DELFT UNIVERSITY OF TECHNOLOGY
FACULTY OF ELECTRICAL ENGINEERING
SECTION TELECOMMUNICATIONS AND
TRAFFIC CONTROL SYSTEMS

Title: 'The Effect Of Error Correcting Coding On Indoor Wireless Communications Systems In The 20 - 60 GHz Region.'

Authors: Bram J. Bout
Willem A. Schouten

Task Report
Size: 30 pages
Date: March 1992

Professor: Prof. dr. J.C. Arnbak
Mentor: Prof. dr. R. Prasad
Assignment Nr.: T - 504
Period: January 1992 - March 1992

As a follow-up of the literature study 'An Overview Of Indoor Wireless Communications Systems In The 20 - 60 GHz Region', published by B.J. Bout and W.A. Schouten in December 1992, a further study has been performed to investigate the effect of Forward Error Correcting Coding on Indoor Wireless Communications Systems. This has been done by calculating the average fade- and non-fade duration as a function of the frequency and the Signal To Noise ratio. These results are conveyed to a Bit Error Probability. After that, the same calculations are done for the same channel, but with Forward Error Correcting Coding.

List of symbols

$f(r)$	distribution function of signal level
I_0	modified Bessel function of first kind and zero order
k	number of data bits
m	number of errors
n	code length
N_T	total noise power
p_0	average power of fading signal
P_d	desired signal power level
P_e	bit error probability
r	signal level
R	Rice factor
R_T	data rate
s	amplitude of direct wave
t	burst length
v	speed difference transmitter - receiver
γ	instantaneous SNR
σ_d	scattered desired signal component

List of Abbreviations

BER	Bit Error Rate
BPSK	Binary Phase Shift Keying
DCA	Dynamic Channel Allocation
DPSK	Differential Phase Shift Keying
FDMA	Frequency Division Multiple Access
FEC	Forward Error Correcting
ISDN	Integrated Services Digital Network
LOS	Line Of Sight
RS	Reed Solomon
SNR	Signal to Noise Ratio

Contents

Introduction

Wireless communication systems are becoming more and more important these days. The next step in this cellular technology is Indoor Wireless Communication. Although the idea of an Indoor Wireless Communication System is not entirely new, some new developments have focussed attention to these systems and have increased the number of studies in this area. One of these new developments is the idea of communicating in the so far hardly used frequency region of 20-60 GHz. One of the institutions that stimulates research in this field is the European Cooperation for research in the field of Science and Technology (COST). In their COST 231 program, a lot of studies are done in the area of Indoor Wireless Communication Systems in the 20-60 GHz region.
As a part of this COST 231 program we have done a literature study to establish the results that have thus far been published in this field. In this literature study, which will be referred to as reference [1], we also gave some recommendations for further research. One of these recommendations was, that the effect of error correcting coding should be investigated, because we thought that the reliability of the channel could be improved with appropriate use of this technique.
Following this recommendation, we investigated the effect of Forward Error Correcting Coding (FEC coding) on the 20-60 GHz radiolinks that are applied in the Indoor Wireless Communication System. The results of this study are published in this report.

In Chapter 1 we will give a brief overview of the channel characteristics of the 20-60 GHz region. Special attention is given to the outage probability. For a more detailed description of the channel characteristics we refer to [1]. To be able to design a Forward Error Correcting code to correct burst errors, we must first establish the average length of such a burst. In Chapter 2 we calculate this expected burst length.
The Error Correcting code we propose to correct bursts of the calculated length is given in Chapter 3. The conclusions are given in Chapter 4.

This task report is the second part of our total task, consisting of the literature study, published in 'An Overview Of Indoor Wireless Communications Systems', December 1991 and this report.

Chapter 1

Channel characteristics in the 20-60 GHz region

1.1 Channel Model

In [1] we established that the indoor radio channel is dominated by the multipath effect. This is the effect that one signal arrives several times at the receiver due to reflections from walls, furniture, etc. However, there is, at least almost always, a strong Line-Of-Sight path, which is the direct path (no reflection) from transmitter to receiver. A channel which has no LOS path can be characterized by a Rayleigh fading channel, but the indoor wireless channel, due to the presence of this Line-Of-Sight (LOS) path plus the multipaths, is described by the Rician channel model.

The Rician distribution function of the signal level r is given by $f(r)$:

$$f(r) = \frac{r}{p_0} \exp\left[-\frac{r^2+s^2}{2p_0} \right] I_0 \left[\frac{rs}{p_0} \right] \qquad (1.1)$$
$$where \quad 0 \leq r < \infty \ , \quad s \geq 0$$

where $I_0()$ is the modified Bessel function of the first kind and zero order, s is the amplitude of the direct (LOS) wave, also known as the specular radio signal and p_0 is the average power of the fading signal. The environment in which the Indoor Wireless Communication takes place is usually characterized by the Rician factor R. This Rician factor is given by the ratio of the signal power from the dominant signal path and the scattered (multipath) signal:

$$R = \frac{s^2}{2p_0} \qquad (1.2)$$

Two typical values for R are 6.8 dB for 'old' brick buildings with closed rooms and 11 dB for offices with an open structure, known as 'office gardens' [2].

1.2 The behaviour of the channel

Beside the Rician fading, explained in the previous section, the indoor radio channel suffers from another type of fading, known as shadow fading. This type of fading is caused by large objects (e.g. furniture, doors, people) obstructing the Line Of Sight path.

We define an *outage* as the state of the channel when the received signal power level is so low, that we are not able to correctly receive the message.
We assume a channel to be in either of two states: on or off. In the 'on' state, the bits are received perfectly, without errors. In the 'off' state (outage) the received bits are uncorrelated, and the probability of error is 0.5.

The channel is 'on' if the received signal power is above a certain threshold, and 'off' when it is below this threshold. From Fig. 1.1 we can see that the channel has relatively long 'on' periods, followed by short 'off' periods. In order to design a Forward Error Correcting Code to correct the 'off' bursts, we must first calculate the average length of such a burst and how often they occur. These calculations are done in the next chapter.

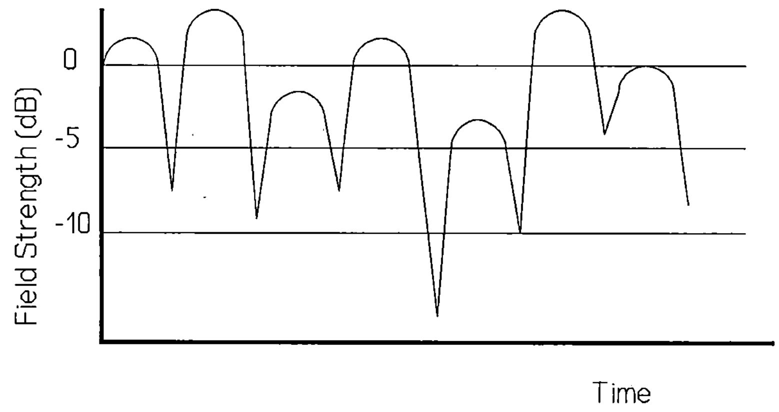

Fig. 1.1 Example of time-variant fading in a channel ([4])

The duration of the channel outages caused by shadow fading can be very long. Therefore, it is impossible to design a Forward Error Correcting code for this type of fading, so shadow fading will not be considered any further in this report. In [1] it is shown that, for high data rates, the indoor radio channel can be characterized as a slow fading, frequency selective channel. This means that the received signal power is assumed to be constant during one bit interval (slow fading) and that the channel characteristics (amplitude and phase) are not constant for all frequencies in the message band (frequency selectivity). An example of received signal power versus time is given in Fig. 1.1.

