Successful Communication Over a Wireless Link Under the Presence of Other Users

Ilias Karampatsos

MSc Report

Committee:
Prof.dr.ir. C.H. Slump
Ing. B.A. Witvliet
Dr.ir. R. Schiphorst
Dr.ir. M.J.Bentum
Dr.ir. J.F.Broenink

Aug 2014

Report nr. 012RAM2014
Robotics and Mechatronics
EE-Math-CS
University of Twente
P.O. Box 217
7500 AE Enschede
The Netherlands

UNIVERSITY OF TWENTE

Abstract

Department of Electrical Engineering, Mathematics and Computer Science

Master of Science

Successful Communication Over a Wireless Link Under the Presence of Other Users

by Ilias Karampatsos

The goal of this Master Thesis was to examine the impact of interference on wireless communication under the scope of throughput efficiency. The research focused on two domains. The first domain was the coexistence of wireless systems in the Licence Excempt (LE) domain. The sensitivity of QPSK, 16 QAM, and GFSK modulation systems against interference was investigated. Such measurements can be used to optimize the Medium Usage(ME) by reducing the spatial overlap of interfering systems. The interference of a multicarrier system (802.11g) and a frequency hopping system (Bluetooth) was evaluated through Simulink. The simulations were performed in a flat fading environment.The second domain of investigation was the comparison of Convolutional Coding (CC), Interleaving, and Hard Decoding(HD) versus the Opportunistic Error Correction (OEC) scheme under narrowband interference. The superiority of the OEC scheme, under constant frequency narrowband interferer and under flat and multipath fading was concluded. Finally, the performance of these schemes versus a Bluetooth jammer signal was investigated through measurements. The measurements were inconclusive about which system is superior, under the same effective throughput requirement. Nevertheless, they showed the ability of the OEC scheme to provide successful communication under Bluetooth interference at a constant frequency.

Acknowledgements

Now that my Master Thesis project has finished I would like to express my gratitude to several people that helped me a lot during this hard but also scientifically fruitful procedure. Through this procedure I had the chance to work on different subjects, and I obtained useful skills on the theoretical analysis of systems, as well as skills on the implementation of these communication systems. First of all, I would like to thank Prof. Dr. Ir. Kees Slump for giving me the opportunity to work on this topic, through which I became a better scientist and engineer. Moreover I would like to thank him for the patience that he showed and for giving me the opportunity to reach the goals I had set for this thesis.

Moreover I would like to thank my supervisor Ir. Ben Witvliet for guiding me through the first part of this thesis. His efforts for inspiring me on the field of dynamic spectrum management, motivating me, as well as guiding me as a Senior engineer are deeply appreciated. Thanks to him I had the opportunity to broaden my scientific and engineering horizons. I would also like to thank my other supervisor Dr. Ir. Xiaoying Shao for her help and support through this procedure. I would like to thank her specifically for enhancing my research abilities and for giving me the opportunity to go through the research chain steps. These steps involve the generation of an idea, the implementation, the test of the system's behavior and the test of the idea through simulations and measurements.

Apart from my supervisors I would like to thank the secretaries Sandra Westhoff and Jolanda Boelema, for helping me integrate in the group.Thanks to their efforts, the time that I spent with the group was really enjoyable and heartwarming. This thesis could not have been completed without the help of Gertjan and Henny. I would specially like to thank them for helping me with the measurement setup and for trying to provide me with the necessary facilities in order to be able to work undistracted.

Finally, I would like to thank my family for their continuous financial and moral support, without which I could not have done what I did. Last but not least, I would like to thank all the friends in Greece and in Enschede for their support and encouragement. Babis, Alexandros , Dimitris, Renos, Kostas, Xristos, Athina, Filippos, Sherry, Dimitris Vlaxos, Paulina, Makis, Pantelis, John Kostakos, Dimitris Kordas, Laura, Amina.

