer FFFs in all four schemes was also investigated. For *Schemes I, II* and *III*, (8,2)DFEs (eight *T*/2-spaced FFF and two *T*-spaced FBF) were used. The number of taps in each FFF of *Scheme IV* was also increased to eight while the FBF remained unchanged. In all the simulations, the 48-bit training sequence was used. Figure 5.34 illustrates the performance of *Schemes I* and *II* using (6,2)DFE and (8,2)DFE at E_b/N_o of 25 dB. The simulation results show that *Schemes I* and *II* benefit from using the DFEs with a longer FFF, particularly at a larger rms delay spread. As shown in Figure 5.35, *Scheme III* also performs better with the (8,2)DFEs. The performance of *Scheme IV* was evaluated at E_b/N_o of 22.5 dB. The inconsistent results for *Scheme IV* are likely attributable to insufficient equalizer training when longer 8-tap FFFs are used.

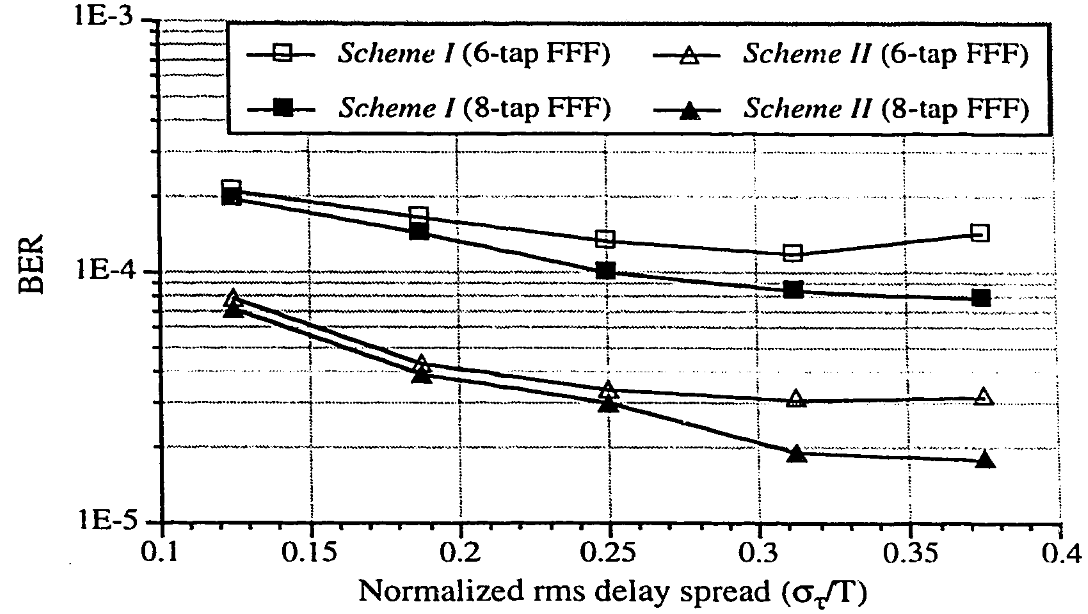

Fig. 5.34. Performance of *Schemes I* and *II* in a six-path Rayleigh fading channel at E_b/N_o of 25 dB (48-bit training sequence).

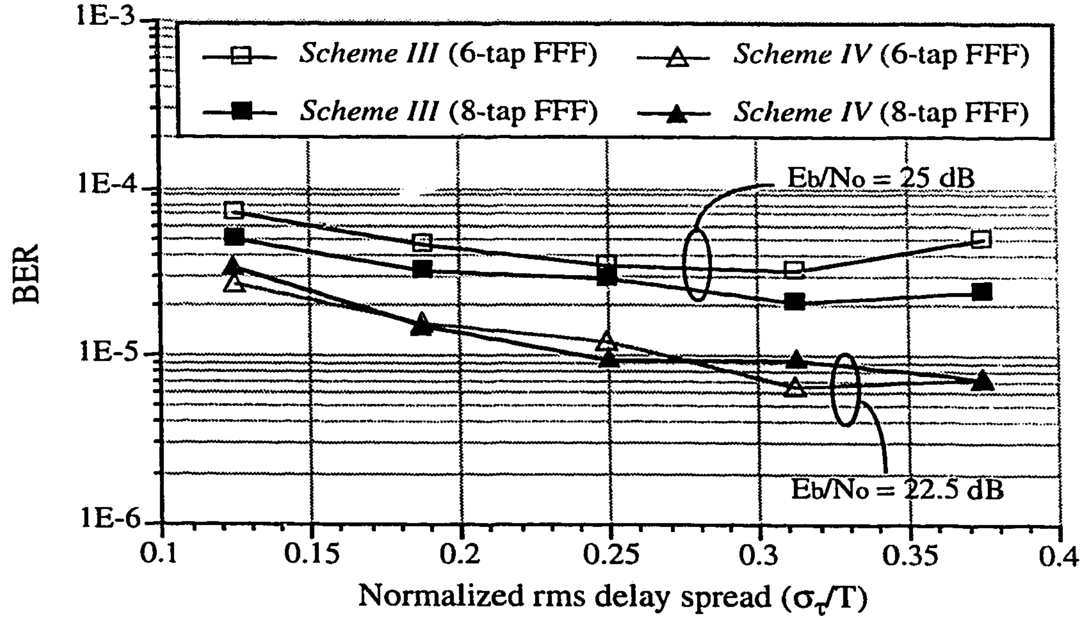

Fig. 5.35. Performance of *Schemes III* and *IV* in a six-path Rayleigh fading channel (48-bit training sequence).

Figure 5.36 compares the performance of a DFE and the four schemes, with all of them using 8-tap FFFs and a 2-tap FBF, at a rms delay spread of $0.3750T$. The 48-bit training sequence was used. At a BER of 10^{-4}, *Scheme I* has a gain of about 5 dB over the (8,2)DFE, and an additional gain of about 2 dB can be achieved by *Schemes II* and *III*. *Scheme IV* yields the best performance, with a performance gain of about 4 dB over *Schemes II* and *III* at a BER of 10^{-4}.

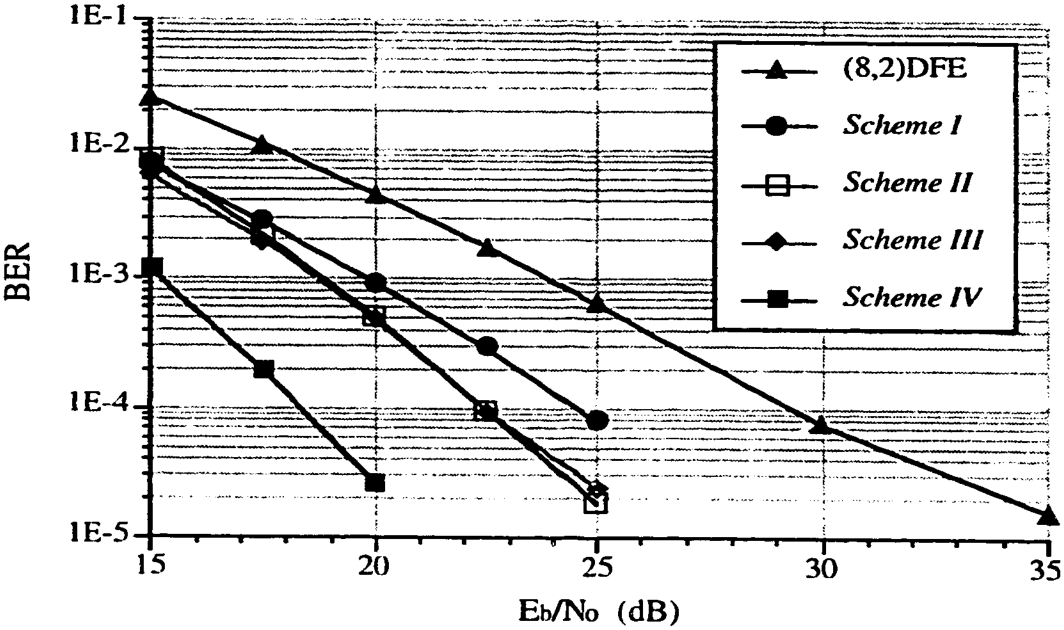

Fig. 5.36. Performance of (8,2)DFE, *Schemes I, II, III*, and *IV* with 8-tap FFFs and 2-tap FBF in a six-path Rayleigh fading channel at a rms delay spread of $0.3750T$ (48-bit training sequence).

Lastly, the effect of envelope cross-correlation between discrete paths of the channel on the performance of the four schemes was investigated by using the experimentally derived seven-path Rayleigh fading channel model, whose rms delay spread is $0.1663T$ (see subsection 5.1.1b). In this study, *Schemes I, II* and *III* used (6,2)DFEs, whereas *Scheme IV* used 4-tap FFFs and 2-tap FBF. The 32-bit training sequence was used for all four schemes. Again, the diversity branches were assumed to be independent. The simulation results, as shown in Figures 5.37, 5.38 and 5.39, indicate that the link performance of these schemes degrades for the case of a channel with correlated fading paths. The degradation varies from about 1 dB in *Scheme I* to 1.7 dB in *Scheme IV* at a BER of 10^{-4}.

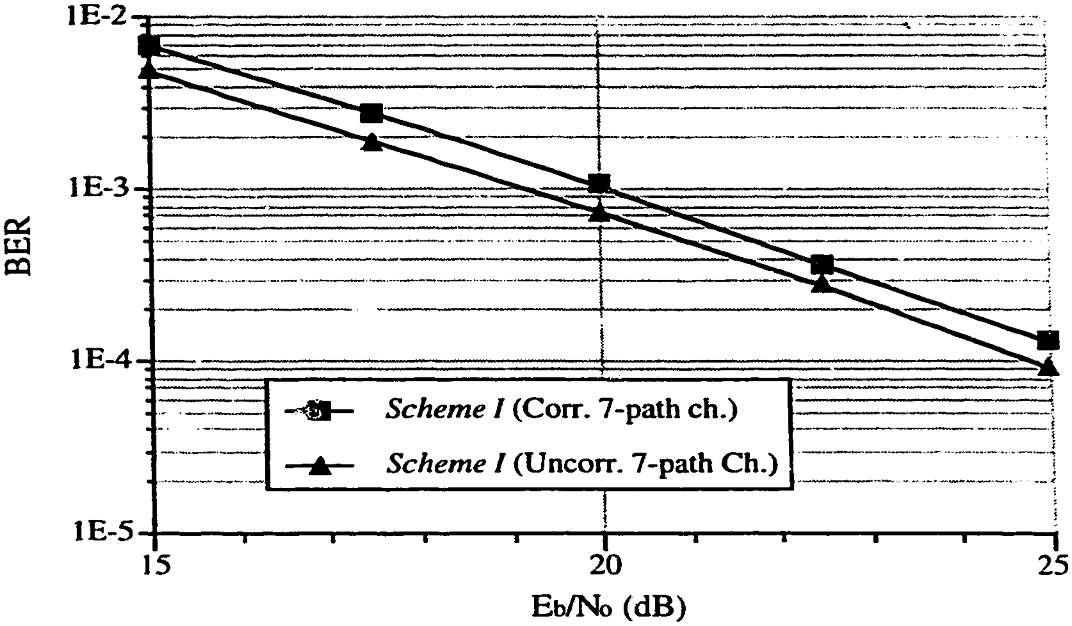

Fig. 5.37. Performance of *Scheme I* in a seven-path Rayleigh fading channel (with correlated or uncorrelated fading paths).

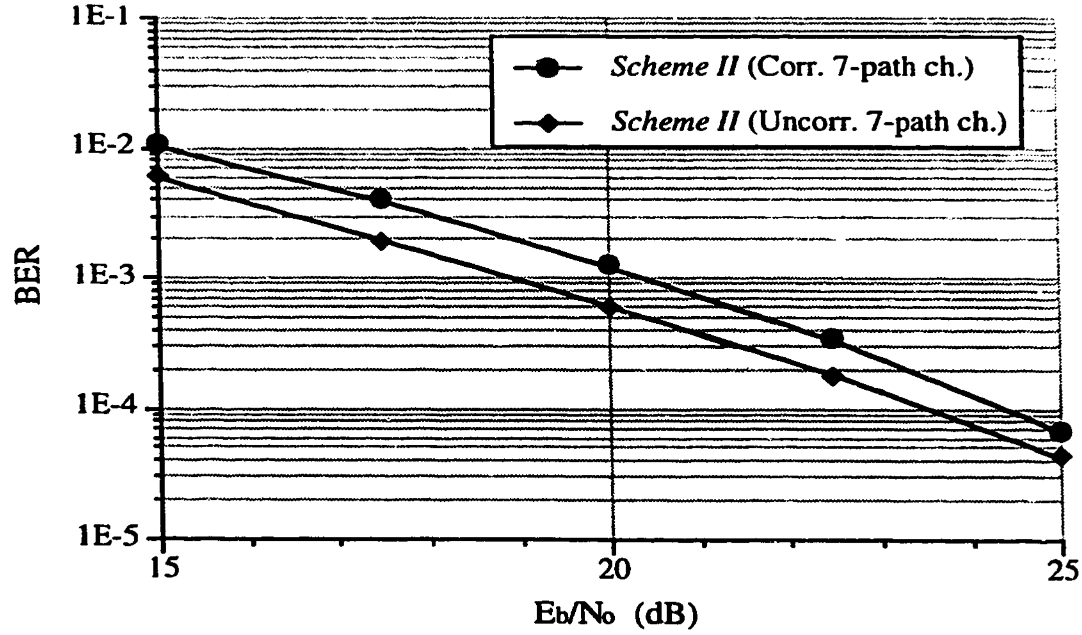

Fig. 5.38. Performance of *Scheme II* in a seven-path Rayleigh fading channel (with correlated or uncorrelated fading paths).

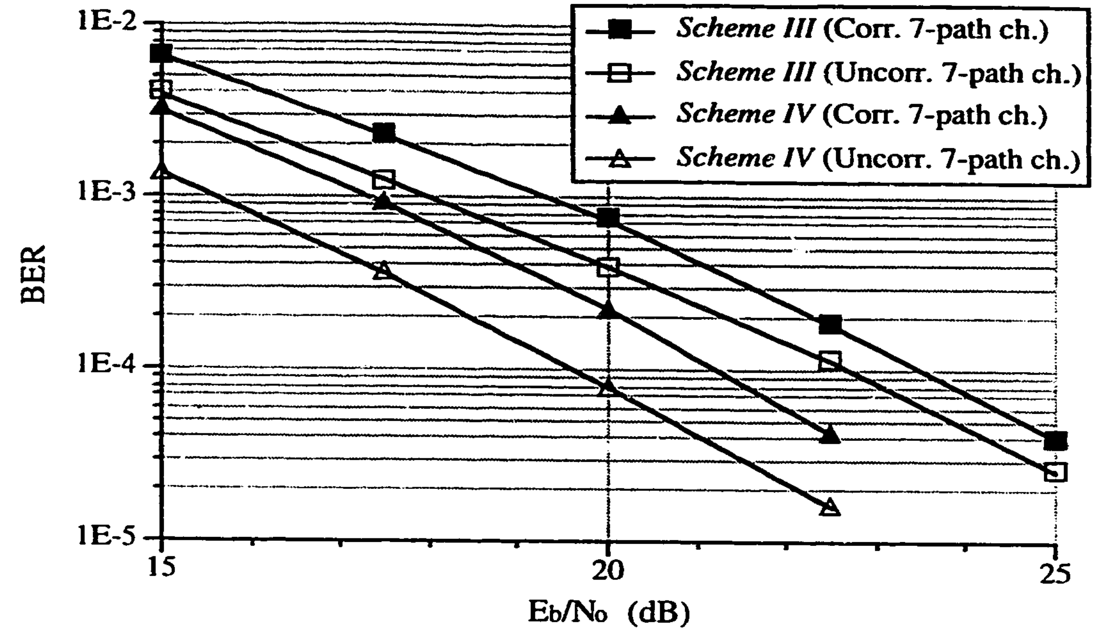

Fig. 5.39. Performance of *Schemes III* and *IV* in a seven-path Rayleigh fading channel (with correlated or uncorrelated fading paths).