Chapter 2

Calculation of the burst length and Bit Error Rate

2.1 The fade- and non-fade duration

A method for calculating the average fade- and non-fade duration for a Rician fading channel is given in [3]. We assume the probability distribution of T_{fade} to be Poisson distributed. Therefore:

$$Pr\{T_{fade} < 4.6\ \overline{T_{fade}}\} \approx 99\% \qquad (2.1)$$

We have calculated the values for the average fade duration using a pascal program, supplied in [3]. We slightly modified this program to make it more suitable for our calculations. The listing is given in the Appendix. The fade- and non-fade duration can be calculated with the level crossings rate N_r, given as:

$$N_R = \sqrt{\left(\frac{b_2}{2\pi}\right)} \times p(R*) \qquad (2.2)$$

where $p(R*)$ is the probability density of signal level $R*$ (the fading threshold) and b_2 is given as:

$$b_2 = \sigma_r^2\ (2\pi f_m)^2 \qquad (2.3)$$

in which f_m is the maximum Doppler shift frequency, defined as

$$f_m = \frac{\lambda}{v} \qquad (2.4)$$

The average fade duration is related to the level crossing rate as follows:

$$T_f = \frac{1}{N_R} \int_0^{R*} p(r)\ d(r) \qquad (2.5)$$

and the non-fade duration can be calculated with

$$T_n = 1 - \frac{1}{N_R} \int_0^{R*} p(r) d(r) \qquad (2.6)$$

These integrals are written as a Marcum Q-function, which is defined as:

$$Q(\alpha,\beta) = \int_{\beta}^{\infty} x \exp[-\tfrac{1}{2}(x^2+\alpha^2)]\, I_0(\alpha x)\, dx \tag{2.7}$$

where

$$\alpha = \sqrt{2R} \quad \wedge \quad \beta = \sqrt{2}\,\rho \tag{2.8}$$

in which R is the Rician parameter and ρ is given as:

$$\rho = \frac{R*}{\sqrt{2\sigma^2}}$$

Plots of the average fade duration versus receiver threshold and versus frequency are given in Fig. 2.1 and Fig. 2.2. As input parameters we used:

v	(speed difference between transmitter and receiver) this is the speed of a slowly moving person	= 0.5 km/h
R	(Rice factor) this is the Rice factor of a brick building	= 6.8 dB

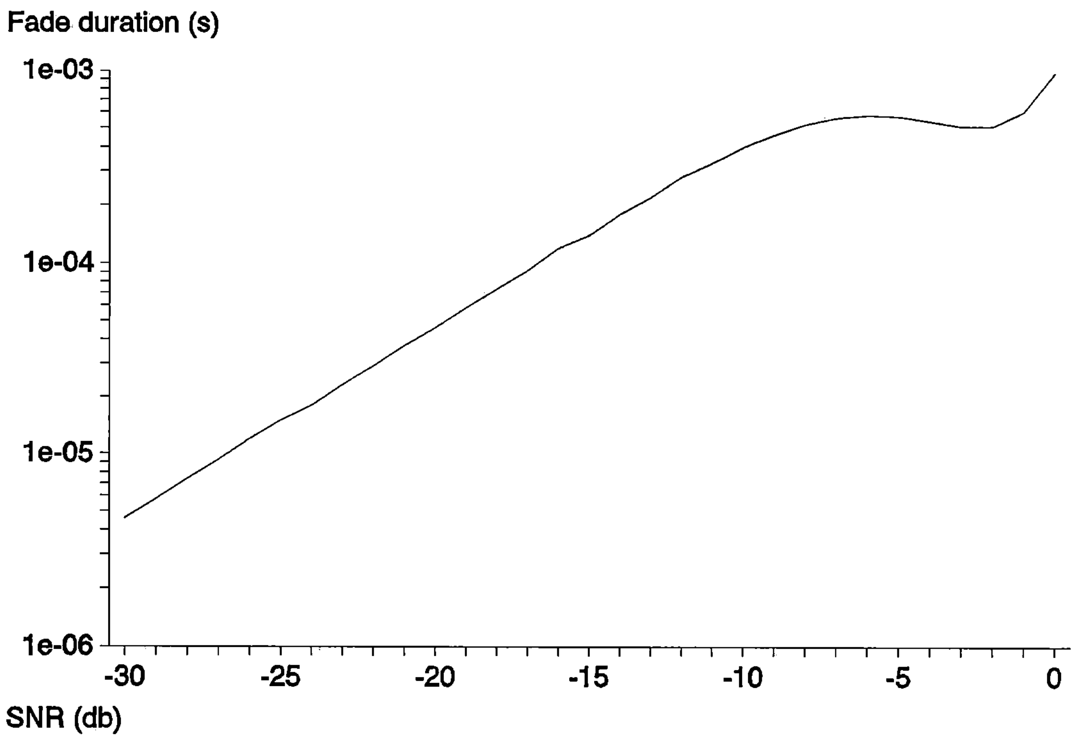

Fig. 2.1 Average fade duration vs receiver threshold (frequency is 60 GHz)

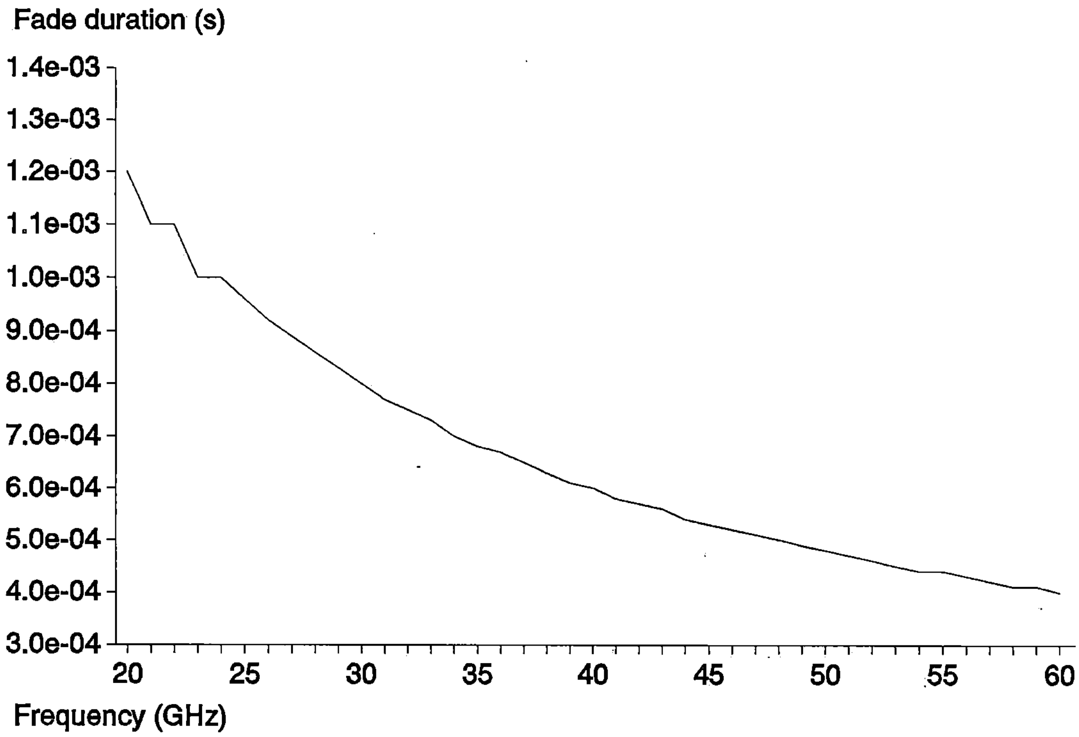

Fig. 2.2 Average fade duration vs frequency (receiver threshold is -10 dB)

From both figures we can read that the average fade duration for a system transmitting at 60 GHz, with a receiver threshold level of minus 10 dB, is 0.4 ms.