Contents

List of Figures

List of Tables

List of Acronyms

- **AWGN** Additive white gaussian noise
- **BCC** Binary symmetric channel
- **BER** Bit error rate
- **BPSK** Binary phase shift keying
- **CC** Convolutional coding
- **CFO** Constant frequency offset
- **CP** Cyclic prefix
- **CPM** Continuous phase modulation
- **CRC** Cyclic redundancy check
- **DSSS** Direct sequence spread spectrum
- **ECC** Error correction code
- **FDM** Frequency division multiplexing
- **FEC** Forward error correction
- **FFT** Fast Fourier transformation
- **GE** Gaussian elimination
- **GFSK** Gaussian frequency shift keying
- **HD** Hard decoding
- **HDD** Hard decision decoder
- **ICI** Inter carrier interference
- **IFFT** Inverse Fast Fourier transformation
- **ISM** Industrial, scientific and medical
- **LDPC** Low density parity check
- **LE** Licence excempt
- **LT** Luby transform
- **MAC** Medium access control

- **MAP** Maximum a posteriori
- **MMSE** Minimum mean square error
- **MP** Message passing
- **OEC** Opportunistic error correction
- **OFDM** Orthogonal frequency division multiplexing
- **OSI** Open systems interconnection
- **PER** Packet error rate
- **PM** Path metric
- **PR** Protection ratio
- **QAM** Quadrature amplitude modulation
- **QPSK** Quadrature phase shift keying
- **RF** Radio frequency
- **SINAD** Signal-to-noise and distortion
- **SIR** Signal-to-interference ratio
- **SNR** Signal-to-noise ratio
- **TDD** Time division duplex
- **WLAN** Wireless local area network
- **ZF** Zero forcing

Symbols

B	Bandwidth
Bd	Baud rate
E_b	Bit energy
E_s	Symbol energy
Fs	Sampling frequency
I	Interference power
I_o	Modified Bessel function of the first kind of order zero
K	Fountain source packets
M	Mapping index
N	Fountain encoder output size
N'	Received Fountain packets
Np	Noise power
Ns	Number of OFDM subcarriers
Nd	Number of OFDM discarded subcarriers
Nds	Number of OFDM data subcarriers
Np	Number of OFDM pilot subcarriers
Nz	Number of OFDM null subcarriers
Nov	Minimum number of Fountain packets which can provide successful decoding
N_{BPSC}	Number of coded bits per subcarrier
N_{CBPS}	Number of coded bits per OFDM symbol
N_o	AWGN power spectral density
Q	Number of available narrowband channels
R	Bit rate
Rc	Code rate
R_{FC}	Fountain code rate
R_{OEC}	Code rate of the opportunistic error correction scheme
R_{OEC}	Effective code rate
S	Signal power
T	Symbol rate

Ts	Sampling period
Tcp	802.11g cyclic prefix duration
Td	802.11g OFDM data duration
T_{hop}	Hopping time
T_h	Uncoded system throughput
Q	Marcum Q function
W	Bluetooth transmission bandwidth
X	OFDM IFFT input vector
Y_{est}	Equalized 802.11g symbol
d_f	Introduced constant frequency offset for testing
f_k	Center frequency of the k-th subcarrier
f_d	GFSK frequency deviation
f_c	RF carrier frequency
h_m	GFSK modulation index
$\hat{a}$	Transmitted training sequence
$\hat{d}_f$	Estimated frequency offset from the short preamble correlation
$\hat{x_r}$	Time domain 802.11g received signal
$\hat{X_r}$	Frequency domain 802.11g received signal
k_p	Data bits per Fountain packet
n	Number of samples
n_o	AWGN signal with zero mean and unity variance
n_p	Number of coded bits per packet
nb	Bits per baud
n_t	Used AWGN signal for the BER computations
p_n	Pilot tone sequence
p_{qpsk}	Error probability of the QPSK mapped symbols
p_{16QAM}	Error probability of the 16QAM mapped symbols
r	Roll off factor
x	OFDM IFFT output vector
x_p	Upsampled 802.11g signal
x_s	OFDM symbol with cyclic prefix
x_{RF}	Transmitted bandpass 802.11g signal
$x_{RF'}$	Received bandpass 802.11g signal
x_r	Downsampled 802.11g signal
ρ	GFSK receiver correlation coefficient
ρ_{ML}	Correlation coefficient of the ML algorithm
α	Percentage of the available bandwidth which is occupied
γ	Ratio of the interferer bandwidth over the signal bandwidth
γ_p	Estimated phase offset