The above results show that a high degree of envelope cross-correlation between resolvable paths of the channel (see (5.1)) has a negative impact on the performance of the four combined diversity-equalization schemes. This is because the equalizers used in these four schemes are capable of exploiting the time diversity inherent in the channel, and the gain provided by this inherent time diversity is reduced when fading among the discrete paths of the channel is not independent, which in turn leads to poorer system performance.

5.5. Summary

In this chapter, simulation models of the DECT baseline system as well as those employing performance enhancing techniques involving dual antenna diversity and decision feedback equalization were introduced. The simulation results were presented and discussed.

It has been shown that channel models with different average power delay profiles may have different effects on the simulated performance of the DECT system, particularly when antenna diversity selection and channel equalization are used. In most cases, the simple two-path Rayleigh fading channel model, with both paths having the same average power, tends to provide more optimistic results than those obtained with the six-path Rayleigh (or Rician) fading channels having an exponentially decaying power delay profile. The presence of a direct unfaded component in the first path of the six-path fading channel (Rician) improves the performance of the baseline system for a rms delay spread up to 0.1250*T*. However, it tends to degrade the system performance slightly at a large rms delay spread, as in the case where diversity selection is used. For a given channel power delay profile, envelope cross-correlation between resolvable (discrete) paths can also affect the performance of a system. Without equalization, a channel with independently fading paths may have a more negative impact on the link performance than one with correlated paths. In contrast, a system using decision feedback equalization performs better in the channel with independently fading paths. This is because the equalizer is also capable of making use of the time-diversity inherent in the channel, and the diversity gain is largest when the discrete paths are independent.

The simulation results obtained with a six-path Rayleigh (uncorrelated) fading channel have shown that the DECT baseline system fails to attain the target BER of 10^{-3} at a rms delay spread as low as 0.1250*T* (109 ns). Diversity selection introduced at the base station is the simplest means of improving the system performance. With either RSSI-driven or ERRI-driven diversity selection, the required BER of 10^{-3} at a rms delay spread of 0.1250*T* can be readily achieved. As expected, ERRI-driven diversity is superior to RSSI-driven diversity. Nevertheless, the error floor due to ISI persists and the system fails to achieve the required BER even with ERRI-driven diversity when the rms delay spread of the six-path fading channel increases beyond 0.2500*T* (217 ns). A DFE can effectively remove the error floor and enables the system to reach the required performance at rms delay spreads up to 0.3750*T*. The four combined diversity-equalization schemes can further improve the system performance. Provided with sufficient equalizer training, significant performance gain can be obtained by the four schemes, with *Scheme IV* being superior to the other three. With corresponding filters of the same length (8-tap FFF and 2-

tap FBF), *Scheme I* has a gain of about 5 dB over a (8,2)DFE, whereas *Schemes II* and *III* achieve an additional gain of about 2 dB at a BER of 10^{-4} in the six-path Rayleigh fading channel with a rms delay spread of $0.3750T$ (refer to Figure 5.36). In the same situation, *Scheme IV* yields a performance gain of about 4 dB over *Schemes II* and *III*.

6. SUMMARY AND CONCLUSIONS

Short-term fading and ISI arising from multipath propagation in radio channels can severely degrade the performance of digital wireless communication systems. Diversity combining and channel equalization are known to be effective techniques for mitigating fading and ISI, respectively. The application of dual antenna diversity and adaptive decision feedback equalization to an indoor wireless system employing GMSK modulation and time division duplexing was investigated in this thesis. To evaluate the performance of these techniques, extensive computer simulations were applied. In this study, DECT was used as the reference system.

In Chapter 2, the theory and modeling of radio channels, and the GMSK modulator and one-bit differential detector were discussed. They formed the basic components of the DECT baseline system model for computer simulation. Issues related to channel equalization, particularly decision feedback equalization and the standard RLS algorithm, and diversity combining were reviewed in Chapter 3. This was followed by the introduction of the four **combined** diversity-equalization schemes in Chapter 4, namely

- *Scheme I*: DFE with RSSI-driven diversity selection
- *Scheme II*: DFE with ERRI-driven diversity selection
- *Scheme III*: DFE with MSE-driven diversity selection
- *Scheme IV*: Integration of diversity combining and DFE through joint-parameter optimization

Simulation models and results for the baseline system and those associated with the different performance enhancing techniques were presented in Chapter 5. Several frequency selective fading channel models were used in this study. They included a two-path Rayleigh fading channel with the same average power in each path, a six-path Rayleigh (or Rician) fading channel with an exponentially decaying average power delay profile, and a seven-path Rayleigh fading channel with an experimentally derived average power delay profile having a fixed rms delay spread of $0.1663T$. For the two-path and six-path channel models, fading among the discrete paths was assumed to be independent, whereas for the seven-path channel model, two cases involving correlated and uncorrelated fading among the discrete paths were considered. The maximum Doppler shift of these channels was set at 5 Hz.

6.1. Research Results

It has been shown that the DECT baseline system failed to achieve a BER of 10^{-3} in either the two-path or six-path Rayleigh (or Rician) fading channels when the rms delay spread increased to 0.1250T (109 ns). Better system performance could be achieved when the fading among discrete paths was correlated, as demonstrated by the results obtained with the seven-path Rayleigh fading channel. However, even with correlated fading paths, the system performance was still marginal at E_b/N_o of 40 dB for a rms delay spread of 0.1663T (144 ns). By exploiting the reciprocity of the channel, antenna diversity selection implemented at the base station alone is a simple technique for enhancing the system performance of both the forward and reverse links.

It is necessary to recapitulate the various methods involving diversity selection before discussing their simulation results. For the forward link transmission, both the RSSI-driven and ERRI-driven diversity selection methods under investigation encounter an antenna switching delay of about half a frame duration (the frame duration for DECT is 10 ms). For the reverse link transmission, a switching delay of one frame duration is incurred in RSSI-driven diversity, whereas no such delay occurs in ERRI-driven diversity. These two diversity techniques are also incorporated directly with DFE to realize *Schemes I* and *II*. *Scheme III*, which also combines diversity selection and DFE, is similar to *Scheme II* except that the criterion for diversity selection is now driven by mean square errors from the DFEs in the two diversity branches. Before proceeding with discussions on the simulation results, it should also be noted that the performance of the two diversity methods and the related combined diversity-equalization schemes was only simulated for the reverse link of the DECT system. However, their forward link performance can also be inferred from the corresponding simulation results of the reverse link. In principle, the basic RSSI-driven diversity selection method and *Scheme I* should perform better on the forward link than on the reverse one, since the antenna switching delay is longer for the reverse link. However, this improvement will not be significant in a slowly fading radio channel, since the time difference in the antenna switching delays for both links is relatively small. On the other hand, the basic ERRI-driven diversity selection method together with *Schemes II* and *III* may suffer some degradation in their forward link performance due to the antenna switching delay of half a frame duration, which does not exist in the reverse link. Again, as a result of slow fading, the degradation will not be significant. *Scheme IV* achieves the same performance on both links since no switching delay is encountered in either link. Lastly, it should be noted that diversity branches were assumed to be independent in the simulations.

The simulation results have shown that the system performance could be improved considerably with either RSSI-driven or ERRI-driven diversity selection. The influence of an average power delay profile of a channel upon the simulated performance became more obvious, particularly for ERRI-driven diversity. The six-path Rayleigh fading channel produced the worst link performance, whereas the two-path Rayleigh fading channel always yielded more optimistic results. For example, at a rms delay spread of 0.1250*T*, the simulated BERs of the system employing ERRI-driven diversity in the two-path and six-path Rayleigh fading channels were $2\text{x}10^{-5}$ and $8\text{x}10^{-5}$, respectively, for E_b/N_o of 40 dB. Results obtained with the seven-path Rayleigh fading channel have again demonstrated that a better performance could be expected when fading among the discrete paths of the channel was correlated. However, according to the more conservative estimates obtained with the six-path Rayleigh fading channel, RSSI-driven and ERRI-driven diversity failed to attain a BER of 10^{-3} at rms delay spreads of 0.1875*T* (163 ns) and 0.2500*T* (217 ns), respectively. This indicates that channel equalization is required if the system were to operate in a large indoor environment, such as an airport terminal or a large warehouse where rms delay spreads up to 300 ns may occur.

It has been shown that a DFE is capable of improving the performance of the DECT system in a time dispersive channel. The standard RLS algorithm was used for equalizer adaptation. By using a DFE (with six *T*/2-spaced taps and two *T*-spaced taps in its FFF and FBF, respectively) and the 16-bit packet synchronization word in the DECT time-slot as a training sequence, the DECT system was able to achieve the required BER of 10^{-3} for a rms delay spread up to 0.3750*T*. However, the DFE had no advantage over ERRI-driven diversity selection when the rms delay spread of the channel was small (e.g. 0.1250*T* or less). The performance of the DFE had been penalized by insufficient training. In order to fully exploit the DFE, the number of training bits must be increased to at least thirty-two. With a 32-bit training sequence, it could achieve an additional gain of 2.5 dB at a BER of 10^{-3} in the six-path Rayleigh fading channel with a rms delay spread of 0.3750*T*. As before, the two-path Rayleigh fading channel tended to provide more optimistic results. On the other hand, in contrast to the previous case where the baseline system performed better in the channel with correlated fading paths, the DFE performed better in the channel with independently fading paths. This is because a DFE is also capable of exploiting time diversity inherent in the multipath channel.

To optimally mitigate both multipath fading and ISI simultaneously, the combined use of channel equalization and antenna diversity has been investigated. It has been shown that, by simply combining the DFE mentioned above with RSSI-driven diversity selection, and using a 16-bit training sequence, *Scheme I* was able to achieve an additional gain of about 5 dB at a BER of 10^{-4} over either ERRI-driven diversity or the DFE in the six-path Rayleigh fading channel with a rms delay spread of 0.1250*T*. Under the same conditions, *Schemes II* and *III* performed equally well with a gain of about 4 dB over *Scheme I*. Due to the short training sequence, the performance of *Scheme IV* was not much better than *Schemes II* and *III*. When the number of training bits was increased to thirty-two, *Scheme IV* was able to attain a BER of 10^{-4} at an E_b/N_o of about 3 dB less than needed for *Schemes II* and *III*, even though it only had four taps in each of its FFFs. With sufficient equalizer training, the DFE and the four schemes performed better in a channel with a larger rms delay spread. Increasing the number of taps in the corresponding FFFs could further improve the performance in such a channel. Table 6.1 summarizes the E_b/N_o required by the DFE and the four schemes, all using an eight-tap FFF and a two-tap FBF, to achieve a

BER of 10^{-4} in the six-path Rayleigh fading channel with a rms delay spread of $0.3750T$, when a 48-bit training sequence was used.

	(8,2)DFE	*Scheme I*	*Scheme II*	*Scheme III*	*Scheme IV*
E_b/N_o (dB)	29.3	24.5	22.5	22.5	18.3

Table 6.1. E_b/N_o required by different schemes to attain a BER of 10^{-4} at a rms delay spread of $0.3750T$.

Re-evaluating the performance of these four schemes using the seven-path Rayleigh fading channel demonstrated that they all encountered performance degradation in the channel with correlated fading paths, in which an additional E_b/N_o varying from about 1 dB in *Scheme I* to 1.7 dB in *Scheme IV* was required to maintain a BER of 10^{-4}. However, all of the simulation results were consistent in showing that the four schemes could significantly improve the system performance, with *Scheme IV* being superior to the others.

Among the four combined diversity-equalization schemes, *Schemes I, II* and *III* are best suited for small handsets since diversity antennas are only implemented at the base station to realize diversity gains for both links. The increased complexity of the handset receiver due to the introduction of a DFE is of no major concern, as more compact and power efficient digital signal processing devices continue to emerge. Furthermore, with these performance enhancing schemes, both the base station and the portable can now transmit at a lower power and still maintain good link quality. This, in particular, will help to increase the battery life-span of the portable, and to reduce the electromagnetic radiation level and potential related health hazards, if any. Lowering transmission power can also reduce co-channel interference, which in turn will improve system performance or increase system capacity. *Scheme IV*, which requires diversity antennas to be implemented at both ends of the link, is not suitable for small handsets. However, it is suited for larger portable terminals such as portable computers.

6.2. Recommendations for Future Work

This research work has demonstrated that the combined use of antenna diversity and channel equalization is very effective in enhancing the performance of an indoor wireless system employing GMSK modulation and time division duplexing by combating slow fading and ISI. The encouraging results of this preliminary investigation provide a motivation for further studies in order to evaluate the performance of the different schemes in greater detail.

In this project, the evaluation of the various schemes involving diversity selection is based on the assumption that the signals from the diversity branches fade independently. In practice, a certain amount of envelope cross-correlation exits between these signals, i.e., they may encounter correlated fading. The reduction of the diversity gain due to an increase in the cross-correlation between diversity branches will degrade the performance of the four schemes. Thus, it is desirable to characterize the performance of these schemes against cross-correlation coefficients between the diversity branches up to, for example, 0.5. Also, the maximum Doppler shift considered in this study was 5 Hz (corresponds to the portable speed of 0.8 m/s at carrier frequency of 1.88 GHz). In warehouses or factories where there is faster moving machinery, the maximum Doppler shift in those environments can be higher. It will then be necessary to again evaluate the performance of *Schemes I, II, III* and *IV* at a higher maximum Doppler shift. Lastly, the study has shown that these schemes perform better at a higher rms delay spread (normalized to bit interval), and their performance can be further improved by increasing the number of taps in the feedforward filter of the equalizer. Therefore, the investigation should also be extended to the optimization of these schemes in systems with higher transmission bit rates.

Bibliography

[1] A. Robin Potter, "Implementation of PCNs using DCS1800", *IEEE Commun. Mag.*, vol. 30, no. 12, Dec. 1992, pp. 32-36.

[2] Jay E. Padgett, Christoph G G. Gunther, and Takeshi Hattori, "Overview of wireless personal communications", *IEEE Commun. Mag.*, vol. 33, no. 1, Jan. 1995, pp. 26-41.

[3] John Gardiner and Barry West, *Personal Communication Systems and Technologies*, Artech House, Inc., 1995, pp. 105-125.

[4] L.B. Lopes, "On the radio link performance of the Digital European Cordless Telecommunications (DECT) system", *Proc. IEEE Globecom 90*, San Diego, Dec. 1990, pp. 1012-1017.

[5] "Radio equipment and system; Digital European Cordless Telecommunications (DECT) Common interface", ETSI 300 175-2, Oct. 1992.

[6] ETSI DECT recommendation Part 3: "Medium access control layer", DE/RES 3001-3, Feb. 1992.

[7] ETSI DECT recommendation Part 2: "Physical layer", DE/RES 3001-2, Feb. 1992.

[8] Walter H. W. Tuttlebee, "Cordless personal communications", *IEEE Commun. Mag.*, vol. 30, no. 12, Dec. 1992, pp. 42-53.

[9] Homayoun Hashemi, "The indoor radio propagation channel", *Proc. IEEE*, vol. 81, no.7, July 1993, pp. 943-968.

[10] Michel Daoud Yacoub, *Foundations of Mobile Radio Engineering*, CRC Press, Inc., 1993, pp. 130.

[11] Asha Mehrotra, *Cellular Radio Performance Engineering*, Artech House, Inc. 1994.

[12] William C.Y. Lee, *Mobile Communications Design Fundamentals*, 2nd Edition, John Wiley & Sons, Inc., 1993.