Plots of the average non-fade duration versus receiver threshold and versus frequency are given in Fig. 2.3 and Fig. 2.4. The same input parameters were used.

We can extract from both figures that the average non-fade duration for a system transmitting at 60 GHz, with a receiver threshold level of minus 10 dB is 5 seconds.

Some values for the average fade- and non-fade durations for the two values of the Rice parameter are given in Table 2.1.

	fade-duration (s)		non-fade duration (s)	
	6.8 dB	11 dB	6.8 dB	11 dB
60 GHz	4.0×10^{-4}	1.0×10^{-4}	4.7	2.9×10^{3}
40 GHz	6.0×10^{-4}	1.5×10^{-4}	7.0	4.3×10^{3}
20 GHz	1.2×10^{-3}	3.1×10^{-4}	14.0	8.6×10^{3}

Table 2.1 Fade and non-fade durations for different values of the Rice parameter. The receiver threshold is -10 dB.

We can extract from this table that the fade-duration at the low Rice-parameter of 6.8 dB is in the same order as the fade duration at the high Rice parameter of 11 dB. However, as the

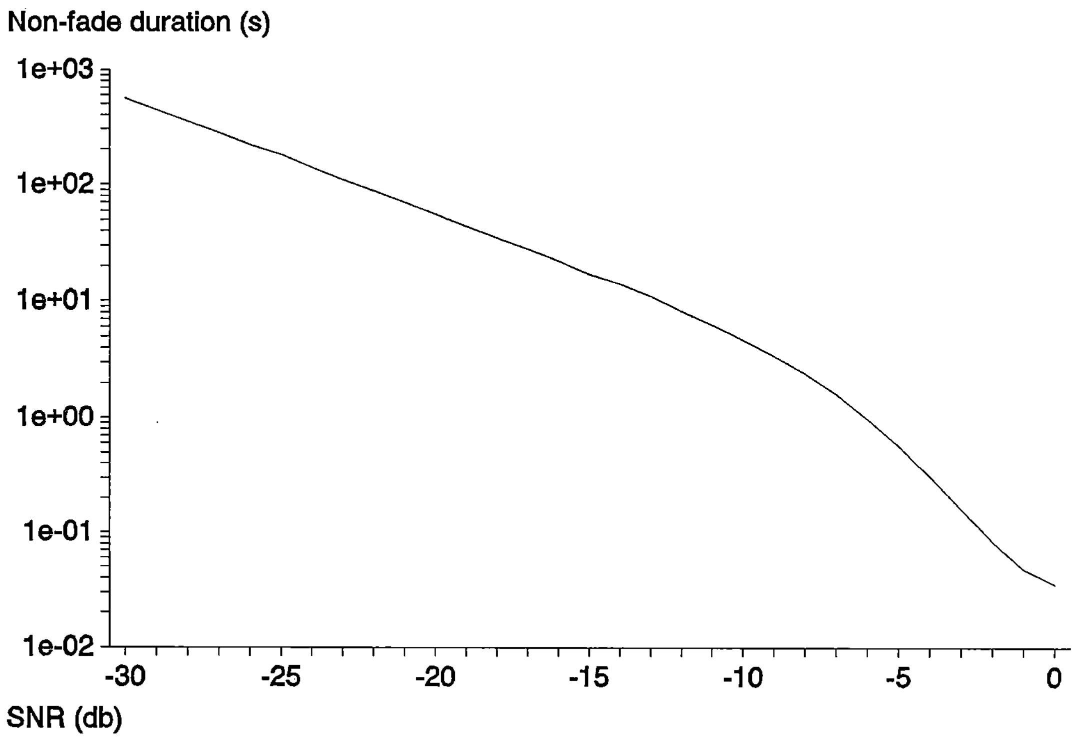

Fig. 2.3 Non-fade duration versus receiver threshold (frequency is 60 GHz)

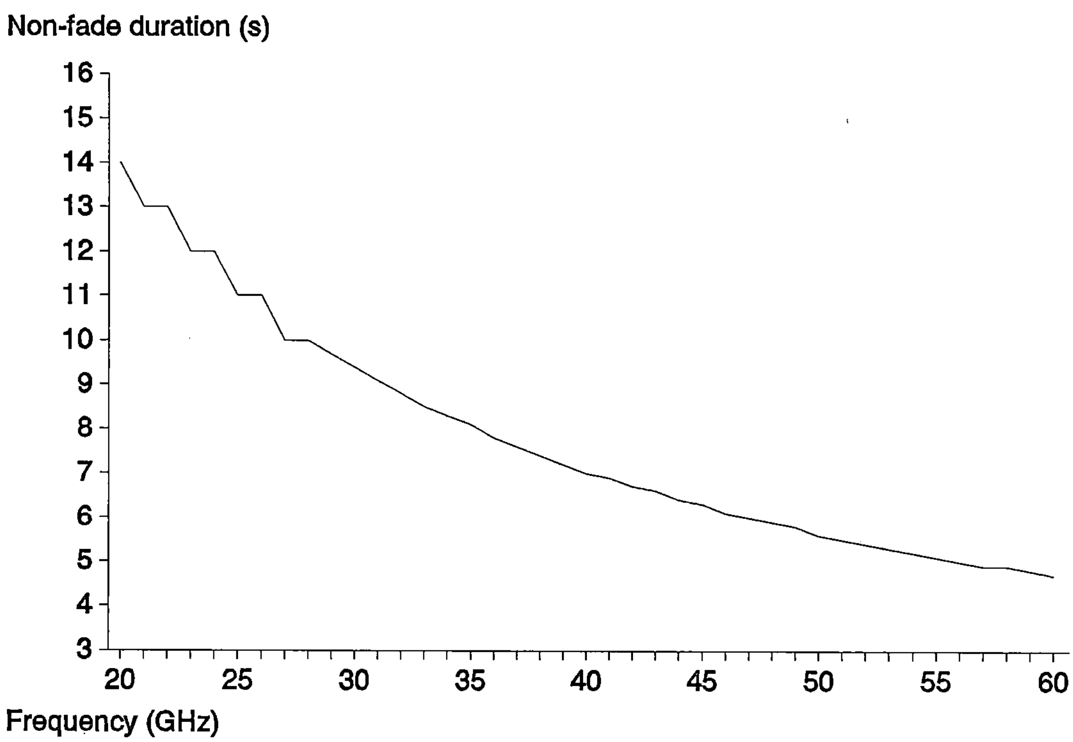

Fig. 2.4 Non-fade duration versus frequency (receiver threshold is -10 dB)

Rice parameter increases, the non-fade-duration increases dramatically. It is almost a factor of 1000 larger at 11 dB than at 6.8 dB. At 60 GHz, a fade occurs once every 48 minutes. Any type of Forward Error Correcting coding is not appropriate here.

Using formula (2.1) we are now able to compute the maximum expected burst length for 99 % of all bursts, given a 0.4 ms. average fade duration. We find that $T_{fade,\ 99\%}$ (the 99*th* percentile) $\approx$ 1.8 ms.

2.2 Calculation of the Bit Error Rate

What we want to do next is to calculate the Bit Error Rate as a function of the Signal to Noise ratio at the receiver, using Differential Phase Shift Keying (DPSK). We do the calculations for DPSK because this is the most commonly used form of digital transmission. We will perform our calculations for three cases. First we give the BER for the ideal case: a channel that does not suffer from fading. After that we will take a look at the more realistic case of a Rician-fading channel and finally (this can be found in Chapter 3) for the Rician-fading channel using FEC coding.