ε	Introduced constant frequency offset
$\hat{\varepsilon}_{ML}$	Estimated constant frequency offset according to the ML algorithm
θ	Time delay of the received signal
$\hat{\theta}_{ML}$	Estimated time delay according to the ML algorithm
σ	Standard deviation of n samples
$\tau_{\frac{a}{2}}$	Critical value from the Student t distribution
τ	Modified Thompson tau
Δ_f	Subcarrier spacing

Dedicated to my family and friends

Chapter 1

Introduction

1.1 General Introduction

Wireless and wired communications have played a significant role in our society over the last decades. The amount of data that can be transferred through them has been constantly increasing. The influence of the wireless communications is huge, since wireless devices allow their users to communicate while being mobile. Although the initial wireless communication was limited to the use of wireless phones, the need for mobility as well as technological progress lead to an even bigger use of wireless systems. Alas, the medium that can be used for wireless transmission imposes several limits and introduces drawbacks. The frequency bands that are used for transmission are already predefined, and the available bandwidth is finite. Since the frequency range is predefined, many wireless systems have to coexist at the same frequency range. An example of wireless systems that operate at the same frequency range are the 802.11 b/g/n systems [2], as well as the Bluetooth system [3]. These communication systems operate at the 2.4 GHz range and they can be found on laptops, tablets etc. The coexistence of multiple systems at the same frequency band introduces man-made interference which deteriorates the reliability of the communication. Apart from the man-made interference, the nature of the wireless environment introduces time-varying multipath propagation. The reliable communication of wireless systems depends on the minimization of the effects of multipath propagation and the minimization of man-made interference.

Minimization of manmade interference is a really difficult subject, and it is the Achilles heel of wireless communications. The main reason for making the minimization of manmade interference so difficult and yet so important is the variety of wireless systems which operate at the same frequency band. There are several approaches for dealing with man-made interference which falls at the desired bandwidth. A description of these approaches is given at the following subsection.

1.2 Aims of This Master Thesis

1.2.1 Research Framework

The methods that are used for dealing with interference can be distinguished into two basic categories. The main difference of these approaches is the layer of the OSI protocol[4] that deals with the minimization of the interference. The first category deals with interference through MAC and physical layer mechanisms. The first methodology for dealing with interference through MAC layer is dynamic spectrum management[5]. An application of dynamic spectrum management is cognitive radio[6]. The purpose of cognitive radio and dynamic spectrum management is to optimize the channel utilization through spectrum sensing. The transceiver performs spectrum sensing and chooses to operate at a licenced band that has minimal or no spectral utilization. Although the initial purpose of cognitive radio and dynamic spectrum sensing is not to minimize interference, the choice for transmission of spectrum bands that are free also aids to that direction. This form of interference mitigation mechanism can be considered to be preventive.

The second technique that can be used in order to achieve an acceptable throughput in the presence of interference involves coding of the transmitted data at the physical layer of the OSI protocol. Coding is a process that introduces redundancy. This redundancy can be used by the decoder which is present at the receiver in order to correct errors in the received bit sequence. . Characteristic examples of codes are the Convolutional codes, the Hamming codes[7], the Turbo codes[8], and the Low-Density Parity-Check (LDPC) codes[9].

The final method for dealing with interference is beamforming[10]. Beamforming is an interference mitigation method that takes place at the physical layer of the OSI protocol. Beamforming is a signal processing technique that requires more than one receive antennas and a processor in order to provide spatial filtering. The spatial diversity of the receive antennas and the known distance between them gives the ability to estimate the angle of arrival of the electromagnetic wave. Afterwards, signal processing algorithms such as the Minimum Mean Square Error (MMSE) algorithm can be used to minimize the gain of the receive antennas at specific directions of arrival. In such a way, if the interference possesses some spatial correlation, it can be minimized. Although the use of beamforming can decrease interference substantially, the use of multiple receive antennas increases the consumed power.