[13] John G. Proakis, "Adaptive equalization for TDMA digital mobile radio", *IEEE Trans. Vehic. Technol.*, vol. 40, no. 2, May 1991, pp. 333-341.

[14] Adel A.M. Saleh and Reinaldo A. Valenzuela, "A statistical model for indoor multipath propagation", *IEEE JSAC.*, vol. SAC-5, no. 2, Feb. 1987, pp. 128-137.

[15] Homayoun Hashemi, "A study of temporal and spatial variations of the indoor radio propagation channel", *Proc. PIMRC '94* , Amsterdam, 1994, pp. 127-134.

[16] Peter Karlsson, Henrik Borjeson and Torleiv Maseng, "A statistical multipath propagation model confirmed by measurements and simulations in indoor environments at 1800 MHz", *Proc. PIMRC '94,* Amsterdam, 1994, pp. 149-155.

[17] S. Hamidreza Jamali, Tho Le-Ngoc, *Coded-Modulation Techniques for Fading Channels*, Kluwer Academic Publishers, 1994, pp.5.

[18] W.C. Jakes, Jr., *Microwave Mobile Communications*, John Wiley & Sons, Inc., 1974, pp. 70.

[19] J. D. Parson, *The Mobile Radio Propagation Channel*, John Wiley & Sons, Inc., 1992, pp. 126.

[20] Said M. Elnoubi, "Analysis of GMSK with differential detection in land mobile radio channels", *IEEE Trans. Vehic. Technol.,* vol. VT-35. no. 4, Nov. 1986, pp. 162-167.

[21] A. Papoulis, *The Fourier Integral and its Applications*, McGraw Hill, 1962.

[22] J.C. Bic, D.Duponteil and J.C. Imbeaux, *Elements of Digital Communication*, John Wiley & Sons Ltd., 1991, pp. 223.

[23] Marvin K. Simon and Charles C. Wang, "Differential detection of Gaussian MSK in a mobile radio environment", *IEEE Trans. Vehic Technol,* vol. VT-33, no.4, Nov. 1984, pp. 307-320.

[24] Shahid U.H. Qureshi, "Adaptive equalization", *Proc. IEEE,* vol. 73, no. 9, Sept. 1985, pp. 1349-1387.

[25] B.P. Lathi, *Modern Digital and Analog Communicatton Systems,* 2nd Edition, The Dryden Press, Saunders College Publishing, 1989, pp. 179-184.

[26] Peter Monsen, "Feedback equalization for fading dispersive channels", *IEEE Trans. Info. Theory,* Jan. 1971, pp. 56-64.

[27] J. Salz, "Optimum mean-squares decision feedback equalization", *The Bell System Technical Journal,* vol. 52, no. 8, Oct. 1973, pp. 1341-1373.

[28] Richard D. Gitlin, Jeremiah F. Hayes and Stephen B. Weinstein, *Data Communications Principles,* Plenum Press, New York, 1992, pp. 508.

[29] John .A. C. Bingham, *The Theory and Practice of Modem Design,* John Wiley & Sons, Inc., 1988, pp. 292-293.

[30] R.A. Ziegler, N.M.W. Al-Dhahir and J.M. Cioffi, "Nonrecursive adaptive decision-feedback equalization from channel estimates", *Proc. IEEE Technol. Conf.,* Denver, 1991, pp. 600-603.

[31] Simon Haykin, *Adaptive Filter Theory,* 2nd Edition, Prentice Hall, Inc. Englewood Cliffs, New Jersey, 1991.

[32] L. Ling, D. Manolakis and J. G. Proakis, "Finite word-length effects in recursive least squares algorithms with application to adaptive equalization", Ann. Télécommun., vol. 41, no. 5-6, 1986, pp. 328-336.

[33] Giovanna D'Aria, Roberto Piermarini and Valerio Zingarelli, "Fast adaptive equalizers for narrow-band TDMA mobile radio", *IEEE Trans. Vehic. Technol.,* vol. 40, no.2, May 1991, pp. 392-404.

[34] John G. Proakis, *Digital Communications,* McGraw-Hill, Inc., 2nd Edition, 1989.

[35] William C. Y. Lee, *Mobile Communications Engineering,* McGraw-Hill, Inc. 1982.

[36] Stephen R. Todd, Mohammed S. El-Tanany, and Samy A. Mahmoud, "Space and frequency diversity measurements of the 1.7 GHz indoor radio channle using a four-branch receiver", *IEEE Trans. Vehic. Technol,* vol. 41, no. 3, Aug. 1992, pp. 312-320.

[37] Shu Lin and Daniel J. Costello, Jr., *Error Control Coding, Fundamentals and Applications*, Prentice-Hall, Inc., Englewood Cliffs, N.J., 1983, pp. 102-103.

[38] Philip Balaban and Jack Salz, "Optimum diversity combining and equalization in digital data transmission with applications to cellular mobile radio -- Part I: Theoretical Considerations", *IEEE Trans. Commun.*, vol. 40, no. 5, May 1992, pp. 885-894.

[39] Michel C. Jeruchim, Philip Balaban and K. Sam Shanmugan, *Simulation of Communication Systems*, Plenum Press, New York, 1992, pp.463-550.

[40] "Report On RF Channel Characterization", JTC(AIR)/93.09.23-238R2.

APPENDIX A

C-Programs for the baseline system. They include:

i. baseline.c (the main program)
ii. source.c (time-slot generator)
iii. Gaussian_filter.c (Gaussian pulse shaping filter)
iv. modulator.c. (CPFSK modulator)
v. 6-path_rayleigh_ch.c (six-path Rayleigh fading channel)
vi. clk.c (sampling phase estimator)
vii. rand.c (AWGN generator)
viii rx_filter.c (receiver filter)
ix detector.c (one-bit differential detector)

```
/*------------------------------------------------ baseline.c ------------------------------------------------*/
/* This is the baseline system with no equalization or diversity.          */

#include "spec.h"
#include "random.h"

/* Function prototypes. */

void data_source(int *);
void gaussian_filter(const int, double *);
void fm_mod(const double, cmplx_t *);
void channel(const cmplx_t, const int, cmplx_t *, int *);
double randn(long *);
void initrandu(FILE *);
void closeru(FILE *);
void clk(const cmplx_t, int *);
void rx_filter(const cmplx_t, cmplx_t *);
void detect(const cmplx_t, int *);

main ()
{
int  iter, i, j, nbit,
        skip = 176640, /*'skip' = 23 packets x 16 samples per bit x 480 bits per packet. 'skip' is the
                        amount of 'time' that the channel is 'fast-forward' after each packet is transmitted.
                        Each actual DECT packet contains 424 data bits (include sync word, control,
                        CRC code etc) and guard band, which is equivalent to 56-bit interval.  */
        new_frame = 1,
        slot[BIT_PER_SLOT], /*  Data slot  */
        time_rec = 0, /* counter set for clock recovery mechanism.     */
        samp_count = 0, /* counter set for a sampler at the receiver.  */
        flag, /* flag for the clock recovery.                 */
        delay = 2000, /* excess delay introduced by the receiver filter with respect to the instant when
                        timing is recovered. The initial value of 'delay' is set to a large value so as to
                        inhibit the action of the sampler before the proper clock-phase is recovered.  */
        x,
        detect_bit; /* detected bit.                          */
```

```
long rseed = 10;

double  fil_out,  /* pre-modulation filter output.            */
        ch_nvolt = 4*sqrt(0.5/pow(10,(double)SNR/10.0)),  /*  Voltage of AWGN before bandpass
                         filtering. 0.5 is the average signal power. Factor 4 is introduced to give an
                         appropriate SNR at the receiver filter output; it is squared root of the
                         oversampling factor, which is 16. */
        pow = 0, /*  average complex signal power at the channel output.  */
        pow_count = 0,
        error = 0,  /* number of bit errors counted.            */
        error_count = 0;  /* number of bits processed.            */

cmplx_t  mod_out,  /* modulator output.                          */
        ch_out,  /* channel output.                         */
        rx_fil_out;  /* receiver filter output.              */

FILE *fp,   /*  pointer to a data file named rand.dat  */
        *datf;

fp = fopen("rand.dat","r+");

initrandu(fp);  /*   this function is found in the file named rand.c and it must be invoked number generator
                   called randn, which is meant to to produce zero mean unit variance random number. The
                   function initrandu() write data from a file called rand.dat. These data are used to generate
                   random numbers.    */

for (iter = 0; iter < MAX; ++iter) {

        x = 64; /* the 3th sync bit arrive at the output of the channel is at about x=74. Set the lower
                bound to trap the approximate clock phase.*/
        time_rec = 0;
        samp_count = 0;
        delay = 2000001;
        i = 3;  /* start error checking from the 4th bit of every packet.        */

        data_source(slot);

        for (nbit = 0; nbit < BIT_PER_SLOT; ++nbit)  { /* Each bit is sampled 16 times. */
          for (j = 0; j < 16; ++j) {
            gaussian_filter(slot[nbit], &fil_out);
            fm_mod(fil_out, &mod_out);
            channel(mod_out, skip, &ch_out, &new_frame);
            pow += ch_out.real*ch_out.real + ch_out.imag*ch_out.imag;
            ++pow_count;
            ++time_rec;
            ++samp_count;

            if (time_rec > x) {
              clk(ch_out, &flag);

              if (flag == 1) {

               delay = 25;  /* this is an excess delay of 25 samples introduced by the receiver filter.     */
                x = 9000000; /* inhibit clock_recovery mechanism.       */
                time_rec = 0; /* reset time recovery counter.            */

              }
              samp_count = 1;  /* reset counter for sampler.              */
              }

            ch_out.real += ch_nvolt*randn(&rseed); /*Adding AWGN to the channel output. */
             ch_out.imag += ch_nvolt*randn(&rseed);
              rx_filter(ch_out, &rx_fil_out);

         /*  Sample, equalize and demodulate the received signal.        */
              if (samp_count == delay) {
                 detect(rx_fil_out, &detect_bit);
                  delay += 16;

                  if (flag > 1) {

                    if (detect_bit != slot[i])
                        ++error;
                     ++i;
                    ++error_count;
                  }

                  flag++;

               }
            }
          if (nbit == 420) {/*inhibit signal processing at the receiver and update error counts and channel
                                        parameters.*/
              samp_count = 0;
              new_frame = 1;
              delay = 20000000;
            }
          }
      }

     datf = fopen("nd_uncor1.d","w");
      fprintf(datf, "%0.0f\t %0.0f\t %f\t %f\n", error_count, error,
            error/error_count, pow/pow_count);

      closeru(fp);     /* overwrite the data file rand.dat with new values. */
       fclose(fp);
      fclose(datf);
}

/*-------------------------------------------- end of file -------------------------------------------*/



/*------------------------------------------------ source.c -------------------------------------------*/
/* The code generates a single time slot. It is not the exact DECT slot format; the actual timing sequence is
not used. However, the same packet-synchronization bit sequence is used.    */

#include "spec.h"

long  seed = 3;

void
data_source(int slot[])
{
        int  i;
        static int  start = 1;

        if (start == 1)  {
                srand48(seed);  /* initializing entry point for mrand48().          */
                start = 0;
        }

        /* The S-field of the DECT packet, which consists of 16-bit preamble word
           and 16-bit synchronization word, is as follow:
        s_field  {0,1,0,1, (Note that only 4 preamble bits are used instead of the
                    original 16 preamble bits used for clock  recovery.)
                    0,0,0,1,0,1,1,0,0,1,1,1,0,1,0,1}                          */

        slot[0]=slot[2]=slot[4]=slot[5]=slot[6]=slot[8]=slot[11]=slot[12]=
        slot[16]=slot[18] = -1;
        slot[1]=slot[3]=slot[7]=slot[9]=slot[10]=slot[13]=slot[14]=slot[15]=
        slot[17]=slot[19] = 1;

        for (i = 20; i < BIT_PER_SLOT; ++i)  {
                if (mrand48() > 0) {
                slot[i] = 1;

                else {
                        slot[i] = -1;
                }
        }

}
/*--------------------------------------------- end of file ---------------------------------------------*/
```

```
/*---------------------------------------------- Gaussian_filter.c ----------------------------------------*/
/* Premodulation Pulse Shaping Filter  */
/* A low-pass Gaussian filter is realized with a transversal filter structure. Filter constants, A and B, are
predetermined. For DECT GMSK transmitter, B is determined from BT = 0.5, where T is the bit interval,
and A is chosen such that the response of the filter to a rectangle pulse (bit) at its input has an area of 0.5.
*/

#include "spec.h"
#define n_taps 51  /* number of taps of a transversal gaussian filter.      */

void
gaussian_filter(const int in_samp,  /* input sample.                    */
                double *filter_out,  /* output of the filter.            */
                const int *new_frame
                )
{
int  i,  taps[n_taps];

static int  start = 1;

double  sum, A, B;

static double  h[n_taps],   /* Tap-weights of the FIR Gaussian filter.     */
               buffer[n_taps];   /* buffer for storing input samples at various taps.  */
sum = 0.0;

/* Initialize the buffer of the FIR filter and define the filter tap-weight.  */

if (start == 1) {
        A = 1/32.0;     /* Gaussian filter constant.                  */
        B = 6.25e5;     /* 3 dB bandwidth of the filter.              */

        for (i = 0; i < n_taps; ++i) {
                taps[i] = i - (n_taps-1)/2;
                h[i] = A*sqrt(2*PI/log(2))*B*exp(-2*square(PI*B*Ts*taps[i])/log(2));

        }

        start = 0;
}

if (*new_frame == 1) {
        for (i = 0; i < n_taps; ++i)
        buffer[i] = 0;
}
/* Shift the contents of the buffer before receiving a new sample.          */

for (i = 1; i < n_taps; ++i) {
        buffer[n_taps-i] = buffer[n_taps-1-i];
        }

        buffer[0] = in_samp;   /* input a new sample.                        */

/* Compute the filter output.                                    */

for (i = 0; i < n_taps; ++i) {
        sum += h[i]*buffer[i];
}

*filter_out = sum;
}

/*------------------------------------------- end of file -----------------------------------------*/



/*------------------------------------------ modulator.c -----------------------------------------*/
/* FM Modulator                                              */
/* The integrator of the FM modulator is implemented with slightly modified Simpson's one-third rule. The
output of the integrator is bounded by 1  and -1. The output of this function is a complex baseband signal
with transmitted information carried in its phase.    */

#include "spec.h"

void
fm_mod(const double in_samp,  /* input pulse sample.                   */
           cmplx_t *signal,       /* output complex baseband signal           */
           const int *new_frame
           )
```

```
{
static double samp[3] ,
          mod_out ;

/* Reset the  condition for every new slot for the purpose of synchronizing the  receiver to the transmitter.
*/

if (*new_frame == 1) {
          samp[0] = samp[1] = samp[2] = 0.0;
          mod_out = 0.0;  /* reset initial condition to 0 for every new frame.     */
}

samp[2] = samp[1];
samp[1] = samp[0];
samp[0] = in_samp;

mod_out += (samp[0] + 4*samp[1] + samp[2])*Ts/6;

/* output of the modulo integrator is bounded between +1 and -1.           */
while (mod_out > 1)
          mod_out -= 2;

while (mod_out < (-1))
          mod_out += 2;

/* Create a complex baseband signal.                              */

signal->real = cos(mod_out*PI);
signal->imag = sin(mod_out*PI);

}

/*--------------------------------------------------- end of file ---------------------------------------------------*/



/*------------------------------------------------ 6-path_rayleigh_ch.c ------------------------------------------------*/
/* A 6-path Rayleigh Fading Channel with uncorrelated paths.                  */

#include "spec.h"
#include <malloc.h>

#define   wm    2*PI*5  /* maximum Doppler shift in radian.            */
#define   N_oscil   26 /* N_oscil = number of oscillators (per fader). */
#define   no_path  6    /* number of independent paths (faders).       */
#define nr  sqrt(N_oscil-1)
#define ni  sqrt(N_oscil)

void *malloc(size_t);
double  oscil;

/*      Rayleigh Fader (Jakes' Model).                          */
void
rayleigh(int j,   /* designate fader for j-th path.              */
         int skip, /* duration between successive transmissions from the same portable in terms of number
                    of samples (i.e. the amount of time (in sample unit) the channel is fast forward for each
                    update).  */
```

```
        cmplx_t *fade  /* complex fading signals.                    */
        )
{
int  i, k;
static int start = 1;
double two_pi = 2*PI;
static double  gain_re[N_oscil], gain_im[N_oscil-1], phase[no_path][N_oscil-1], wn[N_oscil-
        1],wmt[no_path], phase_inc[N_oscil-1], wmt_inc;

f ( start == 1)  {
        for (i = 1; i < N_oscil; ++i) {
                gain_re[i-1] = 2*cos(PI*i/N_oscil);
                gain_im[i-1] = 2*sin(PI*i/N_oscil);
                wn[i-1] = wm*cos(2*PI*i/(4*N_oscil -2));
                for (k = 1; k <= no_path; ++k) {
                        phase[k-1][i-1] = PI*(i+2*(k-1))/N_oscil+10*wn[i-1]*k;

        /* note that this equation is slightly modified such that each fader starts at a different time, i.e.
           10*k seconds from reference time of 0. This is to ensure uncorrelated faders.  */

           }
        phase_inc[i-1] = wn[i-1]*skip*Ts;
        }

        gain_re[N_oscil-1] = sqrt(2);

        for (k = 0; k < no_path; ++k)
                wmt[k] = 10*wm*(k+1);  /* This is to ensure that each fader start at a different time
                        instant.        */

        wmt_inc = skip*Ts*wm;
         start = 0;
}

wmt[j-1] += wmt_inc;
while (wmt[j-1] > two_pi)
        wmt[j-1] -= two_pi;

fade->real = gain_re[N_oscil-1]*cos(wmt[j-1]);
fade->imag = 0.0;

for (i = 1; i < N_oscil; ++i) {
        phase[j-1][i-1] += phase_inc[i-1];
        while (phase[j-1][i-1] > two_pi)
        phase[j-1][i-1] -= two_pi;
        oscil = cos(phase[j-1][i-1]);
        fade->real += oscil*gain_re[i-1];
        fade->imag += oscil*gain_im[i-1];
}
fade->real = fade->real/nr;
fade->imag = fade->imag/ni;

}

/*-----------------------------------------------------------------*/
/* Time Dispersive (frequency selective) Radio Channel.
Indoor radio channel is modelled by a tapped delay line. It consists of 6 independent Rayleigh fading paths.
The average impulse response of the channel has an exponentially decaying power profile.            */
```

```
void
channel(const cmplx_t signal_in,  /* input complex baseband signal.  */
          const int skip,  /*  update parameter. */
          cmplx_t *signal_out,       /* output complex baseband signal.  */
          int *update  /* flag for updating channel parameters.        */
          )
{
int   i;

static int   start = 1, N,
          nfad[no_path] = {1,2,3,4,5,6},/*designate Rayleigh faders for each discrete path. */
          delay[no_path];
          static double  *reg_real,  /* pointer to a buffer for holding tap-inputs (in-phase component).  */
          *reg_imag,  /* pointer to a buffer for holding tap-inputs (quadrature component). */
          ch_gain,    /* approximated channel gain.          */
          attn[no_path],  /* Attenuations of the multipaths.     */
          fad_re[no_path],  /* fad_re[] and fad_im[] are real and imaginary components       */
          fad_im[no_path];  /* of fader's output for each of the six independent paths. */

cmplx_t  fader, attn_sig;

if (start == 1) {

  /* Each path (ray) is independent of the other (from empirical calculations, it was found that the absolute
value of the envelope cross-correlation coefficient is less than 0.05. So, it is justifiable to say that the
power gain of the channel is the sum of the averaged powers of all paths in the channel. Since each
Gaussian random generator modelled by Jakes' fader has a variance of 1, the mean power of the Rayleigh
generator is 2.  */

          ch_gain = 0;

   /* Choose appropriate parameter values for a given rms delay spread.     */
          switch (spread) {

      /* delay = relative delay normalized to Ts.                         */
          case 50:
              delay[0] = 0;   delay[1] = 1;   delay[2] = 2;
              delay[3] = 3;   delay[4] = 4;   delay[5] = 5;
                N = 6;
                for (i = 0; i < 6; ++i) {
                    attn[i] = sqrt(2*exp(-delay[i]*Ts*1e9/55.9));
                  /* the constant 2 in the above expression is arbitrary.  */

                    ch_gain += 2*exp(-delay[i]*Ts*1e9/55.9);
                }

              ch_gain = sqrt(2*ch_gain); /* the constant 2 is to take into account the power of the Rayleigh
                                                                  fader.  */
                break;

          case 100:
              delay[0] = 0;   delay[1] = 2;   delay[2] = 4;
              delay[3] = 6;   delay[4] = 8;   delay[5] = 10;
                N = 11;
                for (i = 0; i < 6; ++i) {
                    attn[i] = sqrt(2*exp(-delay[i]*Ts*1e9/111.8));
                    ch_gain += 2*exp(-delay[i]*Ts*1e9/111.8) ;
```

```
     }
    ch_gain = sqrt(2*ch_gain);
    break;

case 150:
    delay[0] = 0;   delay[1] = 3;  delay[2] = 6;
    delay[3] = 9;   delay[4] = 12;  delay[5] = 15;
     N = 16;
     for (i = 0; i < 6; ++i) {
       attn[i] = sqrt(2*exp(-delay[i]*Ts*1e9/167.7));
        ch_gain += 2*exp(-delay[i]*Ts*1e9/167.7) ;
     }
    ch_gain = sqrt(2*ch_gain);
     break;

case 200:
    delay[0] = 0;   delay[1] = 4;  delay[2] = 8;
    delay[3] = 12;  delay[4] = 16;  delay[5] = 20;
     N = 21;
     for (i = 0; i < 6; ++i) {
       attn[i] = sqrt(2*exp(-delay[i]*Ts*1e9/223.6));
        ch_gain += 2*exp(-delay[i]*Ts*1e9/223.6);
     }
    ch_gain = sqrt(2*ch_gain);
    break;

case 250:
    delay[0] = 0;   delay[1] = 5;  delay[2] = 10;
    delay[3] = 15;  delay[4] = 20;  delay[5] = 25;
     N = 26;
     for (i = 0; i < 6; ++i) {
       attn[i] = sqrt(2*exp(-delay[i]*Ts*1e9/279.5));
        ch_gain += 2*exp(-delay[i]*Ts*1e9/279.5);
     }
    ch_gain = sqrt(2*ch_gain);
     break;

   case 300:
    delay[0] = 0;   delay[1] = 6;  delay[2] = 12;
    delay[3] = 18;   delay[4] = 24;  delay[5] = 30;
     N = 31;
     for (i = 0; i < 6; ++i) {
       attn[i] = sqrt(2*exp(-delay[i]*Ts*1e9/335.4));
        ch_gain += 2*exp(-delay[i]*Ts*1e9/335.4);
     }
    ch_gain = sqrt(2*ch_gain);

     break;

case 350:
    delay[0] = 0;   delay[1] = 6;  delay[2] = 12;
    delay[3] = 18;   delay[4] = 24;  delay[5] = 30;
     N = 31;
     for (i = 0; i < 6; ++i) {
       attn[i] = sqrt(2*exp(-delay[i]*Ts*1e9/417.9));
        ch_gain += 2*exp(-delay[i]*Ts*1e9/417.9);
     }
    ch_gain = sqrt(2*ch_gain);
```

```
            break;

         default:
            printf(" invalid value: choose 50, 100, 150, 200, 250, 300 or
                  350 ns for rms delay spread.\n");
         }

      /* Allocate sufficient memory to buffers.                     */
       reg_real = (double *)malloc(N*sizeof(double));
       reg_imag = (double *)malloc(N*sizeof(double));

      /* Initialize buffers for proper tap-inputs                 */
         for (i = 0; i < N ; ++i)  {
            reg_real[i] = 0.0;
            reg_imag[i] = 0.0;
          }
        start = 0;
         }

         for (i = 1; i < N; ++i)  {
         reg_real[N-i] = reg_real[N-1-i];  /* shifting contents.       */
         reg_imag[N-i] = reg_imag[N-1-i];
      }

      reg_real[0] = signal_in.real;
       reg_imag[0] = signal_in.imag;

      if (*update == 1) {
         for (i = 0; i < no_path; ++i)  {
           rayleigh(nfad[i], skip, &fader);
           fad_re[i] = fader.real;
           fad_im[i] = fader.imag;
         }
        *update = 0;
       }

       signal_out->real = 0.0;
       signal_out->imag = 0.0;

   /* Calculate the channel output and normalized it to the channel gain.    */

       for (i = 0; i < no_path; ++i) {
         attn_sig.real = reg_real[delay[i]]*attn[i];
         attn_sig.imag = reg_imag[delay[i]]*attn[i];
         signal_out->real += attn_sig.real*fad_re[i] - attn_sig.imag*
                          fad_im[i];
          signal_out->imag += attn_sig.real*fad_im[i] + attn_sig.imag*
                          fad_re[i];
          }

      signal_out->real /= ch_gain;
      signal_out->imag /= ch_gain;

     }

/*---------------------------------------------- end of file ----------------------------------------------*/
```

```
/*------------------------------------------------ clk.c ------------------------------------------------*/
/* This is used to estimate the optimum clock phase at the channel output. It is based on tracking the 3rd
preamble bit. The preamble word starts with -1, 1, -1,1,1, ......, . Assuming that the initial phase of the
signal is zero. In tracking the 3rd bit of the preamble word, we just need to track the the second minimum
phase value. The first minimum phase of-course corresponds to the first bit. Note that the phase respond is
bounded by -pi and +pi and the phase transition between 2 adjacent symbols is less than 0.5 pi.     */

#include "spec.h"
void
clk(const cmplx_t ch_out,
         int *flag /* flag = 1 indicates the optimum clock phase is obtained.      */
)
{
 static int  start = 1, upper = 0;
double  new, diff;
static double  old;

*flag = 0;
++upper;
new = atan2(ch_out.imag, ch_out.real);

if ( start == 3) {
         diff = new - old;

         if ((diff > 0 && diff < PI)|| diff < -PI) {
                  start = 0;
                  *flag = 1;
                  upper = 0;
                  }

         else
                  old = new;
 }

 else if (start == 2) {
         diff = new - old;
         if (( diff < 0 && diff > -PI)|| diff > PI)
                  start = 3;
         old = new;
 }

 else {
         old = new;
         start = 2;
         upper = 0;
 }

 if (upper == 25 ) {
         *flag = 1;
         start = 0;
         upper = 0;
 }

 }
 /*------------------------------------------------ end of file ------------------------------------------------*/
```

```
/* ------------------------------------------------ random.h ------------------------------------------------*/
/* Functions for generating random number are found in the c-file named rand.c. NOTE THAT the
function initrandu must be invoked prior to calling the random number generating functions (randu or
randn).

randu - returns a non-negative and double-precision value uniformly distributed over the range of y values
such that 0.0 <= y < 1.0.

randn - returns a normally distributed double-precision value with unit variance and zero mean.

initrandu - read a table of coefficients from the data file named rcoeffs.dat.

closeru - overwrites the table in rcoeffs.dat after each call to the random number generators.

*/

/* These files were obtained from Anthony Soong and Peng Mok. */

#include<stdio.h>
#include<math.h>

extern double randn(long *);
extern double randu(long *);
extern void closeru(FILE *);
extern void initrandu(FILE *);


/*------------------------------------------------------------ end of file------------------------------------------------------------*/



/*-------------------------------------------------------------- rand.c --------------------------------------------------------------*/
/* The following files used to generate Gaussian distributed RV written by Anthony Soong and Peng Mok.
*/

#include "random.h"
long rcff[100];

/* closeru -- overwrites coeffs.dat */
void closeru( FILE *fp )
{
int i;          /* write out rcff[1]...rcff[98]              */

rewind( fp );

for(i = 0; i < 98; i++)
          fprintf(fp,"   %ld\n",rcff[i+1]);
          fprintf(fp,"\n");
}

/*------------------------------------------------------------*/


double randu(long *j)   /*  generate uniform random no's          */
          /* long *j;           requires external long rcff[1]...[100]   */
{                    /* Note: initrand() must be called first  */
```

```
int p, q, k;
long n;

p = 98;  q = 27;    /* polynomial parameters                  */
n = 2147483647;     /* n = 2**31 - 1, sets bits = '1'           */

*j = *j + 1;

if (*j > p)
        *j = 1;

k = *j + q;
if (k > p)
        k =  k - p;

rcff[*j] = rcff[*j] ^ rcff[k];
return(( (float) rcff[*j] ) / ( (float) n ));

}

/*-------------------------------------------------------------*/


/* FUNCTION RANDN - generates random normal deviates by the method of Kinderman  and Ramage.
Described in W.J. Kennedy & J.E. Gentle, "Statistical Computing" pp. 205-207.  */

double randn(long *pj)
{
static double   a = 2.216035867166471;
static double asq = 4.91081476211548;     /*  a * a              */
static double cst = 0.398942291736603;    /*  1/sqrt(2 * pi)      */

double u1, u2, u3, min, max, tempval, t;

double randu(long *);
double log(double);
double exp(double);
double sqrt(double);
double _fabs(double);

u1 = randu(pj);

if (u1 < 0.884070402298758) {
        u2 = randu(pj);
        return( a  * (1.13113163544180 * u1 + u2 - 1.0) );
}
else
        if (u1 < 0.973310954173898)
          if (u1 < 0.958720824790463)
          if (u1 < 0.911312780288703)
            {
              do
                 {
                 u2 = randu(pj);
                 u3 = randu(pj);
                  min = u2;
                  max = u3;
```

```
         if (u3 < u2)
            {
            min = u3;
            max = u2;
            }
         } while (max > 0.805577924423817);
      t = 0.47972740422241 -
            0.595507138015940 * min;
      if (u2 < u3)
         return(t);
      else
         return(-t);
      }
   else
      {
      do
         {
        u2 = randu(pj);
        u3 = randu(pj);
         min = u2;
         max = u3;
         if (u3 < u2)
            {
            min = u3;
            max = u2;
            }
         t = 0.479727404222441 + 1.105473661022070 * min;
         } while ( (max > 0.872834976671790) &&
             ((0.049264496373128 * fabs(u2 - u3)) >
             ( cst  * exp(-0.5 * t * t) -
              0.180025191068583 * ( a  - fabs(t)))) );
      if ( u2 < u3)
         return(t);
      else
         return(-t);
      }
else
   {
   do
      {
     u2 = randu(pj);
     u3 = randu(pj);
      min = u2;
      max = u3;
      if (u3 < u2)
         {
         min = u3;
         max = u2;
         }
      t =  a  - 0.630834801921960 * min;
      } while ( (max > 0.755591531667601) &&
          ((0.034240503750111 * fabs(u2 - u3)) >
          ( cst  * exp(-0.5 * t * t) -
           0.180025191068563 * ( a  - fabs(t)))) );
   if (u2 < u3)
      return(t);
   else
      return(-t);
```

```
         }
      else
         {
         do
            {
            u2 = randu(pj);
            u3 = randu(pj);
            tempval =  asq  - 2.0 * log(u3);
            } while ( (u2 * u2) >= ( asq  *
                 1.0 / tempval) );
         if (u1 < 0.986655477086949)
            return(sqrt(tempval));
         else
            return(-sqrt(tempval) );
         }
      }

/*------------------------------------------ end of file ------------------------------------------*/
```

```
/*------------------------------------------- rand.dat -------------------------------------------*/
/* These are the data used by rand.c to generate random Gaussian variable. Read in the data column-wise.  */
```