For the ideal case, the bit error probability can be given by [5]:

$$P_e = \frac{1}{2} e^{-SNR} \tag{2.10}$$

where SNR is the signal-to-noise ratio. We will discuss the result in Chapter 3.

As described above, the next step is to take fading into account. This is done by combining the error probability without fading and the Rician distribution function, as given in Formula (1.1).

We are interested in the average of the error probability, which can be calculated by taking the integral from zero to infinity of the combined functions [4]:

$$\overline{P_e} = \int_0^\infty P_e \frac{1}{\sigma_d^2} \exp\left(\frac{-2P_d + s^2}{2\sigma_d^2}\right) I_0\left(\frac{\sqrt{2P_d}\, s}{\sigma_d^2}\right) dP_d \tag{2.11}$$

where P_e is error probability without fading, P_d is the desired signal power, σ_d is the scattered desired signal component, s is the desired specular signal component and $I_0()$ is the modified Bessel function of the first kind and zero order.

When we put Formula (1.1) in Formula (2.11), we get:

$$\overline{P_e} = \int_0^\infty \frac{1}{2} \exp\left(-\frac{P_d}{N_t}\right) \frac{1}{\sigma_d^2} \exp\left(\frac{-2P_d + s^2}{2\sigma_d^2}\right) I_0\left(\frac{\sqrt{2P_d}\, s}{\sigma_d^2}\right) dP_d \tag{2.12}$$

where N_t is the total noise power.

Next we define

$$\frac{P_d}{N_t} = \gamma = \textit{instantaneous SNR}, \tag{2.13}$$

and

$$R=\frac{s^2}{2\sigma_d^2} \Rightarrow \sigma_d^2=\frac{P_d}{1+R} \tag{2.14}$$

where R is the Rician parameter.

After some calculations we can rewrite formula (2.12) in a very complicated way. This very large formula, given in [4], can be simplified for our channel model, because in indoor wireless communications the signal does not suffer from cochannel interference. The reason is elaborately given in [1], so here we suffice by stating that the picocells are well shielded from each other by walls and that oxygen absorption also helps reducing the reuse distance. Formula (2.11) can thus be rewritten as:

$$\overline{P_e}=\int_0^\infty \tfrac{1}{2}\exp(-\gamma)\,\frac{1+R}{SNR}\exp\left(-\left(\gamma\frac{1+R}{SNR}+R\right)\right)I_0\left(\sqrt{2\gamma\frac{1+R}{SNR}}\,\sqrt{2R}\right)d\gamma \tag{2.15}$$

In this, still overwhelming, formula SNR is defined as:

$$SNR=\frac{\left(\frac{S_d^2}{2}+\sigma_d^2\right)}{N_t} \tag{2.16}$$

We can calculate this integral using a simple pascal program. This results in the following Bit Error Rate versus SNR:

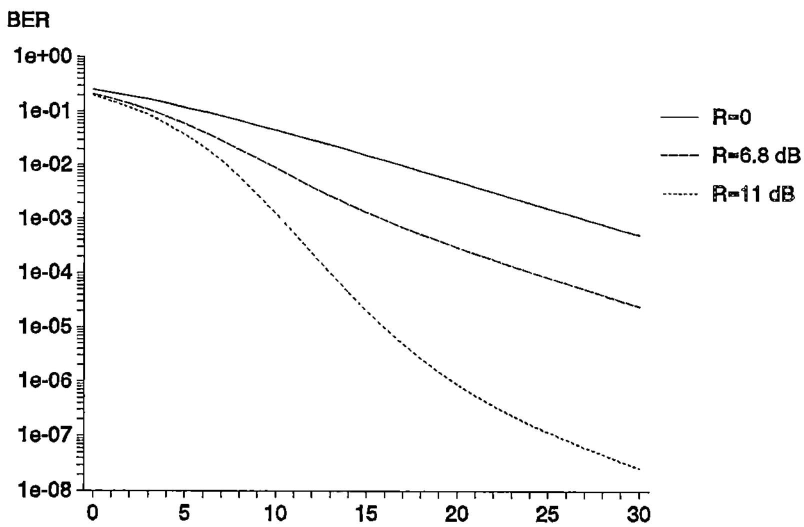

Fig. 2.5 BER versus SNR for a channel without FEC coding.

Chapter 3

The Error Correcting Code

3.1 Description of an example system

In the previous chapter we have calculated the expected maximum burst length. The next thing we want to do is to convert this knowledge into a practical Error Correcting Code. For this we need to know the bit rate of the channel. We will now describe an example system for which we will design a code. The values of the parameters in this system are not chosen at random, but are quite typical for Indoor Wireless Communication Systems [1].

We assume an FDMA system with users transmitting at a fixed frequency at 100 kbps. At this low data-rate the channel is frequency non-selective. We have chosen 100 kbps for each user, to be able to grant them an error-free 64 kbps voice/data channel, which can be used as a wireless extension of an ISDN D1 channel. The 36 kbps surplus bits are redundancy bits, which can be used to correct the errors.

3.2 The burst

Combining the results of sections 2.1 and 3.1, a 100 kbps channel and a maximum burst length of 1.8 ms, gives us a maximum expected burst length of 180 bits. This is the burst length that we want to correct. When we are able to do so, this means that we are able to correct 99% of all the bursts, or, in other words, only 1% of the bursts can not be corrected.

3.3 Reed-Solomon codes

A Reed-Solomon (n,k) code is defined as a code of length n bits, containing k databits and n-k redundancy bits. It can be shown [6] that the RS(n,k) code is capable of correcting a burst of length t,

$$t = \frac{n-k}{2} \tag{3.1}$$

The code rate R of the RS(n,k) code is defined as:

$$R_T = \frac{k}{n} \tag{3.2}$$

and, in our example, should not go below 0.64.

From the above equations, using t=180, we can calculate the values for n and k using:

$$t = \frac{(1 - 0.64)n}{2} = 180 \Rightarrow \qquad (3.3)$$
$$n = 1000, \; k = 640$$

These values seem rather high, but the length of the code can be shortened by making use of a technique called interleaving. Here, the databits are not transmitted sequentially, but are arranged in columns. The rows of the matrix are now transmitted and a RS(n',k') code is applied to the *rows* of the transmitted data. See Fig. 3.1 for an explanation. The resulting Reed-Solomon code is much shorter, which simplifies the design, whilst the error correcting capability of the code remains the same.
The technique is further explained in [6].

data bits | redundancy bits

d0	d6						r0		
d1	d7						r1		
d2	..					..	r2		..
d3	..					..	..		..
d4						d40	..		r16
d5						d41			r17

Fig. 3.1 The interleaving technique. The bit transmission order is d0, d6, d12, ..., r0, r6, r12, d1, d7, ...

3.4 The outage probability using FEC coding

Given an error probability P_e, as calculated in section 2.2, we can calculate the probability of exactly m errors occurring in a block of length n with basic probability theory:

$$P(m \; errors|P_e) = \binom{n}{m} P_e^m (1-P_e)^{n-m} \qquad (3.4)$$

The Reed-Solomon code considered in this report can correct any combination of up to t errors, so the probability of successful reception is:

$$P(succes|P_e) = \sum_{m=0}^{t} \binom{n}{m} P_e^m (1-P_e)^{n-m} \tag{3.5}$$

When this is the probability of successful reception, the probability of an error is easily found as:

$$P_{e,FEC} = 1 - P(success|P_e) \tag{3.6}$$

Numerical calculation of the above formulas with a computer poses accuracy problems if the value of n becomes very large, or P_e becomes very small, both of which are true in our case. However, when n is (much) larger than 10, which is certainly the case in our system, the binomial distribution function can be approximated by the Poisson distribution.
Equation (3.5) then becomes:

$$Pr(success|P_e) = 1 - e^{-\frac{t}{nP_e}} \tag{3.7}$$

The results of these calculations, where P_e is taken from Fig. 2.5, are given in Fig. 3.2 and Fig. 3.4 for different values of n and k. It is clear from these figures that the BER falls very rapidly when the signal to noise ration becomes better than approximately 15 dB.