1.2.2 Motivation for this Master Thesis

In the previous subsection the basic methodologies for dealing with interference were presented. These methodologies are not confined to a specific frequency range or system. An interesting area of research is the Licence Exempt (LE) spectrum and this is the area that this Master Thesis will focus. As the title of this thesis declares, the topic is to investigate ways to improve the throughput, on the presence of interference. The motivation for the first part of this thesis derives from the PhD project Spectrum Access Mechanisms for Licence Exempt Radio Systems which is conducted at the University of Twente by Ir. Ben Witvliet. The purpose of this project is to develop a spectrum access mechanism for heterogeneous systems operating at the LE spectrum. This mechanism aims to provide equal spectrum sharing between dissimilar systems and to improve the collective spectrum efficiency[11], [12]. In order to achieve these, the developped spectrum access mechanism should be aware of the sensitivity of the various modulation types under interference. By consequence, if a system is aware of the other systems transmitting in the same area, it can modify its transmit power accordingly in order to reduce the spatial overlap of its interference footprint. Since some systems have predefined levels of output power, if a system cannot modify the output power, it can take other actions according to [11] in order to optimize the use of the spectrum.

The sensitivity of the various modulation systems can be depicted at a matrix, which is called interference matrix. The elements of the interference matrix are called Protection Ratio (PR) values, and correspond to the ratio of the average system power over the average interference power, which can lead to a specific BER, in decibels. The average power for the calculation of the PR is the average power per MHz, thus the PR can be also characterized as the ratio of the system power spectral density over the interferer power spectral density, in decibels. The first goal of this thesis is to create such an interference matrix for a multicarrier and a singlecarrier system. The multicarrier system is 802.11g under OFDM and the singlecarrier is the Bluetooth system.

The motivation for the second part of this thesis comes from the work of Dr. Ir. Xiaoying Shao during her PhD a the University of Twente. The research area is channel coding and the topic of research is the improvement in the throughput that can be achieved with the use of Fountain coding along with LDPC and Cyclic Redundancy Check (CRC) codes. The examined topology is the Opportunistic Error Correction (OEC) scheme [13], [14] and the purpose is to evaluate whether there is an improvement in the throughput of the transmission of 802.11g system that uses the OEC on the presence of an narrowband interferer. The OEC scheme is compared to the scheme that uses Convolutional Coding(CC) and Interleaving. This part of the thesis approaches the topic of interference minimization purely at the physical layer. The second goal of this master thesis is to evaluate the behavior of the OEC and CC-Interleaving on the presence of narrowband interference through simulations and hardware measurements. Since the CC

scheme with interleaving requires retransmissions of data in case the system performance is very poor, the purpose is to examine whether with the opportunistic error correction scheme these retransmissions can be avoided.In that case, there are two benefits. The first one is that the throughput of the system can be improved and the second one is that if retransmissions can be avoided,there is no congestion to the medium and by consequence there is less interference to the other systems.

1.2.3 Research Questions

Based on the previously mentioned motivation, the main research question that has been formulated is:
How can we improve the throughput of a wireless system under the presence of other wireless interfering systems?

The main research question can be answered with the use of the following secondary questions.

- Which are the maximum interferer power levels that can be tolerated by the 802.11g and Bluetooth systems in order to achieve a raw bit error rate of 0.1%. The specific BER threshold is the Bluetooth receiver sensitivity level [3].
- Which scheme provides better communication throughput under narrowband interference, the Opportunistic Error Correction Scheme or a combination of Forward Error Correction and Interleaving?
- Can the superiority of one of the above schemes be verified through measurements, in case of a Bluetooth interferer and a fading environment?

1.3 Used Methodology

The first part of this thesis involves bandpass simulations in order to obtain the interference matrix. The rows of the interference matrix are the wanted transceiver systems and the columns of the matrix are the interferer systems. The elements of the matrix contain the Protection Ratio (PR) value which defines the sensitivity of each system to interference.