1327272865	1881161636	1745095944	1673561186	1673561186
53675950	1167716718	27665504	802041206	802041206
247022513	1495355194	1652510585	663347946	663347946
1699311560	2103363402	1698011084	72636422	72636422
1231082430	1814346850	106991299	1172903330	1172903330
1771860397	2129578319	1446446012	1802904343	1802904343
385689608	1047444040	239315414	1692341095	1692341095
1946258148	1896574377	2055163764	11799787	11799787
889248243	1133611041	1837385924	793872728	793872728
525067715	1127125120	1926001396	426305383	426305383
1341700781	1250300693	470759001	519179293	519179293
310408603	1301374977	452893118	1601120191	1601120191
709545553	1662784463	1949920392	1577582331	1577582331
625343442	1402014885	238825068	1491950165	1491950165
1688247805	1640948365	2133527825	2016705741	2016705741
1091585658	672281885	1608031693	1027028368	1027028368
494575369	2612803	551940275	84789720	84789720
2058861524	1329344259	477472265	202178719	202178719
1076256944	409203979	449464566	1941036745	1941036745
349083705	104788258	862404973	1158594586	1158594586

```
/*------------------------------------------ end of file ------------------------------------------*/
```

```
/*---------------------------------------------- rx_filter.c ----------------------------------------------*/
/* Receiver Filter For 1-bit differential Detection of GMSK.          */
/* The filter is modelled with the assumption that the gaussian band-pass filter causes negligible phase-
distortion.                     */

#include "spec.h"
#define n_taps 51 /* number of taps of a transversal gaussian filter. */

double  Br = 1.375e6; /* 3 dB bandwidth of the receiver filter. Br is determined from the equation Br*T =
                           1.1, where T is the bit duration, which is 800ns in this case. */

int  taps[n_taps];

void
rx_filter(const cmplx_t in_samp,  /* input sample.                 */
          cmplx_t *filter_out  /* output of the filter.        */
)
{
int i;

static int  start = 1;

double fil_re, fil_im;

static double    h[n_taps], /* Tap-weights of the FIR Gaussian filter. */
                  buf_re[n_taps], /* buffer for storing input samples at various taps.     */
                  buf_im[n_taps];

fil_re = 0.0;
fil_im = 0.0;

/* Initialize the buffer of the FIR filter and define the filter tap-weight.*/

if (start == 1) {
          for (i = 0; i < n_taps; ++i) {
                    buf_re[i] = 0;
                    buf_im[i] = 0;
                    taps[i] = i - (n_taps-1)/2;
                    h[i] = exp(-2*square(PI*Br/2*Ts*taps[i])/log(2));
          }
          start = 0;
}

/* Shift the contents of the buffer when receive a new sample.       */

for (i = 1; i < n_taps; ++i) {
          buf_re[n_taps-i] = buf_re[n_taps-1-i];
          buf_im[n_taps-i] = buf_im[n_taps-1-i];
}

buf_re[0] = in_samp.real;
buf_im[0] = in_samp.imag;

/* Compute the filter output.                          */

for (i = 0; i < n_taps; ++i) {
          fil_re += h[i]*buf_re[i];
```

```
        fil_im += h[i]*buf_im[i];
}

filter_out->real = fil_re;
filter_out->imag = fil_im;

}


/*------------------------------------------------- end of file ------------------------------------------*/



/*-------------------------------------------------- detector.c -----------------------------------------*/
/* 1-bit Differential Detector.                              */
/* The decision rule is based on testing the value of sine of a phase difference of a complex signal over 1-bit
interval duration against a fixed threshold of zero. If the sine of a phase difference is greater than the
threshold value then, a "1" bit is detected else, "-1" bit is detected.    */


#include "spec.h"

void
detect(const cmplx_t  sig_in,  /* output of a receiver filter.        */
        int *detect_bit   /* detected bit.                          */
        )
{
double  ph_new,
        delta_ph;  /*phase difference between sample signals which are one-bit interval apart.    */
static double  ph_old = 0;

ph_new = atan2(sig_in.imag, sig_in.real);
/* atan2() returns the arc tangent of y/x in the range [-pi, +pi] radians. It uses the signs of both y and x to
determine the quadrant of the return value.                          */

delta_ph = ph_new - ph_old;

*detect_bit = sin(delta_ph) > 0 ? 1 : -1;

ph_old = ph_new;

}

/*------------------------------------------------- end of file ------------------------------------------*/



/*-------------------------------------------------- spec.h ----------------------------------------------*/
/* Header file for basic system specification.  */

        #include<stdio.h>
        #include<stdlib.h>
        #include<math.h>
```

```
       #define spread 300 /* rms delay spread of the multipath channel in ns. Choose 100, 150, 200, 250
                             or 300  */
       #define SNR  30.0   /* signal to noise ratio in dB               */
       #define MAX 40000  /* number of frames used in the simulation.    */


       #define BIT_PER_SLOT  438  /* 424 data bits + 14 "artificial guard bits".
       #define SAMP_PER_BIT  16  /* oversampling factor  */
       #define Ts  50e-9  /* sampling interval (800e-9 is the bit interval).     */
       #define PI  3.141592653589793
       #define square(x)  (x)*(x)

    typedef struct cmplx_s {
         double  real, imag;
     } cmplx_t;

/*------------------------------------------ end of file ------------------------------------------*/
```

APPENDIX B

C-programs for *Schemes I, II, III* and *IV*. Only functions which are not included in Appendix A are presented. They include:

i. SchemeI.c (the main program for the system employing *Scheme I*)
ii. SchemeII.c (the main program for the system employing *Scheme II*)
iii SchemeIII.c (the main program for the system employing *Scheme III*)
iv SchemeIV.c (the main program for the system employing *Scheme IV*)
v source2.c (slot-generator; unlike source.c in Appendix, it also generates training sequence of MSK signal)
vi 7-path_ch.c (7-path Rayleigh fading channel with correlated paths)
vii sync.c (function for synchronizing receiver to the dominant path)
viii (6,2)DFE.c (DFE for *Schemes I, II* and *III*)
ix comb_div_dfe.c (combined diversity-equalizer function for *Scheme IV*)
x two-branch_detect.c (one-bit differential detector for *Schemes II* and *III*. *Schemes I* and *IV* use detector.c presented in Appendix A.)
xi two-branch_rx_filter.c (receiver filter for *Schemes II, III* and *IV*. *Scheme I* uses rx_filter.c given in Appendix A.)

```
/*------------------------------------------------ SchemeI.c ------------------------------------------------*/

/* This is the system implemented with Scheme I. The selection diversity scheme works in such a way that
the branch with a stronger signal measured in the preceeding time slot (frame) is selected for reception in
the current time slot. The receiver is synchronized to the transmitter through the dominant path of a
multipat channel.  */

#include "spec.h"
#include "random.h"

/*  FUNCTION PROTOTYPES          */

      void data_source(int*, cmplx_t*);
      void gaussian_filter(const int, double*,const int*);
      void fm_mod(const double, cmplx_t*, const int*);
      void channel(const cmplx_t, const int, cmplx_t*, cmplx_t*, int*);
      void sync(const cmplx_t, int*);
      void rx_filter(const cmplx_t, cmplx_t*);
      void dfe(cmplx_t*, cmplx_t*, int*, cmplx_t*);
      void detect(const cmplx_t, int*);
      void initrandu(FILE*);
      double randn(long*);
      void closeru(FILE*);

main ()
{
int  iter, i, j, d, nbit,
         slot[BIT_PER_SLOT],  /* Data slot */
```

```
        skip = 176640, /*'skip' = 23 packets x 16 samples per bit x 480 bits per packet. 'skip' is the
                          amount of 'time' that the channel is 'fast-forward' after each packet is transmitted.
                          Each actual DECT packet contains 424 data bits (include sync word, control, CRC
                          code etc) and guard band, which is equivalent to 56-bit interval. */
        time_rec = 0,   /* counter set for synchronization mechanism.        */
        samp_count = 0, /* counter set for the sampler at the receiver.      */
        flag,
        select,     /* indicator for the branch selection.                */
        delay = 2000, /* excess delay introduced by the receiver filter with respect to the instant when
                          timing is recovered. The initial value of 'delay' is set to a large value so as to
                          inhibit the action of the sampler before the proper clock-phase is recovered.  */
        new_frame,
        x,
        detect_bit; /* detected bit.                                      */

long rseed = 10;

double  fil_out, /* pre-modulation filter output.                       */
        ch_nvolt = 4*sqrt(0.5/pow(10,(double)SNR/10.0)),  /*  Voltage of AWGN before bandpass
                            filtering. 0.5 is the average signal power. Factor 4 is introduced to give an
                            appropriate SNR at the receiver filter output; it is squared root of the
                            oversampling factor, which is 16. */
        pow = 0, /* long term average complex signal power at one of the branches.*/
        pow_count = 0,
        pow1 = 1,  /* short term average signal power at Branch#1.      */
        pow2 = 0,  /* short term average signal power at Branch#2.      */
        error = 0,  /* number of bit errors.                             */
        error_count = 0; /* number of bits processed.   */

cmplx_t  mod_out, /* modulator output.                                    */
        ch_out[2],  /* channel output.                                   */
        sync_path, /* undistorted signal from the channel used for synchronizing the receiver to the
                            transmitter.     */
        branch_sel,  /* selected branch.                                 */
        rx_out,  /* receiver filter output.                               */
        samples[2],/* signal samples fed into the DFE (2 samples per bit). */
        train_seq[16],  /* training sequence for DFE (MSK signal).       */
        dfe_out;  /* output of DFE.                                      */

FILE *fp,   /*  pointer to a data file named rand.dat  */
        *datf;
fp = fopen("rand.dat","r+");

initrandu(fp);  /*  this function is found in the file named rand.c and it must be invoked number generator
                    called randn, which is meant to to produce zero mean unit variance random number. The
                    function initrandu() write data from a file called rand.dat. These data are used to generate
                    random numbers.   */

for (iter = 0; iter < MAX; ++iter) {
        x=85;  /* instant when synchronization starts.                  */
        time_rec = 0;
        new_frame = 1;
        i = 29 ;  /* start checking from the 30th bit of every packet. Skip the preamble word (12 bits in
                      this case), the 16-bit synchronization word which is used to train the DFE and the
                      first data bit.          */
        samp_count = 0;
        delay = 20000000;  /* inhibit rx-sampler till clock phase is recovered.  */
```

```
/* Diversity scheme based on choosing the branch with a larger average power measured in a previous
      time slot.            */
      if (pow1 > pow2)
        select = 1;

      else
        select = 2;

      pow1 = pow2 = 0;

     data_source(slot, train_seq);

      for (nbit = 0; nbit < BIT_PER_SLOT; ++nbit)  { /* Each bit is sampled 16 times. */
         for (j = 0; j < 16; ++j) {
           gaussian_filter(slot[nbit], &fil_out, &new_frame);
           fm_mod(fil_out, &mod_out, &new_frame);
           channel(mod_out, skip, ch_out, &sync_path, &new_frame);
          pow += ch_out[1].real*ch_out[1].real + ch_out[1].imag*ch_out[1].imag;
            ++pow_count;

          if (nbit > 300  && nbit < 410) {
           pow1 += ch_out[0].real*ch_out[0].real + ch_out[0].imag*ch_out[0].imag;
           pow2 += ch_out[1].real*ch_out[1].real + ch_out[1].imag*ch_out[1].imag;
           }

           ++time_rec;
           ++samp_count;

          if (time_rec > x) {
              sync(sync_path, &flag);

              if (flag == 1) {
               delay = 165;  /* this includes an excess delay of 25 samples introduced by the receiver
                             filter and also the skipping of the rest of the preamble bits for synchronization.
                             */
                 x = 9000000;  /* inhibit clock_recovery mechanism.          */
                 time_rec = 0; /* reset time recovery counter.          */
                 d = 0;
               }
              samp_count = 1;  /* reset counter for sampler.            */

              }

          /* Antenna selection.                           */
            switch (select) {
                case 1:

                   branch_sel.real = ch_out[0].real + ch_nvolt*randn(&rseed);
                   branch_sel.imag = ch_out[0].imag + ch_nvolt*randn(&rseed);
                     break;

                case 2:

                   branch_sel.real = ch_out[1].real + ch_nvolt*randn(&rseed);
                   branch_sel.imag = ch_out[1].imag + ch_nvolt*randn(&rseed);
                     break;

                default:
```

```
                fprintf(datf,"error occurs in selection scheme");
                 break;
            }

           rx_filter( branch_sel, &rx_out);

       /*  Sample, equalize and demodulate the received signal. Samples are
            taken at T/2 interval.                                    */
           if (samp_count == delay) {
              samples[d].real = rx_out.real;
              samples[d].imag = rx_out.imag;
              delay += 8;
               ++d;

              if (d == 2) {
                d = 0;
                dfe(samples, train_seq, &flag, &dfe_out);
                detect(dfe_out, &detect_bit);

     /* Note that 'flag' is incremented by 1 before exiting 'dfe()'. So, start detecting the second data bit
         at 'flag' = 21, instead of at 'flag' = 20. That is, the 2nd data bit starts with flag = 2-bit delay (by
         DFE) + 16 training bits + 2 data bits + one extra increment at the dfe output.          */

                 if (flag > 20) {
                  if (detect_bit != slot[i])
                      ++error;
                   ++i;
                  ++error_count;

                 }
               }
             }
          }

        }
    }

datf = fopen("sd_corr.d","w");

fprintf(datf, "%d\t %0.0f\t %0.0f\t %f\t %f\n", select,
             error_count, error, error/error_count, pow/pow_count);

closeru(fp);    /* overwrite the data file rand.dat with new values.      */
fclose(fp);
fclose(datf);
}

/*------------------------------------------------ end of file ----------------------------------------------------*/


/*---------------------------------------------------- SchemeII.c ------------------------------------------------------*/
/* This is the system implemented with Scheme II. The selection scheme is driven by ERRI criterion.
Practically, CRC code is used to check for errors in the receiving slot. In the simulation, CRC code is not
implemented but the occurence of errors can still be detected since transmitted bits are known. The selection
scheme works in such a way that when there are errors in both branches, or when there is no error in either
branche, the branch selected in the previous frame continues to be used; when there is 1 or more errors in
one branch and no error in the other branch, the branch with no error is selected. */
```

```
#include "spec.h"
#include "random.h"

/*  FUNCTION PROTOTYPES      */

void data_source(int*, cmplx_t*);
void gaussian_filter(const int, double*,const int*);
void fm_mod(const double, cmplx_t*, const int*);
void channel(const cmplx_t, const int, cmplx_t*, cmplx_t*, int*);
void sync(const cmplx_t, int*);
void two_rx_filter(const cmplx_t*, cmplx_t*);
void dfe(cmplx_t*, cmplx_t*, int*, cmplx_t*);
void dfe1(cmplx_t*, cmplx_t*, const int*, cmplx_t*);
void detect(const cmplx_t*, int*);
void initrandu(FILE*);
double randn(long*);
void closeru(FILE*);

main ()
{
int  iter, i, j, d, nbit,
          slot[BIT_PER_SLOT],  /* Data slot */
          skip = 176640, /*'skip' = 23 packets x 16 samples per bit x 480 bits per packet. 'skip' is the
                         amount of 'time' that the channel is 'fast-forward' after each packet is transmitted.
                         Each actual DECT packet contains 424 data bits (include sync word, control, CRC
                         code etc) and guard band, which is equivalent to 56-bit interval.  */
          time_rec = 0,   /* counter set for synchronization mechanism.          */
          samp_count = 0,  /* counter set for the sampler at the receiver.        */
          flag,
          select,      /* indicator for the branch selection.                 */
          delay = 2000, /* excess delay introduced by the receiver filter with respect to the instant when
                         timing is recovered. The initial value of 'delay' is set to a large value so as to
                         inhibit the action of the sampler before the proper clock-phase is recovered.  */
          new_frame,
          x,
          detect_bit[2],  /* detected bit.     */
          erri1 = 0, erri2 = 1,/* error rate indicators for Branch#1 and Branch#2.*/
          er_p_slt1 = 0, er_p_slt2 = 0; /* number of errors per slot.             */

 long rseed = 10;

 double  fil_out,  /* pre-modulation filter output.                    */
          ch_nvolt = 4*sqrt(0.5/pow(10,(double)SNR/10.0)),  /*  Voltage of AWGN before bandpass
                           filtering. 0.5 is the average signal power. Factor 4 is introduced to give an
                           appropriate SNR at the receiver filter output; it is squared root of the
                           oversampling factor, which is 16. */
          pow = 0, /* long term average complex signal power at one of the branches. */
          pow_count = 0,
          error = 0,  /* number of bit error detected in the system.       */
          error_count = 0;  /* number of bits processed.    */

 cmplx_t  mod_out, /* modulator output.              */
          ch_out[2],  /* channel output.          */
          sync_path, /* undistorted signal from the channel used for synchronizing the receiver to the
                        transmitter.              */
          rx_out[2],  /* receiver filter output (2 branches).            */
          samples1[2], /* signal samples fed into DFE#1 (2 samples per bit).  */
```

```
         samples2[2], /* signal samples fed into DFE#2 (2 samples per bit).  */
         train_seq[16],  /* training sequence for DFE (MSK signal).          */
         dfe_out[2];  /* outputs of DFEs.                                    */

FILE *fp,   /*  pointer to a data file named rand.dat  */
         *datf;
fp = fopen("rand.dat","r+");

initrandu(fp);  /*  this function is found in the file named rand.c and it must be invoked number generator
                    called randn, which is meant to to produce zero mean unit variance random number. The
                    function initrandu() write data from a file called rand.dat. These data are used to generate
                    random numbers.   */

for (iter = 0; iter < MAX; ++iter) {
         x=85;  /* instant when synchronization starts.                   */
         time_rec = 0;
         new_frame = 1;
         i = 29 ;  /* start checking from the 30th bit of every packet. Skip the preamble word (12 bits in
                      this case), the 16-bit synchronization word which is used to train the DFE and the first
                      data bit.              */
         samp_count = 0;
         delay = 20000000; /* inhibit rx-samplers till respective clock  phases are recovered.  */

         /* Selection diversity based on ERRI criterion.          */
         if (erri1 == 0 && erri2 == 1)
                  select = 1;

         else if (erri1 == 1 && erri2 == 0)
                  select = 2;

         if (select == 1) {
                  error += er_p_slt1;
         }

         else {
                  error += er_p_slt2;

         erri1  = erri2 = er_p_slt1  = er_p_slt2 = 0;

         data_source(slot, train_seq);

         for (nbit = 0; nbit < BIT_PER_SLOT; ++nbit)  { /* Each bit is sampled 16 times.*/
           for (j = 0; j < 16; ++j) {
             gaussian_filter(slot[nbit], &fil_out, &new_frame);
             fm_mod(fil_out, &mod_out, &new_frame);
             channel(mod_out, skip, ch_out, &sync_path, &new_frame);
             pow += ch_out[1].real*ch_out[1].real + ch_out[1].imag*ch_out[1].imag;
              ++pow_count;

              ++time_rec;
              ++samp_count;

             if (time_rec > x) {
               sync(sync_path, &flag);

                if (flag == 1) {
```

```
        delay = 165;  /* this includes an excess delay of 25 samples introduced by the receiver
                               filter and also the skipping of the rest of the preamble bits used to for
                               synchronization.          */

          x = 9000000;  /* inhibit clock_recovery mechanism.           */
          time_rec = 0; /* reset time recovery counter.             */
          d = 0;
        }
       samp_count = 1;  /* reset counter for sampler.               */

      }

  /* Add white Gaussian noise to each branch. */
      ch_out[0].real += ch_nvolt*randn(&rseed);
      ch_out[0].imag += ch_nvolt*randn(&rseed);
      ch_out[1].real += ch_nvolt*randn(&rseed);
      ch_out[1].imag += ch_nvolt*randn(&rseed);

       two_rx_filter(ch_out, rx_out);

  /*  Sample, equalize and demodulate the received signal. Samples are
       taken at T/2 interval.                                   */
      if (samp_count == delay) {
         samples1[d].real = rx_out[0].real;
          samples1[d].imag = rx_out[0].imag;
         samples2[d].real = rx_out[1].real;
          samples2[d].imag = rx_out[1].imag;
          delay += 8;
          ++d;

       if (d == 2) {
        d = 0;
       dfe1(samples1, train_seq, &flag, &dfe_out[0]); /*DFE for Branch#1 */
       dfe(samples2, train_seq, &flag, &dfe_out[1]);  /*DFE for Branch#2 */
       detect(dfe_out, detect_bit);

 /*  Note that 'flag' is incremented by 1 before exiting 'dfe()'. So, start detecting the second data bit
       at 'flag' = 21, instead of at 'flag' = 20. That is, the 2nd data bit starts with flag = 2-bit delay
       (by DFE) + 16 training bits + 2 data bits + one extra increment at the dfe output.
       */

          if (flag > 20) {
           if (detect_bit[0] != slot[i]) {
               ++er_p_slt1;

             if (flag < 409)  /* error checking in Branch I.  */
                erri1 = 1;
             }

             if (detect_bit[1] != slot[i]) {
                 ++er_p_slt2;

                if (flag < 409) /* error checking in Branch II.   */
                    erri2 = 1;
             }

              ++error_count;
              ++i;
        }

if (erri1 == 0 && erri2 == 1)
        elect = 1;

else if (erri1 == 1 && erri2 == 0)
        select = 2;

if (select == 1)
        error += er_p_slt1;

else
        error += er_p_slt2;

datf = fopen("ers_uncorr.d","w");

fprintf(datf, " %0.0f\t %0.0f\t %f\t %f\n",
        error_count, error, error/error_count, pow/pow_count);

closeru(fp);    /* overwrite the data file rand.dat with new values.      */
fclose(fp);
fclose(datf);
}

/*------------------------------------------------------ end of file ------------------------------------------------------*/


/*------------------------------------------------------ SchemeIII.c ------------------------------------------------------*/
/* This is the system implemented with Scheme III. This file is very similar to SchemeII.c. Repeated
documentation may be avoided. */

#include "spec.h"
#include "random.h"

/*  FUNCTION PROTOTYPES                         */

      void data_source(int*, cmplx_t*);
      void gaussian_filter(const int, double*,const int*);
      void fm_mod(const double, cmplx_t*, const int*);
      void channel(const cmplx_t, const int, cmplx_t*, cmplx_t*, int*);
      void sync(const cmplx_t, int*);
      void two_rx_filter(const cmplx_t*, cmplx_t*);
      void dfe(cmplx_t*, cmplx_t*, int*, cmplx_t*, double*);
      void dfe1(cmplx_t*, cmplx_t*, const int*, cmplx_t*, double*);
      void detect(const cmplx_t*, int*);
      void initrandu(FILE*);
      double randn(long*);
      void closeru(FILE*);

main ()
```

```
{
int  iter, i, j, d, nbit,