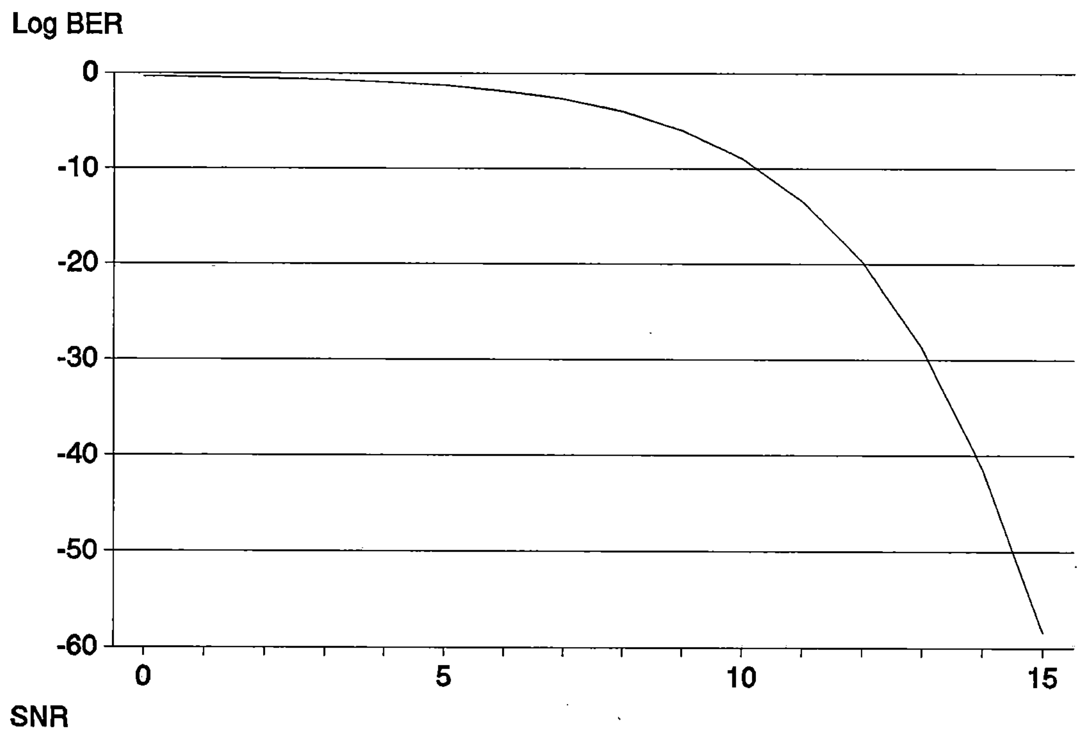

Fig. 3.2 Log BER versus SNR for n=1000 and k=640 (180 bits corection capability)

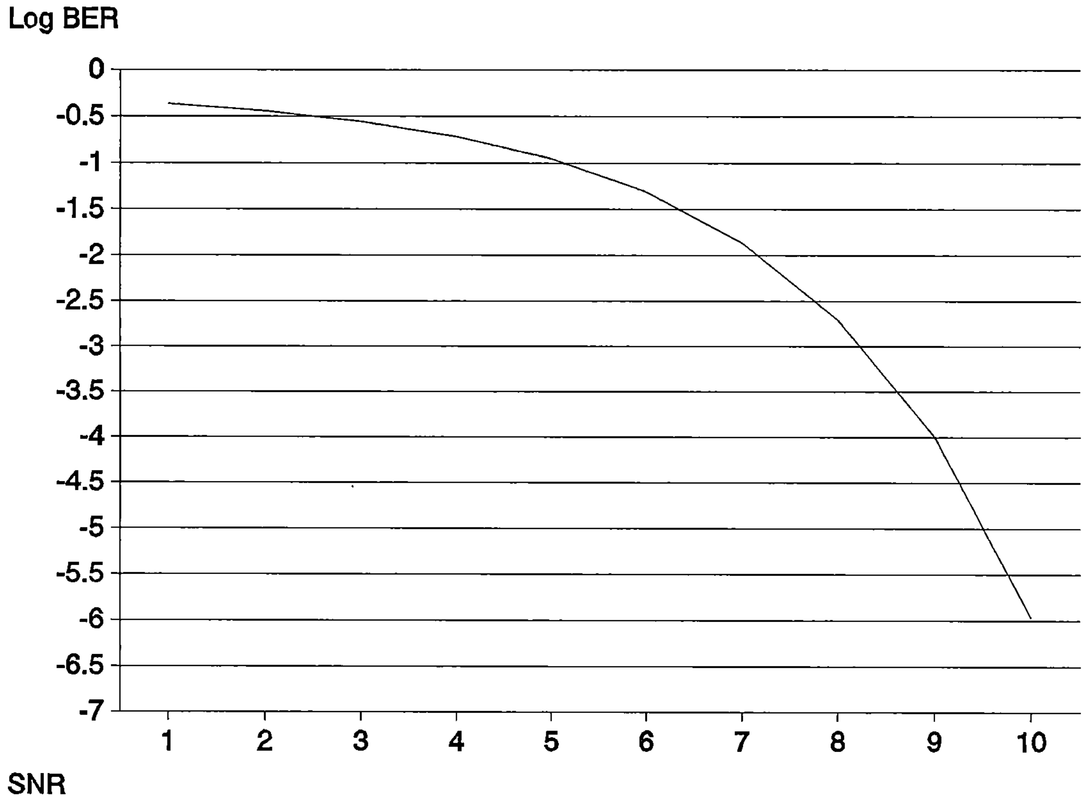

Fig. 3.3 Enlargement of previous figure, with the same parameters

3.5 Comparison of the results

In order to compare the effect of fading on the BER, and the effect of FEC coding (on the faded signal), we have summarized the bit error rates for the three cases (no fading, fading and fading with coding) in Fig. 3.5.
It can be seen that, for practical values of the signal-to-noise ratio (above 10 dB), the fading of the channel greatly increases the Bit Error Rate, compared to the non-fading (ideal case). From the figure it is also clear that FEC coding improves performance drastically, especially for higher signal-to-noise ratios. Of course, the increase in performance will be smaller if the error correcting capability of the code is decreased by using a smaller n (in this plot n=1000 and k=640), see Fig. 3.4, but the results are still encouraging.