The two evaluated systems, 802.11g and Bluetooth have been modelled with the help of Matlab's Simulink. The baseband models were converted to bandpass since the measurements were performed in bandpass. For the calculation of the interference matrix, the examined systems did not include coding. The reason for doing that is that the interference matrix should mainly depend on the modulation of the system, as well as on the wideband or narrowband nature of the

system. Initially, the interference matrix is calculated assuming an ideal environment without noise or fading.

The second part of this thesis involves baseband simulations of the 802.11g system. For these simulations two versions of the 802.11g system have been modeled. The first version, version a, used convolutional coding and interleaving and the second version, version b, used Fountain coding as well as LDPC and CRC coding. A narrowband interference signal with variable bandwidth and spectral density has been also created. The performance of versions a and b over the narrowband interferer was evaluated first at ideal conditions and afterwards under a multipath channel with SNR=20 dB. The measurements were conducted for different interferer bandwidth and fixed Signal-to-Interference ratio (SIR), equal to zero. The measured quantity was the number of wrong received information bits.

The third part involves the hardware measurements that were performed inside the laboratory of the Signals and Systems group. Versions a and b of the 802.11g system were interfered by a narrowband signal which had the spectral characteristics of a Bluetooth signal.

1.4 Outline

Chapter 2 includes a short theoretical analysis of the multicarrier and single carrier systems. Moreover, chapter 2 presents the physical layer of the two systems that have been investigated. Specifically, it contains the implementation of the physical layer of the 802.11g system as well as the implementation of the Bluetooth system. The implementation of the physical layer has a big impact on the behavior of the system versus errors and affects the spectrum characteristics of the transmitted signal. The main reason for performing the simulations mainly at the physical layer, is that we are interested in dealing with interference below the MAC layer.

Chapter 3 describes the procedure that has been followed for the Protection Ratio simulations and the measurements which correspond in an ideal scenario. The last part of this chapter includes the analysis of the simulation results.

Chapter 4 begins with a description of the two versions of the 802.11g that has been used to simulate the suppression of narrowband interference. The first version has FEC and interleaving and the second version uses the OEC scheme. The second part of chapter 4 presents the methodology that has been followed for the interference suppression measurements and the analysis of the results for an ideal channel without fading.

Chapter 5 presents the experimental topology for the hardware measurements. The hardware as well as the wireless environment introduce further distortions to the signal. In order to overcome these distortions the 802.11g system of chapter 5 has to be extended. The new blocks that perform timing estimation and frequency compensation are also analyzed at this chapter. The next part of chapter 5 introduces the methodology that has been used for the hardware measurements, and the last part presents the results from the measurements that have been conducted inside the laboratory of Robotics and Mechatronics.

Finally, the conclusions of this master thesis are depicted in chapter 6 along with recommendations for future work.

Chapter 2

Physical Layer of the Investigated Systems, 802.11g and Bluetooth

2.1 Theoretical Background

2.1.1 Multicarrier Systems

As it was already stated in the introduction, the available bandwidth for transmission is limited. Several methods for improving the efficiency of wireless transmission have been developped. One of the first methods, involved splitting the available bandwidth into a number of subchannels. These subchannels can be used by multiple users concurrently. This approach is called Frequency Division Multiplexing (FDM). Each one of these subchannels can be assigned to a subcarrier. Alas, in the case that the subcarriers are spaced closely to each other, they suffer from Inter-Carrier Interference (ICI) because frequency components of one subchannel might fall into a adjacent band. This problem can be also solved by establishing guard bands between the subcarriers and diminish the effects of ICI but in such case, these guard bands would be a waste of the available bandwidth. Figure 2.1 depicts a possible use of the the spectrum with FDM and guard bands.

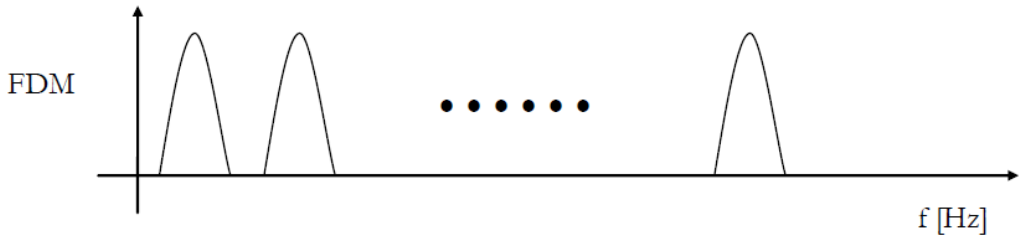


FIGURE 2.1: Subchannel allocation according to FDM with guard bands between the carriers. In order to avoid adjacent channel interference, two sequential carriers, are separated by a guard band. As far as this figure is concerned, three carriers are depicted with the appropriate guard bands. An example of a guard band is the unused spectrum between the first two carriers.