          slot[BIT_PER_SLOT],
          skip = 176640,
          time_rec = 0,
          samp_count = 0,
          flag,
          select,
          delay = 2000,
          new_frame,
          x,
          detect_bit[2],  /* detected bit.     */
          er_p_slt1 = 0, er_p_slt2 = 0; /* number of errors per slot.              */

long rseed = 10;

double  fil_out,  /* pre-modulation filter output.                         */
          ch_nvolt = 4*sqrt(0.5/pow(10,(double)SNR/10.0)),
          pow = 0,
          pow_count = 0,
          error = 0,  /* number of bit error detected in the system.         */
          dfe_er[2] = {0,0},
          er_pow1  = 1, er_pow2 = 0,  /* square of error signals from the DFEs. */
          error_count = 0;  /* number of bits processed.    */

cmplx_t  mod_out, /* modulator output.               */
          ch_out[2],  /* channel output.            */
          sync_path, /* undistorted signal from the channel used for synchronizing the receiver to the
                            transmitter.              */
          rx_out[2],  /* receiver filter output (2 branches).             */
          samples1[2], /* signal samples fed into DFE#1 (2 samples per bit).  */
          samples2[2], /* signal samples fed into DFE#2 (2 samples per bit).  */
          train_seq[16],  /* training sequence for DFE (MSK signal).        */
          dfe_out[2];  /* outputs of DFEs.                          */

FILE *fp,   /*  pointer to a data file named rand.dat  */
          *datf;
fp = fopen("rand.dat","r+");

initrandu(fp);

for (iter = 0; iter < MAX; ++iter) {
          x=85;  /* instant when synchronization starts.                    */
          time_rec = 0;
          new_frame = 1;
          i = 29 ;
          samp_count = 0;
          delay = 20000000; /* inhibit rx-samplers till respective clock  phases are recovered.        */

        /* Selection diversity based on MSE criterion.                    */
          if (er_pow1 > er_pow2)
             select = 2;

          else
             select = 1;

          er_pow1 = er_pow2 = 0.0;
```

```
if (select == 1) {
   error += er_p_slt1;
    }

 else {
   error += er_p_slt2;
    }

 er_p_slt1 = er_p_slt2 = 0;

data_source(slot, train_seq);

 for (nbit = 0; nbit < BIT_PER_SLOT; ++nbit)  {
   for (j = 0; j < 16; ++j) {
     gaussian_filter(slot[nbit], &fil_out, &new_frame);
     fm_mod(fil_out, &mod_out, &new_frame);
     channel(mod_out, skip, ch_out, &sync_path, &new_frame);
     pow += ch_out[1].real*ch_out[1].real + ch_out[1].imag*ch_out[1].imag;
      ++pow_count;

      ++time_rec;
      ++samp_count;

     if (time_rec > x) {
        sync(sync_path, &flag);

       if (flag == 1) {
          delay = 165;
          x = 9000000;  /* inhibit clock_recovery mechanism.          */
          time_rec = 0; /* reset time recovery counter.            */
          d = 0;
        }
        samp_count = 1;  /* reset counter for sampler.              */

      }

 /* Add white Gaussian noise to each branch. */
     ch_out[0].real += ch_nvolt*randn(&rseed);
     ch_out[0].imag += ch_nvolt*randn(&rseed);
     ch_out[1].real += ch_nvolt*randn(&rseed);
     ch_out[1].imag += ch_nvolt*randn(&rseed);

      two_rx_filter(ch_out, rx_out);

/*  Sample, equalize and demodulate the received signal. Samples are
      taken at T/2 interval.                               */
     if (samp_count == delay) {
        samples1[d].real = rx_out[0].real;
         samples1[d].imag = rx_out[0].imag;
        samples2[d].real = rx_out[1].real;
         samples2[d].imag = rx_out[1].imag;
         delay += 8;
          ++d;

     if (d == 2) {
       d = 0;
     dfe1(samples1, train_seq, &flag, &dfe_out[0], &dfe_er[0]); /*DFE for Branch#1 */
     dfe(samples2, train_seq, &flag, &dfe_out[1], &dfe_er[1]);  /*DFE for Branch#2 */
```

```
            detect(dfe_out, detect_bit);
            er_pow1 += dfe_er[0];
            er_pow2 += dfe_er[1];


              if (flag > 20) {
               if (detect_bit[0] != slot[i]) {
                  ++er_p_slt1;
                      }

              if (detect_bit[1] != slot[i]) {
                  ++er_p_slt2;

                      }

               ++error_count;
               ++i;

               }
              }
             }
           }

         }
}

if (er_pow1 > er_pow2)
        select = 2;

else
        select = 1;

if (select == 1)
        error += er_p_slt1;

else
        error += er_p_slt2;

datf = fopen("dfesel25.d","w");
fprintf(datf, " %0.0f\t %0.0f\t %f\t %f\n",
          error_count, error, error/error_count, pow/pow_count);

closeru(fp);    /* overwrite the data file rand.dat with new values.       */
fclose(fp);
fclose(datf);
}

/*------------------------------------------------ end of file ------------------------------------------------*/


/*------------------------------------------------ SchemeIV.c ------------------------------------------------*/
/* This is the system implemented with Scheme IV.         */

#include "spec.h"
#include "random.h"

/*  FUNCTION PROTOTYPES                                     */
```

```
        void data_source(int*, cmplx_t*);
        void gaussian_filter(const int, double*, const int*);
        void fm_mod(const double, cmplx_t*, const int*);
        void channel(const cmplx_t, const int, cmplx_t*, cmplx_t*, int* );
        void sync(const cmplx_t, int*);
        void two_rx_filter(const cmplx_t*, cmplx_t*);
        void comb_div_dfe(cmplx_t(*)[], cmplx_t*, int*, cmplx_t*);
        void detect(const cmplx_t, int*);
        void initrandu(FILE*);
        double randn(long*);
        void closeru(FILE*);

main ()
{
int  iter, i, j, d, nbit,
        slot[BIT_PER_SLOT],  /* Data slot */
        skip = 176640, /*'skip' = 23 packets x 16 samples per bit x 480 bits per packet. 'skip' is the
                        amount of 'time' that the channel is 'fast-forward' after each packet is transmitted.
                        Each actual DECT packet contains 424 data bits (include sync word, control, CRC
                        code etc) and guard band, which is equivalent to 56-bit interval.  */
        time_rec = 0,   /* counter set for synchronization mechanism.        */
        samp_count = 0,  /* counter set for the sampler at the receiver.      */
        flag,
        select,     /* indicator for the branch selection.                */
        delay = 2000, /* excess delay introduced by the receiver filter with respect to the instant when
                        timing is recovered. The initial value of 'delay' is set to a large value so as to
                        inhibit the action of the sampler before the proper clock-phase is recovered.  */
        new_frame,
        x,
        detect_bit;  /* detected bit.                                 */

long rseed = 10;

double  fil_out,  /* pre-modulation filter output.                 */
        ch_nvolt = 4*sqrt(0.5/pow(10,(double)SNR/10.0)),  /*  Voltage of AWGN before bandpass
                          filtering. 0.5 is the average signal power. Factor 4 is introduced to give an
                          appropriate SNR at the receiver filter output; it is squared root of the
                          oversampling factor, which is 16. */
        pow = 0, /* long term average complex signal power at one of the branches.*/
        pow_count = 0,
        error = 0,  /* number of bit errors.                       */
        error_count = 0;  /* number of bits processed.   */

cmplx_t  mod_out, /* modulator output.                            */
        ch_out[2],  /* channel output.                          */
        sync_path, /* undistorted signal from the channel used for synchronizing the receiver to the
                        transmitter.      */
        rx_out[2],  /* receiver output.                          */
        samples[2][2],/* signal samples fed into the two FFFs (2 samples per bit).*/
        train_seq[16],  /* training sequence for DFE (MSK signal).        */
        dfe_out;  /* output of DFE.                              */

FILE *fp,   /*  pointer to a data file named rand.dat  */
        *datf;
fp = fopen("rand.dat","r+");
```

```
initrandu(fp);  /*  this function is found in the file named rand.c and it must be invoked number generator
                    called randn, which is meant to to produce zero mean unit variance random number. The
                    function initrandu() write data from a file called rand.dat. These data are used to generate
                    random numbers.    */

for (iter = 0; iter < MAX; ++iter) {
        x=85;  /* instant when synchronization starts.                    */
        time_rec = 0;
        new_frame = 1;
        i = 29 ;  /* start checking from the 30th bit of every packet. Skip the preamble word (12 bits in
                     this case), the 16-bit synchronization word which is used to train the DFE and the
                     first data bit.            */
        samp_count = 0;
        delay = 20000000;  /* inhibit rx-sampler till clock phase is recovered.  */

       data_source(slot, train_seq);

        for (nbit = 0; nbit < BIT_PER_SLOT; ++nbit)  {
          for (j = 0; j < 16; ++j) {
            gaussian_filter(slot[nbit], &fil_out, &new_frame);
            fm_mod(fil_out, &mod_out, &new_frame);
            channel(mod_out, skip, ch_out, &sync_path, &new_frame);
            pow += ch_out[1].real*ch_out[1].real + ch_out[1].imag*ch_out[1].imag;
             ++pow_count;
             ++time_rec;
             ++samp_count;

            if (time_rec > x) {
              sync(sync_path, &flag);

              if (flag == 1) {

              delay = 165;  /* this includes an excess delay of 25 samples introduced by the receiver
                               filter and also the skipping of the rest of the preamble bits for synchronization.
                               */
                x = 9000000;   /* inhibit clock_recovery mechanism.          */
                time_rec = 0; /* reset time recovery counter.              */
                d = 0;
              }
              samp_count =  1;  /* reset counter for sampler.               */                 }

            ch_out[0].real += ch_nvolt*randn(&rseed);
            ch_out[0].imag += ch_nvolt*randn(&rseed);
            ch_out[1].real += ch_nvolt*randn(&rseed);
            ch_out[1].imag += ch_nvolt*randn(&rseed);

             two_rx_filter(ch_out, rx_out);

        /*  Sample, equalize and demodulate the received signal. Samples are taken
             at T/2 interval.                                            */
            if (samp_count == delay) {
              samples[d][0].real = rx_out[0].real;
               samples[d][0].imag = rx_out[0].imag;
              samples[d][1].real = rx_out[1].real;
               samples[d][1].imag = rx_out[1].imag;
                delay += 8;
                 ++d;
            if (d == 2) {
              d = 0;
             comb_div_dfe(samples, train_seq, &flag, &dfe_out);
             detect(dfe_out, &detect_bit);

      /* Note that 'flag' is incremented by 1 before exiting 'dfe()'. So, start detecting the second data bit
          at 'flag' = 20, instead of at 'flag' = 19. That is, the 2nd data bit starts with flag = 1-bit delay (by
          DFE) + 16 training bits + 2 data bits + one extra increment at the dfe output.            */

                if (flag > 19) {
                  if (detect_bit != slot[i])
                      ++error;
                    ++i;
                   ++error_count;

                }
              }
            }
          }

        }
      }

     datf = fopen("sy_uncorr.d","w");
      fprintf(datf, "%d\t%0.0f\t %0.0f\t %0.8f\t %f\n", iter,
           error_count,error,error/error_count, pow/pow_count);

         closeru(fp);     /* overwrite the data file rand.dat with new values.        */
      fclose(fp);
     fclose(datf);
         }

/*---------------------------------------------------- end file -----------------------------------------------------*/


/*---------------------------------------------------- source2.c ----------------------------------------------------*/
(Similar to the one shown in APPENDIX A except that "timing bits" are changed for convenience, and
training sequence in terms of MSK signal sample is also generated).
/* The code generates a single time slot. It is not the exact DECT slot format; the actual timing sequence is
not used. However, the same packet-synchronization bit sequence is used, which in this case is used for
equalizer training.        */

#include "spec.h"
void
data_source(int slot[],
          cmplx_t train[]  /* training sequence for DFE.                */
          )
{
int i;
static int  start = 1;
double ph = 0;   /* MSK signal phase.                              */

if (start == 1)  {
          srand48(3);   /* initializing entry point for mrand48(). The seed is arbitrary chosen as '3'.  */
          start = 0;
}
```

```
/* Preamble word for timing approximations (not DECT format) */
slot[0] = slot[1] = slot[2] = 0;
slot[3] = -1;  slot[4] = 1;
slot[5] = slot[6] = slot[7]= 1;
slot[8] = slot[10] = -1;
slot[9] = slot[11] = 1;

/* DFE training bits (same as DECT packet-synchronization bit sequence)      */
slot[12]=slot[13]=slot[14]=slot[16]=slot[19]=slot[20]=slot[24]=slot[26] = -1;
slot[15]=slot[17]=slot[18]=slot[21]=slot[22]=slot[23]=slot[25]=slot[27] = 1;

/* Data bits (a few more data bits than the actual format).                 */
for (i = 12; i < 28; ++i) {
          if (slot[i] == 1)
                    ph += PI/2;
          else

          ph -= PI/2;
          train[i-12].real = cos(ph);
          train[i-12].imag = sin(ph);

 }

for (i = 28; i < BIT_PER_SLOT; ++i)  {
          if (mrand48() > 0)
                    slot[i] = 1;

          else
                    slot[i] = -1;
 }

 }
 /*-------------------------------------------------- end of file ---------------------------------------------------------------*/


 /*------------------------------------------ 7-path_ch.c (with correlated fading paths)-----------------------------------*/
 /* Two independent 7-path Rayleigh Fading Channels each with With an averaged power profile
 recommended by The Joint Technical Committe of Committee T1 T1P1.4 And  The TIA
 TR46.3.3/TR45.4.4 On Wireless Access for indoor commercial.  The fading paths have various degree of
 correlations. See descritption in the thesis.   */

 #include "spec.h"
 #include <malloc.h>

 #define   wm   2*PI*5    /* maximum Doppler shift in radian.                  */
 #define   N_oscil   26   /* N_oscil = number of oscillators  (per fader).    */
 #define   no_fader  14    /* number of independent faders.                 */
 #define   no_path   7     /* number of paths per branch (channel).           */
 #define nr  sqrt(N_oscil-1)
 #define ni  sqrt(N_oscil)
 double  oscil;
 double two_pi = 2*PI;
 void *calloc(size_t, size_t);
 /*.........................................................................................*/
 void
 rayleigh(int j,   /* designate fader for j-th path.                    */
        int skip, /* duration between successive transmissions from the same portable in terms of number
                   of samples (i.e. the amount of time (in sample unit) the channel is fast forward for each
                   update). */
        cmplx_t *fade  /* complex fading signals.                    */
        )
{

int  i, k;

static int start = 1;

static double  gain_re[N_oscil], gain_im[N_oscil-1], phase[no_fader][N_oscil-1],            wn[N_oscil-1],
wmt[no_fader], phase_inc[N_oscil-1], wmt_inc;
     if ( start == 1)  {
        for (i = 1; i < N_oscil; ++i) {
          gain_re[i-1] = 2*cos(PI*i/N_oscil);
          gain_im[i-1] = 2*sin(PI*i/N_oscil);
          wn[i-1] = wm*cos(2*PI*i/(4*N_oscil -2));
          for (k = 1; k <= no_path; ++k) {
             phase[k-1][i-1] = PI*(i+2*(k-1))/N_oscil+10*wn[i-1];
          }
          for (k = no_path+1; k <= 2*no_path; ++k ) {
             phase[k-1][i-1] = PI*(i+2*(k-1))/N_oscil+20*wn[i-1];

      /* note that this equation is slightly modified such that two sets of
        fader starts at different times to ensure independent diversity branches. */
          }

          phase_inc[i-1] = wn[i-1]*skip*Ts;
        }

        gain_re[N_oscil-1] = sqrt(2);

         for (k = 0; k < no_path; ++k)
           wmt[k] = 10*wm ;

         for (k = no_path; k < 2*no_path; ++k)
           wmt[k] = 20*wm ;

         wmt_inc = skip*Ts*wm;
         start = 0;
     }

     wmt[j-1] += wmt_inc;
     while (wmt[j-1] > two_pi)
           wmt[j-1] -= two_pi;

    fade->real = gain_re[N_oscil-1]*cos(wmt[j-1]);
     fade->imag = 0.0;

     for (i = 1; i < N_oscil; ++i) {
        phase[j-1][i-1] += phase_inc[i-1];
         while (phase[j-1][i-1] > two_pi)
               phase[j-1][i-1] -= two_pi;

        oscil = cos(phase[j-1][i-1]);
        fade->real += oscil*gain_re[i-1];
         fade->imag += oscil*gain_im[i-1];
        }
       fade->real = fade->real/nr;
       fade->imag = fade->imag/ni;

}

/*-............................................................................*/
/* Time Dispersive (frequency selective) Radio Channels. Each of the two independent branches consists of
7 independent Rayleigh fading paths. See thesis for the power delay profile.   */

void
channel(const cmplx_t signal_in,  /* input complex baseband signal.          */
          const int skip,  /* update parameter.        */
          cmplx_t sig_out_2b[],  /* This is an array carrying complex signals from two independent
                                   branches (channels).        */
          cmplx_t *sync_path,/* undistorted signal from the average dominant path.         */
          int *new_frame
          )
{
int   i,j;

static int  start = 1,
          nfad1[no_path] = {1,2,3,4,5,6,7}, /*index for faders in Branch#1  */
          nfad2[no_path] = {8,9,10,11,12,13,14}, /*index for faders in Branch#2 */
          N, delay[no_path];

double  *corr;

static double  *reg_real,  /* pointer to a buffer for holding tap-inputs (in-phase component).  */
          *reg_imag,  /* pointer to a buffer for holding tap-inputs  (quadrature component).  */
          ch_gain,     /* approximated channel gain.                  */
          attn[no_path],  /* attenuation of the 14 rays.              */
          fad_re[2][no_path],/* fad_re and fad_im are real and          */
          fad_im[2][no_path];

cmplx_t  fader, attn_sig;

if (start == 1) {

          corr = (double *)malloc((no_path-1)*sizeof(double));

 /* There are correlations between real components of different faders as well as between the imaginary
components. Time averaged correlations of the real components of faders i and j are <xi xj>, whereas those
of the imaginary  components are <yi, yj>. These values are determined empirically.               */
            corr[0] = 1.942;    /* <x1x2> + <y1y2>                           */
            corr[1] = 1.775;    /* <x1x3> + <y1y3>                           */
            corr[2] = 1.506;    /* <x1x4> + <y1y4>                           */