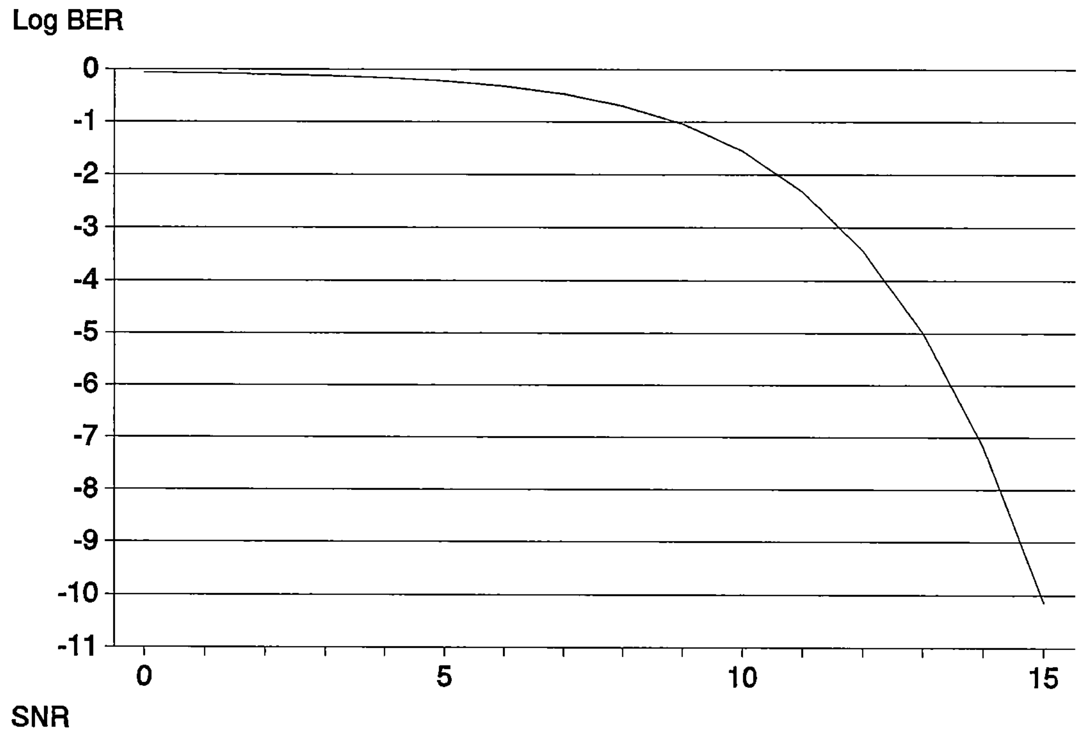

Fig. 3.4 Log BER versus SNR for n=64, k=60 (2 bit correction capability)

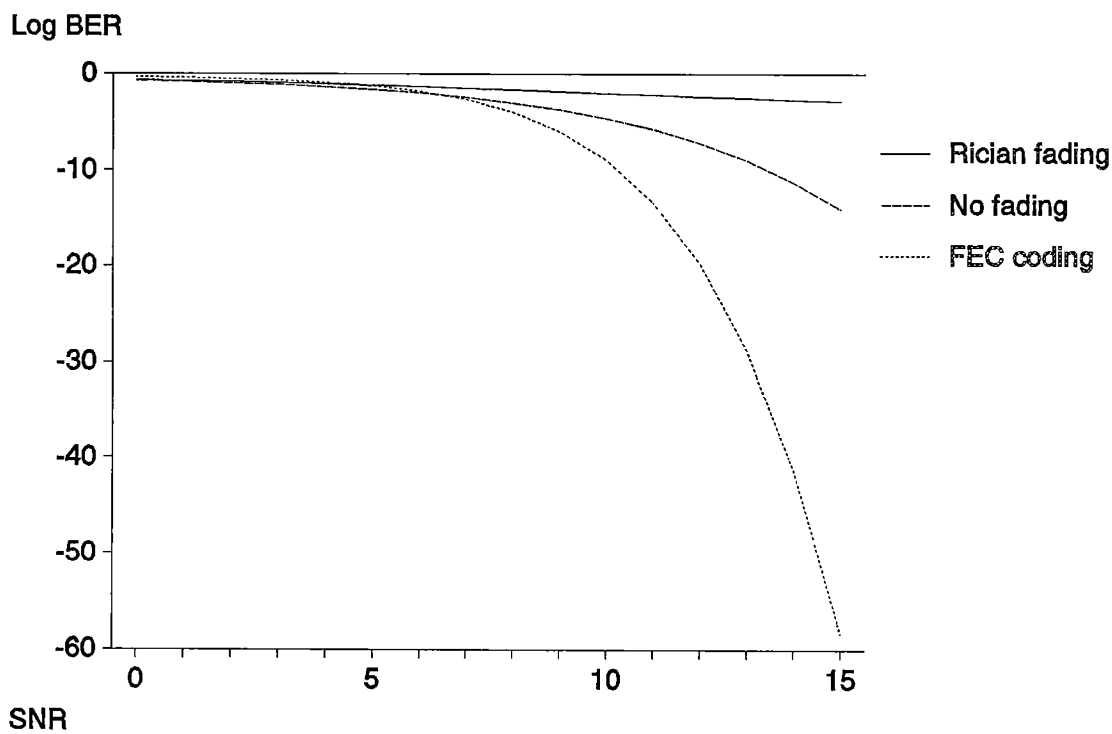

Fig. 3.5 The BER for various cases. The Rician fading parameter used is 6.8 dB

3.6 The time delay caused by the FEC coding

The Reed-Solomon code is a block code, which means that all the *k* databits in a block must be known before transmission of the block can start. Also, decoding of the block can only be done once the total block (*n* bits) has been received. This causes a time delay, which may be annoying, especially if the data represents real-time speech. The delay can be calculated by:

$$\tau_d = \frac{k}{R_{data}} + \frac{n}{R_{channel}} \tag{3.8}$$

where R_{data} and $R_{channel}$ are the data collection- and data transmission rates respectively. The delay invoked by the Reed-Solomon(1000, 640) code described here is 20 ms.

Studies have shown that the maximum delay which does not cause annoyance in a speech conversation is 12 ms [1], although in satellite communication delays up to 400 ms can occur easily. If the delay caused by FEC coding is considered too long, the only option is to shorten the code and thus the error correction capability of the code (the length of the burst).

Chapter 4

Conclusions and recommendations

The indoor wireless communication channel suffers from burst errors, caused by the multipath effect. The length of the burst will increase if the Rice parameter, which depends on the building type, is decreased.
We studied the behaviour of the channel for a Rice parameter of 6.8 dB, which is typical for old brick buildings. The average burst length was found to be 0.4 ms at 60 GHz, for slow moving persons. For high Rice parameters, for example 11 dB, which is typical for modern offices without separate rooms, known as 'office gardens', the average non-fade duration becomes very large. The fade duration for this Rice parameter is about the same as for the case of the 6.8 dB Rice parameter. This means that Forward Error Correcting coding can be considered superfluous under these circumstances.

Based on a 100 kbps radio channel, used for transmitting 64 kbps data, a FEC code was designed which improved the BER several orders of magnitude. For signal-to-noise ratio's of above 15 dB, even very little redundancy in the FEC coding (code rate 0.94 in our example) the channel reliability improves enormously. For lower values of the SNR more redundancy is needed, but a code rate of 0.64 gives a BER of 10^{-10} for a signal-to-noise ratio of 10 dB.

The cost for using FEC coding is threefold:
1. FEC coding uses extra bandwidth, thus the spectrum efficiency decreases with higher reliability
2. FEC coding introduces a time delay, which may be unacceptable. Note, however, that using error *detection* and retransmission also introduces a time delay.
3. FEC coding complicates the design.

For milimetre-wave applications, as considered in this report, there is a lot of bandwith available. The frequency reuse distance in pico-cells is very short, because co-channel interference is very low in buildings, due to the shielding of concrete walls and the phenomenon of Oxygen absorption. We have shown that FEC coding can greatly improve the performance of an indoor radio channel, and therefore FEC coding seems suitable for indoor radio links at high frequencies.

We suggest that the effect of other types of FEC coding, such as continuous coding, on channel reliability and data delay are investigated. Another topic of interest might be the influence of co-channel interferers on BER (without FEC coding) at high frequencies.