The above mentioned problems of ICI and inefficient use of bandwidth can be solved with the use of Orthogonal Frequency Division Multiplexing(OFDM). OFDM uses the same principle of dividing the available bandwidth in subchannels similar to FDM. The difference of OFDM is that the subchannels are assigned to subcarriers that are orthogonal to each other. This is achieved by assigning to the subcarriers sinusoids with frequencies that are of the form [15] :

$$s_k = cos(2\pi\, f_k t), \quad k = 0, 1, 2, ..., N-1 \tag{2.1}$$

where N is the number of subchannels on which the available bandwidth is split, and f_k is the mid frequency of the k_{th} subcarrier. If the symbol rate T, is equal to the spacing Δ_f between the subcarriers $f_{k+1} - f_k$, then according to the orthogonality principle, the subcarriers are orthogonal between the time intervals $[mT, (m+1)T]$, $m = 0, 1, 2, ...$

The spectrum of an OFDM signal is shown at figure2.2

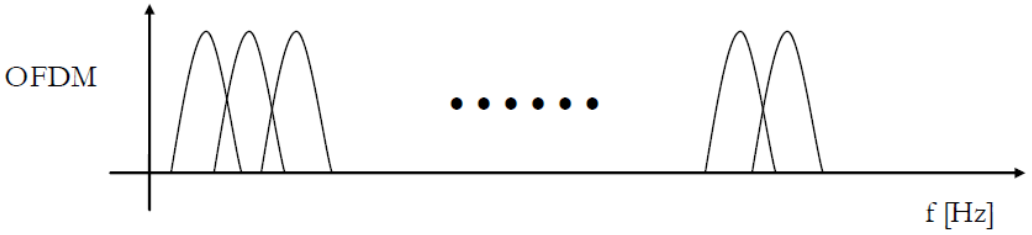


FIGURE 2.2: Subchannel allocation according to OFDM. Due to the orthogonality of the subcarriers, no guard band is needed. In this figure, a subcarrier overlaps the neighboring subcarriers without causing ICI.

With OFDM the efficient use of the available bandwidth is maximized since no guard band are required and as it can be seen from figure 2.2 the created subchannels can even overlap each other without ICI problems. The input of each subcarrier can be symbols that follow different modulations. The only requirement is that the symbol rate is equal to $1/\Delta_f$.

2.1.2 Frequency Hopping - Spread Spectrum

Frequency hopping is one of two basic modulation techniques used in spread spectrum signal transmission. The other modulation technique is Direct Sequence Spread Spectrum (DSSS). As the term frequency hopping implies, the wireless communication frequency of such a system, changes over time. The second term, spread spectrum, denotes that in order to be able to achieve variable transmission frequency, the transceivers should be able to provide a much higher operation bandwidth. Thus, although the bandwidth of the transmitted signal is usually narrowband, the total bandwidth that is required for transmission is usually much higher. The pseudorandom changes of the used frequencies randomize the medium occupancy, and by consequence they allow multiple access over a wide range of frequencies. Thus, frequency hopping is a way to avoid interference in a congested medium. The total available bandwidth for transmission, W, is divided into Q narrow bands, which have an effective bandwidth of $B = \frac{W}{Q}$ [16]. Bandwidth B is called instantaneous bandwidth while bandwidth W is called total hopping bandwidth. Bandwidth B corresponds to a symbol rate of T, while T_{hop} is the time between hops and it is called hopping period or time slot.