            corr[3] = 1.153;    /* <x1x5> + <y1y5>                           */
            corr[4] = 0.734;    /* <x1x6> + <y1y6>                           */
            corr[5] = 0.276;    /* <x1x7> + <y1y7>                           */
            ch_gain = 0;

         delay[0] = 0;   delay[1] = 1;   delay[2] = 3;
         delay[3] = 4;   delay[4] = 7;   delay[5] = 10;  delay[6] = 14;

          N = 15;

          attn[0] = sqrt(pow(10,-0.46)); attn[1] = 1;
        attn[2] = sqrt(pow(10,-0.43)); attn[3] = sqrt(pow(10,-0.65));
        attn[4] = sqrt(pow(10,-0.30)); attn[5] = sqrt(pow(10,-1.52));
        attn[6] = sqrt(pow(10,-2.17));

      /* Approximate channel gain.                    */
          for (i = 0; i < no_path-1; ++i) {
             for (j = i+1; j < no_path; ++j) {

               ch_gain += attn[i]*attn[j]/(delay[j]-delay[i])*sin(0.025*PI*(
                        delay[j]-delay[i]))*corr[j-i-1];
               }
          }
        ch_gain = sqrt((ch_gain*40/PI+attn[0]*attn[0]+attn[1]*attn[1]+attn[2]*
                attn[2]+attn[3]*attn[3]+attn[4]*attn[4]+attn[5]*attn[5])*2);
/* The easier way to find the channel gain is by running simulation; i.e. to determine it empirically. In this
case, the above approximation can be avoided altogether.     */

      /* Allocate sufficient memory to buffers and initialize them to zero.       */
        reg_real = (double *)calloc(N,sizeof(double));
        reg_imag = (double *)calloc(N,sizeof(double));

         start = 0;
}

     if (*new_frame == 1) {

   /* Reset the  condition for every new slot(frame) for the purpose of synchronizing
      the receiver to the transmitter.                             */
        for (i = 0; i < N; ++i ) {
           reg_real[i] = 0;
           reg_imag[i] = 0;
        }

    /*  Update the fading parameters for each frame.                      */
         for (i = 0; i < no_path; ++i)  {
           rayleigh(nfad1[i], skip, &fader);
          fad_re[0][i] = fader.real;
           fad_im[0][i] = fader.imag;
           rayleigh(nfad2[i], skip, &fader);
          fad_re[1][i] = fader.real;
           fad_im[1][i] = fader.imag;
          }
       *new_frame = 0;
      }

      for (i = 1; i < N; ++i)  {
         reg_real[N-i] = reg_real[N-1-i];  /* shifting contents.           */
        reg_imag[N-i] = reg_imag[N-1-i];
      }

      reg_real[0] = signal_in.real;  /* This is the complex signal at the        */
      reg_imag[0] = signal_in.imag;  /* output of a FM modulator. The            */
                                 /* amplitude of the signals is 1, i.e.      */
                                /* sqrt(2*E/T) = 1. (see notes, page 42)     */

      sig_out_2b[0].real = sig_out_2b[0].imag = 0.0;
      sig_out_2b[1].real = sig_out_2b[1].imag = 0.0;
    /* Calculate the channel outputs and normalized them to the channel gain.      */

      for (i = 0; i < no_path; ++i) {

     /* in-phase component of the attenuated complex baseband signal.          */
       attn_sig.real = reg_real[delay[i]]*attn[i];

     /* quadrature component of the attenuated complex baseband signal.        */
        attn_sig.imag = reg_imag[delay[i]]*attn[i];

       sig_out_2b[0].real += attn_sig.real*fad_re[0][i]-attn_sig.imag*fad_im[0][i];
       sig_out_2b[0].imag += attn_sig.real*fad_im[0][i]+attn_sig.imag*fad_re[0][i];
       sig_out_2b[1].real += attn_sig.real*fad_re[1][i]-attn_sig.imag*fad_im[1][i];
       sig_out_2b[1].imag += attn_sig.real*fad_im[1][i]+attn_sig.imag*fad_re[1][i];

          }

/* Output of the two independent multipath channels (sum of outputs from all the multipaths for each
branch)         */
       sig_out_2b[0].real /= ch_gain;
       sig_out_2b[0].imag /= ch_gain;
       sig_out_2b[1].real /= ch_gain;
       sig_out_2b[1].imag /= ch_gain;

    /* Undistorted second path output (dominant path for this profile)         */
      sync_path->real = reg_real[1];
      sync_path->imag = reg_imag[1];

    }

/*---------------------------------------------------- end file ------------------------------------------------*/


/*---------------------------------------------------- sync.c ------------------------------------------------
*/
/* This code is used to synchronize the receiver to the transmitter by using undistorted signal from the
dominant path in a multipath channel.          */

#include "spec.h"
void
sync(const cmplx_t ch_out,
        int *flag /* flag = 1 indicates the optimum clock phase is obtained.     */
        )
{
static int  start = 1;

double  new, diff;

static double  old;

*flag = 0;

new = atan2(ch_out.imag, ch_out.real);

if ( start == 3) {
        diff = new - old;
```

```
		if ((diff > 0 && diff < PI) || diff < -PI) {
			start = 1;
			*flag = 1;
		}

		else
			old = new;
		}

	else if (start == 2) {
		diff = new - old;

		if ( (diff < 0 && diff > -PI)|| diff > PI)
			start = 3;

		old = new;
	}

	else {
		old = new;
		start = 2;
	}
	}

/*------------------------------------------------------ end of file ---------------------------------------------------*/


/*---------------------------------------------- two-branch_rx_filter.c ---------------------------------------------*/
 /* This is a function of two receiver filters for Schemes II, III and IV. Each receiver filter is identical to the
one presented in APPENDIX A.	*/

#include "spec.h"
#define n_taps 51 /* number of taps of a transversal gaussian filter.	*/

void
two_rx_filter(const cmplx_t in_samp[],	/* input sample.		*/
		cmplx_t filter_out[]	/* output of the filter.		*/
		)
{
int i;

static int  start = 1, taps[n_taps];

static double  h[n_taps],	/* Tap-weights of the FIR Gaussian filter.	*/
		Br = 1.375e6; /* 3 dB IF bandwidth of the receiver filter. Br is determined from the equation Br*T
				= 1.1, where T is the bit duration, which is 800ns in this case.  */

static cmplx_t   buff1[n_taps], buff2[n_taps]; /* buffer for storing input samples at various taps. */

filter_out[0].real = filter_out[0].imag = 0;
filter_out[1].real = filter_out[1].imag = 0;

/* Initialize the buffer of the FIR filter and define the filter tap-weight. */
 if (start == 1) {
		for (i = 0; i < n_taps; ++i) {
			buff1[i].real = buff1[i].imag = 0;
			buff2[i].real = buff2[i].imag = 0;
```

```
                taps[i] = i - (n_taps-1)/2;
                h[i] = exp(-2*square(PI*Br/2*Ts*taps[i])/log(2));
        }
        start = 0;
}

/* Shift the contents of the buffer when receive a new sample.          */

for (i = 1; i < n_taps; ++i) {
        buff1[n_taps-i] = buff1[n_taps-1-i];
        buff2[n_taps-i] = buff2[n_taps-1-i];
}

buff1[0] = in_samp[0];
buff2[0] = in_samp[1];

/* Compute the filter output.                                  */
for (i = 0; i < n_taps; ++i) {
        filter_out[0].real += h[i]*buff1[i].real;
        filter_out[0].imag += h[i]*buff1[i].imag;

        filter_out[1].real += h[i]*buff2[i].real;
        filter_out[1].imag += h[i]*buff2[i].imag;
}

}

/*---------------------------------------------------- end file ----------------------------------------------------*/


/*---------------------------------------------------- (6,2)DFE.c ----------------------------------------------------*/
/* This DFE function is used in branch#2 of Scheme III. Branch#1 uses an almost identical function except
that the reference parameter is set by the DFE in branch#2. Schemes I and II also use a very similar DFE
function except that there is no need to return the error signal to the calling function.        */
/* This code is used to simulate the function of a fractional-spaced decision feedback equalizer with 6 T/2-
spaced taps in its feedforward filter and 2 T-spaced taps in its feedback filter. The adaptation algorithm
employed is the Recursive Least Square algorithm. Since the channel is assumed to be constant within one
time slot interval, the forgetting factor is set to 1. The constant for initializing the inverse autocorrelation
of the tap inputs is set at 1e5. A 16-bit training sequence is used. After the training mode the equalizer is
switched back */

#include "spec.h"

void
dfe(cmplx_t inputs[], /* two samples per bit input; inputs[0] preceeds inputs[1].        */
        cmplx_t train[], /* training MSK signal.                        */
        int *flag,
        cmplx_t *output, /* equalizer output.                          */
        double *er_pow  /* this error signal power is omitted in Schemes I & II.  */

        )
{
         int  i, j;

         static double  u_re[8], u_im[8], /* filters' complex tap inputs.        */
                 d_re, d_im,  /* real and imaginary components of a
                                 desired response.              */
```

```
           w_re[8] = {0,0,0,0,0,0,0,0}, /* filters' tap-weights.    */
           w_im[8] = {0,0,0,0,0,0,0,0},
           P_re[8][8], P_im[8][8];

      double  y_re, y_im, /* input to the decision device of the DFE.        */
         g_re, g_im, kp_re, kp_im, kp_pow, pi_re[8], pi_im[8],
        K_re[8], K_im[8], /* real and imaginary gain vectors.        */
        e_re, e_im; /* the a posterior error (real and imaginary).     */


      if (*flag > 2)  {  /* Start adaptation and equalizaton.               */

    /* Shifting feed-back filter tap inputs.                    */
     u_re[7] = u_re[6];     u_im[7] = u_im[6];
     u_re[6] = d_re;        u_im[6] = d_im;

    /* Shifting feed-forward filter tap inputs.                   */
     u_re[5] = u_re[3];     u_im[5] = u_im[3];
     u_re[4] = u_re[2];     u_im[4] = u_im[2];
     u_re[3] = u_re[1];     u_im[3] = u_im[1];
     u_re[2] = u_re[0];     u_im[2] = u_im[0];

    /* Input new samples (2 samples per bit) into the DFE.              */
     u_re[1] = inputs[0].real;  u_im[1] = inputs[0].imag;
     u_re[0] = inputs[1].real;  u_im[0] = inputs[1].imag;

     y_re = 0.0;       y_im = 0.0;
     kp_re = 1.0; /* The forgetting factor used is 1.0 .           */
     kp_im = 0.0;

     for(i = 0; i < 8; ++i) {
       pi_re[i] = pi_im[i] = 0.0;

       for(j = 0; j < 8; ++j) {
         pi_re[i] += u_re[j]*P_re[j][i] + u_im[j]*P_im[j][i];
         pi_im[i] += u_re[j]*P_im[j][i] - u_im[j]*P_re[j][i];

       }

      kp_re += pi_re[i]*u_re[i] - pi_im[i]*u_im[i];
      kp_im += pi_re[i]*u_im[i] + pi_im[i]*u_re[i];

      y_re += w_re[i]*u_re[i] + w_im[i]*u_im[i];
      y_im += w_re[i]*u_im[i] - w_im[i]*u_re[i];
     }

     kp_pow = kp_re*kp_re + kp_im*kp_im;

     if (*flag > 18) {       /* end of training. */

/* Note that only equalized signal is passed to the differential detector, for simplicity. Simulation results
had shown that there was no difference in the system performance when the equalized signal instead of the
actual output of the decision device was passed to the differential detector .     */
      output->real = y_re;
      output->imag = y_im;

     /* making decision for feedback.    */
      if (y_re > 0.5)
```

```
        d_re = 1;

      else if (y_re < -0.5)
         d_re = -1;

      else
         d_re = 0;


      if (y_im > 0.5)
         d_im = 1;

      else if (y_im < -0.5)
         d_im = -1;

      else
         d_im = 0;

    }

    else {
      d_re = train[*flag-3].real;
      d_im = train[*flag-3].imag;
    }

   e_re = d_re - y_re;
   e_im = d_im - y_im;
   *er_pow = e_re*e_re + e_im*e_im;  /* this is omitted for Schemes I & II  */

    for (i = 0; i < 8; ++i) {
      g_re = g_im = 0;

      for (j = 0; j < 8; ++j)  {
        g_re += u_re[j]*P_re[i][j] - u_im[j]*P_im[i][j];
        g_im += u_re[j]*P_im[i][j] + u_im[j]*P_re[i][j];
      }

      K_re[i] = (g_re*kp_re + g_im*kp_im)/kp_pow;
      K_im[i] = (g_im*kp_re - g_re*kp_im)/kp_pow;
      w_re[i] += K_re[i]*e_re + K_im[i]*e_im;
      w_im[i] += K_im[i]*e_re - K_re[i]*e_im;

      for (j = 0; j < 8; ++j) {
       P_re[i][j] -= K_re[i]*pi_re[j] - K_im[i]*pi_im[j];
        P_im[i][j] -= K_re[i]*pi_im[j] + K_im[i]*pi_re[j];
      }
    }

   }

/*  Initialization.                          */

  else if (*flag == 1) {
    for (i = 0; i < 8; ++i) {
      for (j = 0; j < 8; ++j) {
        if ( i != j)
          P_re[i][j] = 0;
          else
            P_re[i][j] = 1e5;

          P_im[i][j] = 0;

        }
      }
      u_re[3] = inputs[0].real;   u_im[3] = inputs[0].imag;
      u_re[2] = inputs[1].real;   u_im[2] = inputs[1].imag;
      u_re[6] = u_im[6] = 0;
      d_re = 0;
      d_im = 0;
    }

    else if (*flag == 2) {
      u_re[1] = inputs[0].real;    u_im[1] = inputs[0].imag;
      u_re[0] = inputs[1].real;    u_im[0] = inputs[1].imag;
    }

    *flag += 1;  /* this reference-control updating is omitted in the DFE function for branch #1)   */
}

/*------------------------------------------------------- end file --------------------------------------------------------*/


/*---------------------------------------------------- comb_div_dfe.c ---------------------------------------------------*/
/* This is a code which simulates the integration of 2-branch diversity-combining and DFE through joint-
parameter optimization. The structure has two feedforward fiters (one for each diversity branch) and a single
feedback filter. Each forward filter has four T/2-spaced taps. The feedback filter has two T-spaced taps. That
is, there are a total of 10 taps in the structure. The tap weights of all the filters are jointly estimated
adaptively by using conventional RLS algorithm.      */

#include "spec.h"

void
comb_div_dfe(cmplx_t inputs[][2], /* two samples per bit from each branch; inputs[0][] preceeds
                                        inputs[1][].      */
         cmplx_t *train, /* training MSK signal.            */
         int *flag,
         cmplx_t *output /* equalizer output.               */
         )
{
 int  i, j;

 static double  u_re[10], u_im[10], /* filters' complex tap inputs.  */
         w_re[10] = {0,0,0,0,0,0,0,0,0,0}, /* tap-weights.   */
         w_im[10] = {0,0,0,0,0,0,0,0,0,0},
    /* The first four elements of each of the above arrays are the tap-inputs(tap-weights) to FFF#1 and the
       following four elements are those to FFF#2 and the last 2 elements for the only FBF.
       */
         d_re, d_im,  /* real and imaginary components of a desired response.  */
         P_re[10][10], P_im[10][10];

double  y_re, y_im, /* input to the decision device of the DFE.    */
         g_re, g_im, kp_re, kp_im, kp_pow, pi_re[10], pi_im[10],
         K_re[10], K_im[10], /* real and imaginary gain vectors.    */
    e_re, e_im; /* the a posterior error (real and imaginary).  */

    if (*flag > 1)  {  /* Start adaptation and equalization.          */

/* Shifting feedback filter tap inputs.                          */
  u_re[9] = u_re[8];      u_im[9] = u_im[8];
  u_re[8] = d_re;         u_im[8] = d_im;

/* Shifting feedforward filters' tap inputs.                     */
/* FFF#1    */

  u_re[3] = u_re[1];      u_im[3] = u_im[1];
  u_re[2] = u_re[0];      u_im[2] = u_im[0];

/* FFF#2    */

  u_re[7] = u_re[5];      u_im[7] = u_im[5];
  u_re[6] = u_re[4];      u_im[6] = u_im[4];

/* Input new samples (2 samples per bit) into the FFF#1.            */
  u_re[1] = inputs[0][0].real;  u_im[1] = inputs[0][0].imag;
  u_re[0] = inputs[1][0].real;  u_im[0] = inputs[1][0].imag;

/* Input new samples (2 samples per bit) into the FFF#2.            */
  u_re[5] = inputs[0][1].real;  u_im[5] = inputs[0][1].imag;
  u_re[4] = inputs[1][1].real;  u_im[4] = inputs[1][1].imag;


   y_re = 0.0;       y_im = 0.0;
   kp_re = 1.0; /* The forgetting factor used is 1.0 .           */
   kp_im = 0.0;

   for(i = 0; i < 10; ++i) {
      pi_re[i] = pi_im[i] = 0.0;

      for(j = 0; j < 10; ++j) {
        pi_re[i] += u_re[j]*P_re[j][i] + u_im[j]*P_im[j][i];
        pi_im[i] += u_re[j]*P_im[j][i] - u_im[j]*P_re[j][i];

      }

     kp_re += pi_re[i]*u_re[i] - pi_im[i]*u_im[i];
     kp_im += pi_re[i]*u_im[i] + pi_im[i]*u_re[i];

     y_re += w_re[i]*u_re[i] + w_im[i]*u_im[i];
     y_im += w_re[i]*u_im[i] - w_im[i]*u_re[i];
   }

   kp_pow = kp_re*kp_re + kp_im*kp_im;


   if (*flag > 17) {        /* end of training. */

      output->real = y_re;
       output->imag = y_im;

    /* making decision for feedback.     */
      if (y_re > 0.5)
```

```
        d_re = 1;

     else if (y_re < -0.5)
         d_re = -1;

     else
         d_re = 0;


     if (y_im > 0.5)
        d_im = 1;

     else if (y_im < -0.5)
         d_im = -1;

     else
         d_im = 0;
  }

  else {
    d_re = train[*flag-2].real;
    d_im = train[*flag-2].imag;
  }

     e_re = d_re - y_re;
  e_im = d_im - y_im;

  for (i = 0; i < 10; ++i) {
    g_re = g_im = 0;

    for (j = 0; j < 10; ++j)  {
      g_re += u_re[j]*P_re[i][j] - u_im[j]*P_im[i][j];
      g_im += u_re[j]*P_im[i][j] + u_im[j]*P_re[i][j];
    }

   K_re[i] = (g_re*kp_re + g_im*kp_im)/kp_pow;
   K_im[i] = (g_im*kp_re - g_re*kp_im)/kp_pow;
   w_re[i] += K_re[i]*e_re + K_im[i]*e_im;
   w_im[i] += K_im[i]*e_re - K_re[i]*e_im;

   for (j = 0; j < 10; ++j) {
     P_re[i][j] -= K_re[i]*pi_re[j] - K_im[i]*pi_im[j];
     P_im[i][j] -= K_re[i]*pi_im[j] + K_im[i]*pi_re[j];
   }
  }
   }

else if (*flag == 1) {  /* Initialization.                    */

 for (i = 0; i < 10; ++i) {
   for (j = 0; j < 10; ++j) {
     if (i != j)
       P_re[i][j] = 0;

     else
       P_re[i][j] = 1e5;

     P_im[i][j] = 0;
```

```
        }
      }
    /* Input new samples (2 samples per bit) into the FFF#1.          */
      u_re[1] = inputs[0][0].real;  u_im[1] = inputs[0][0].imag;
      u_re[0] = inputs[1][0].real;  u_im[0] = inputs[1][0].imag;

    /* Input new samples (2 samples per bit) into the FFF#2.          */
      u_re[5] = inputs[0][1].real;  u_im[5] = inputs[0][1].imag;
      u_re[4] = inputs[1][1].real;  u_im[4] = inputs[1][1].imag;

      u_re[8] = u_im[8] = 0;
      d_re = d_im = 0;
     }

     *flag += 1;

       }

/*------------------------------------------ end of file ------------------------------------------*/


/*---------------------------------------- two_branch_detect.c ----------------------------------------*/
/* One-bit Differential Detector for two independent branches in Schemes II, and III.  */
/* Its basic function is the same as the one presented in APPENDIX A.  */

#include "spec.h"


void
detect(const cmplx_t  sig_in[], /* output of a receiver filter.    */
          int detect_bit[]   /* detected bit.                  */
          )
          {

double  ph_new[2],
           delta_ph[2];

          static double  ph_old[2] = {0,0};

       ph_new[0] = atan2(sig_in[0].imag, sig_in[0].real);
       ph_new[1] = atan2(sig_in[1].imag, sig_in[1].real);
      /* atan2() returns the arc tangent of y/x in the range [-pi, +pi]
          radians. It uses the signs of both y and x to determine the
           quadrant of the return value.                   */

      delta_ph[0] = ph_new[0] - ph_old[0];
      delta_ph[1] = ph_new[1] - ph_old[1];

      detect_bit[0] = sin(delta_ph[0]) > 0 ? 1 : -1;
      detcct_bit[1] = sin(delta_ph[1]) > 0 ? 1 : -1;

      ph_old[0] = ph_new[0];
      ph_old[1] = ph_new[1];
        }

/*---- ---------------------------------------- end of file -------------------------------------------*/
```