References

[1] **B.J. Bout and W.A. Schouten**
'An Overview Of Indoor Wireless Communication Systems In The 20-60 GHz Region'
Thesis Report, december 1991
Delft University of Technology
Faculty of Electrical Engineering

[2] **E. Walther**
'Performance Analysis of hybrid frequency hopping / direct sequence spread spectrum communication systems with DPSK modulation for the indoor wireless environment'
Graduation thesis
Delft University of Technology
Faculty of Electrical Engineering
19-8-1991

[3] **P.A.D. Spaargaren**
'Tele-information system for train travellers using teletext'
Graduation thesis
Delft University of Technology
Faculty of Electrical Engineering
3-6-1991

[4] **H.J. Wesselman**
'Evaluation of the bit error probability in the microcellular mobile radio system using DPSK modulation'
Taskreport
Delft University of Technology
Faculty of Electrical Engineering
18-2-1991

[5] **A. Bruce Carlsson**
'Communication Systems'
Mac Graw Hill Book Company
1986

[6] **J.H. Weber, D.E. Boekee**
'Foutenverbeterende codes'
Collegediktaat L73e
Delft University of Technology
Faculty of Electrical Engineering

Appendix

In this appendix the Pascal programs are given that were written to calculate the numerical results in this report.

This program calculates the BER as a function of the SNR, with the influence of fading.

```
program BerSnr ;
{$n+}
uses crt ;

{ Calculates the BER, given the SNR, assuming rician fading }

{ DPSK modulation is assumed }

const tol=1.0E-5 ;

var
   sum,
   upper,
   lower,
   factor1,
   factor2,
   factor3,
   snr,
   b,
   k,
   deel: double ;
   A: INTEGER ;
   z: text ;

{********************************}
{* HULP FUNCTIES BESSEL + POWER *}
{********************************}

function pow(x:DOUBLE;l:LONGINT):DOUBLE;
begin
     if x=0 then
       pow:=0
     else
       begin
            if (x<0) and (odd(l)) then
              pow:=-1*exp(l*ln(abs(x)))
            else
              pow:=exp(l*ln(abs(x)))
       end
end;

function bess(y:DOUBLE):DOUBLE;

var t,bess1,bess2,bess3,bess4,bess5:EXTENDED;

begin
     t:=y/3.75;
     if abs(y)<3.75 then
     begin
       bess1:=1+3.5156229*pow(t,2)+3.0899424*pow(t,4)+1.2067492*pow(t,6);
       bess2:=0.2659732*pow(t,8)+0.0360768*pow(t,10)+0.0045813*pow(t,12);
       bess:=bess1+bess2
     end
     else
     begin
       bess1:=0.39894228+0.01328592*pow(t,-1)+0.00225319*pow(t,-2);
```

```
        bess2:=-0.00157565*pow(t,-3)+0.00916281*pow(t,-4)-0.02057706*pow(t,-5);
        bess3:=0.02635537*pow(t,-6)-0.01647633*pow(t,-7)+0.00392377*pow(t,-8);
        bess4:=bess1+bess2;
        bess5:=bess3+bess4;
        bess:=bess5/(sqrt(y)*exp((-1*y)))
      end
end;

{***************************************************}
{* FUNCTIE BESCHRIJVING VOOR DISTRIBUTIE INTEGRAAL *}
{***************************************************}

FUNCTION FX(X: DOUBLE): DOUBLE ;

BEGIN
     FACTOR1 := 0.5*EXP(-X) ;
     DEEL := (K+1) / SNR ;
     FACTOR2 := EXP(-((X*DEEL)+K)) ;
     FACTOR3 := BESS(SQRT(2*X*DEEL)*SQRT(2*K)) ;
     FX := FACTOR1*FACTOR2*FACTOR3*DEEL ;
END ;

{********************************************}
{* INTEGRAAL BEREKENING DISTRIBUTIE FUNCTIE *}
{********************************************}

PROCEDURE INT(LOWER, UPPER, TOL:DOUBLE;
                            VAR SUM:DOUBLE);

VAR Y,DELTA_Y, EVEN_SUM,
    ODD_SUM, END_SUM, SUM1 :DOUBLE;
    PIECES,I               :LONGINT;

BEGIN
    PIECES    :=2;
    I         :=0;
    DELTA_Y   :=(UPPER-LOWER)/PIECES;
    ODD_SUM   :=FX(LOWER+DELTA_Y);
    EVEN_SUM  :=0.0;
    END_SUM   :=FX(LOWER)+FX(UPPER);
    SUM       :=((END_SUM+4.0*ODD_SUM)*DELTA_Y/3.0);
    REPEAT
    BEGIN
        PIECES:=PIECES*2;
        SUM1:=SUM;
        DELTA_Y:=(UPPER-LOWER)/PIECES;
        EVEN_SUM:=EVEN_SUM+ODD_SUM;
        ODD_SUM:=0.0;
        FOR I:=1 TO PIECES DIV 2 DO
        BEGIN
            Y:=LOWER+DELTA_Y*(2.0*I-1.0);
            ODD_SUM:=ODD_SUM+FX(Y);
        END;
        SUM:=((END_SUM+4.0*ODD_SUM+2.0*EVEN_SUM)*DELTA_Y/3.0);
    END;
    UNTIL ABS(SUM-SUM1)<=ABS(TOL*SUM);
END;

BEGIN
     ASSIGN(Z, 'A:BERSNR.TXT') ;
     REWRITE(Z) ;
     CLRSCR ;
     K := EXP(0.1*6.8*LN(10)) ;
     LOWER := 0 ;
     UPPER := 6 ;
     FOR A:= 0 TO 30 DO    { SNR IN DB }
     BEGIN
          SNR := EXP(0.1*A*LN(10)) ;
          INT(LOWER, UPPER, TO1, SUM) ;
          WRITELN(Z,  SUM) ;
```

```
          WRITELN(A:4,'     ',SUM:10:6) ;
     END ;
     CLOSE(Z) ;
     REPEAT
     UNTIL KEYPRESSED ;
END .
```

The next program calculates the BER as a function of the SNR without fading.

```
program BerSnrSimple ;

{ Calculates the BER, given the SNR, for an ideal channel (no fading) }

{ DPSK modulation is assumed }

var
   outfile: text ;
   A: integer ;
   SNR,
   BER: extended ;

begin
     assign(outfile, 'B:BERSNRS.TXT') ;
     rewrite(outfile) ;
     for A := 0 to 30 do
     begin
          SNR := exp(0.1*A*ln(10)) ;   { from dB to normal }
          BER := 0.5*exp(-SNR) ;
          BER := ln(BER)/ln(10) ;       { log 10 }
          writeln(BER) ;
          writeln(outfile, BER) ;
     end ;
     close (outfile) ;
end.
```

This program calculates the BER with FEC, given the BER without FEC.

```
program berfec ;

{ Calculates the BER after FEC coding }

{ DPSK is assumed }

{ uses file BERSNR.TXT as input file for Pe without FEC coding }

var
   infile, outfile: text ;
   m, i, n, k, t: integer ;
   Pe, mu, PeFEC: extended ;

function power(xz, yz : real) : real ;
var
   uu : real ;

begin
     if yz=0 then
     begin
          if xz=0 then power := 0
          else power := 1
     end ;
     if xz=0 then power := 0
     else power := exp(yz*ln(xz)) ;
end ;

begin
     assign(infile, 'B:BERSNR.TXT') ;
     reset(infile) ;
     assign(outfile,'B:BERFEC.TXT') ;
     rewrite(outfile) ;
     writeln('Reed Solomon coding:') ;
     writeln ;
     write('n = ') ;
     readln(n) ;
     write('k = ') ;
     readln(k) ;
     t := (n-k) div 2 ;
     writeln('t = ',t:6) ;
     for i :=  0 to 15 do
     begin
          readln(infile, Pe) ;  { values in infile start at 0 dB, end at 30
dB }
          mu := n*Pe ;
          PeFEC := (-t/mu)/ln(10) ;  { 10log van PeFEC ! }
          writeln(PeFEC) ;
          writeln(outfile, PeFEC) ;

     end ;
     close(infile) ;
     close(outfile) ;
end.
```

The final program calculates the average fade- and non-fade duration

```
Program FadeRice ;

uses WinDos, WinCrt ;

type  fak = array[1..40] of real ;

const                                              snel := 0.5 ;              {
snelheid in km/u }

var
   Ac, m, cros, func1, func2 : extended ;
   pa, eta, b0, b2, d0, d2, rho, krice, fd, a, b, marcum : real ;
   SNR, i, keuze, snrhi, snrlo, freqtel : integer ;
   g, gi, g1 : text ;
   z, fadedur, non_fadedur, fade, nonfade : extended ;
   freq, golf : real;

function power(xz, yz : real) : real ;
var uu : real ;
begin
  if yz=0 then
  begin
    if xz=0 then power := 0
    else power := 1
  end ;
  if xz=0 then power := 0
  else power := exp(yz*ln(xz)) ;
end ;

function Io(y : extended) : extended ;
var t, b : extended ;
begin
  t := y/3.75 ;
  if y<=3.75 then
  begin
    Io := 1 + 3.5156229*sqr(t) + 3.0899424*sqr(sqr(t)) +
          1.2067492*sqr(t)*sqr(sqr(t)) + 0.2659732*sqr(sqr(sqr(t))) +
                0.0360768*sqr(t)*sqr(sqr(sqr(t)))       +
0.0045813*sqr(sqr(t))*sqr(sqr(sqr(t))) ;
  end
  else
  begin
    b := 0.39894228 + 0.01328592/t + 0.00225319/sqr(t) -
         0.00157565/(t*sqr(t)) + 0.00916281/(sqr(sqr(t))) -
         0.02057706/(t*sqr(sqr(t))) + 0.02635537/(sqr(t)*sqr(sqr(t))) -
         0.01647633/(t*sqr(t)*sqr(sqr(t))) + 0.00392377/(sqr(sqr(sqr(t)))) ;
    Io := b/(sqrt(y)*exp(-y)) ;
  end ;
end ;

function Rice(Ac : extended) : extended ;
begin
  Rice := (Ac/b0) * exp(-(sqr(Ac)+1)/(2*b0)) * Io(Ac/b0) ;
end ;

function Q(a, b : real) : real ;
var x, w : array[1..10] of real ;
    i : integer ;
    r : real ;
begin
  x[1] := 0.2453407083009 ;
  x[2] := 0.7374737285454 ;
  x[3] := 1.2340762153953 ;
  x[4] := 1.7385377121166 ;
  x[5] := 2.2549740020893 ;
  x[6] := 2.7888060584281 ;
  x[7] := 3.3478545673832 ;
```

```
  x[8] := 3.9447640401156 ;
  x[9] := 4.6036824495507 ;
  x[10] := 5.3874808900112 ;

  w[1] := 4.622436696006e-1 ;
  w[2] := 2.866755053628e-1 ;
  w[3] := 1.090172060200e-1 ;
  w[4] := 2.481052088746e-2 ;
  w[5] := 3.243773342238e-3 ;
  w[6] := 2.283386360163e-4 ;
  w[7] := 7.802556478532e-6 ;
  w[8] := 1.086069370769e-7 ;
  w[9] := 4.399340992273e-10 ;
  w[10] := 2.229393645534e-13 ;

  r := 0 ;
  for i := 1 to 10 do
  begin
    if x[i] >= b/sqrt(2) then
    r := r + w[i]*2*x[i]*exp(-(sqr(a)+sqr(x[i]))/2) * Io(a*sqrt(2)*x[i]) ;
  end ;
  Q := r ;
end ;

function fac(n : integer) : extended ;
var som : extended ;
begin
  som := 1 ;
  while n>1 do
  begin
    som := som*n ;
    n := n-1 ;
  end ;
  fac := som ;
end ;

function bess(x : real; y : integer) : real ;
var j : integer ;
    sum : real ;
begin
  sum := 0 ;
  for j := 0 to 10 do
  begin
    sum := sum + power((0.5*x),y+2*j)/fac(j)/fac(j+y) ;
  end ;
  bess := sum ;
end ;

function Qf(a, b : real) : real ;
var i : integer ;
    sum : real ;
begin
  sum := 0 ;
  for i := 1 to 10 do
  begin
    sum := sum + power(b/a,i) * bess(a*b,i) ;
  end  ;
  Qf := 1-exp(-(a*a+b*b)/2)*sum ;
end ;

{ hoofdprogramma }

begin
  assign(g, 'A:fadedur.txt') ;
  rewrite(g) ;
  assign(g1, 'a:nofaddur.txt') ;
  rewrite(g1) ;
  clrscr ;

  writeln ('Fade duration as a function of   1. SNR');
```

```
  writeln ('                              or   2. Frequency');
  readln (keuze);
  while (keuze <> 1) and (keuze <> 2) do readln (keuze);

  krice := power(10,0.68) ; { rice factor }
  b0 := 1/(2*krice) ;
  a := sqrt(2*krice) ;
  if keuze = 1 then
  begin
    writeln ('Enter the fixed frequency in MHz:') ;
    readln (freq) ;
    golf := 300/freq;
    fd := snel/golf/3.6 ;
    b2 := b0*sqr(2*pi*fd) ;
    writeln ('Enter the range of the SNR:') ;
    write ('Lower bound of SNR : ') ;
    readln(snrlo) ;
    write ('Higher bound of SNR : ') ;
    readln (snrhi) ;

    for SNR := snrlo to snrhi do
    begin
     pa := power(10,SNR/10) ;
     b := sqrt(2)*pa ;
     if b>a then
     writeln('b is groter dan a!') ;
       func1 := sqrt(b2/(2*pi)) ;
     func2 := rice(pa) ;
     cros := func1 * func2 ;
     marcum := Qf(a,b) ;
     fadedur := (1-marcum)/cros ;
     non_fadedur := marcum/cros ;

     writeln ('SNR = ',SNR:3,' fadedur = ', fadedur:12, '   nonfadedur = ',
non_fadedur:12) ;
     writeln(g, fadedur:10) ;
     writeln(g1, non_fadedur:10) ;
    end ;
  end

  else
  begin
    writeln ('Enter the fixed value for the SNR :') ;
    readln(SNR) ;
    writeln ('The frequency range is 20-60 GHz.') ;

    for freqtel:= 20 to 60 do
    begin
     pa := power(10,SNR/10) ;
     b := sqrt(2)*pa ;
     if b > a then writeln ('b is groter dan a!') ;

     freq := 1000.0*freqtel ;
     golf := 300.0/freq ;
     fd := snel/golf/3.6 ;
     b2 := b0*sqr(2*pi*fd) ;

     func1 := sqrt(b2/(2*pi)) ;
     func2 := rice(pa) ;
     cros := func1 * func2 ;
     marcum := Qf(a,b) ;
     fadedur := (1-marcum)/cros ;
     non_fadedur := marcum/cros ;
     writeln('Freq  =  ',freqtel:2,'  GHz      fadedur  =  ',  fadedur:12,  '
nonfadedur = ', non_fadedur:12) ;
     writeln(g, fadedur:10) ;
     writeln(g1, non_fadedur:10) ;
    end;
  end;
```

```
  close(g) ;
  close(g1) ;
end.
```