There are two types of frequency hopping, depending on the relation of T and T_{hop}. If $T > T_{hop}$, then frequency hopping is called slow frequency hopping. During each hopping period multiple symbols can be transmitted. On the other hand, if $T < T_{hop}$, the system is said to perform fast frequency hopping.

Figure 2.3, shows an example of a system that employs slow frequency hopping transmission scheme. In this example $T_{hop} = \frac{T}{4}$, and during one hop period, 4 symbols are transmitted. A possible transmission by another system at the same frequency channel, could affect many symbols. The same thing could also happen if the communication channel presents high attenuation at this specific frequency. The Frequency Hopping Spread Spectrum (FHSS) system that will be evaluated is the slow frequency hopping system Bluetooth. The specific implementation of Bluetooth, is without coding, adaptive frequency hopping or adaptive equalization according to the requirements of [11]. By consequence, possible interference or strong fading could cause a burst of errors.

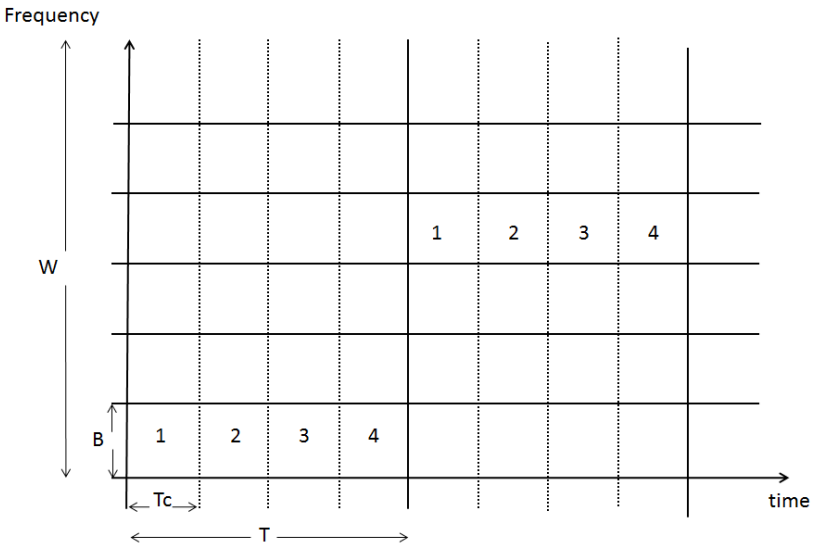


FIGURE 2.3: Slow frequency hopping example.

2.2 802.11g Transmitter

802.11g [2] is an example of a typical multicarrier system, since it can support transmission through OFDM modulation. Apart from the OFDM modulation, the 802.11g standard also supports Direct Sequence Spread Spectrum (DSSS) transmission. It was chosen as a representative multicarrier system, since it is operating at the frequency range of 2.4 GHz, where there are many transmitting systems, and it is interesting to investigate the amount of interference power it can tolerate, while maintaining a desired quality of service. Moreover, the great expansion of the 802.11g at the market increases the importance of investigating such a system.

In this subsection, the basic parts of the 802.11g transmitter are be analyzed. Figure 2.4 depicts the basic blocks of the 802.11g transmitter and the corresponding signal notation.

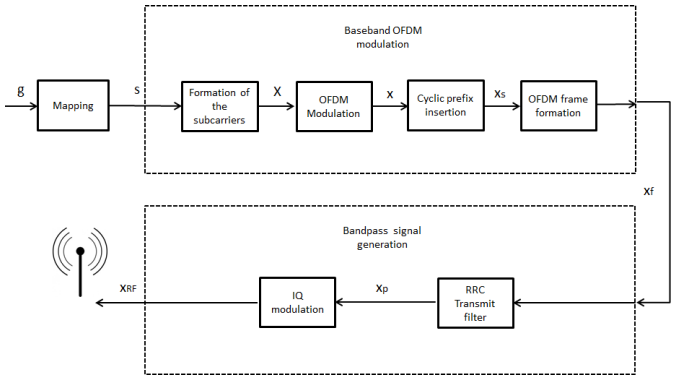


FIGURE 2.4: 802.11g transmitter schematic. The randomly generated input bits are first mapped into QPSK and 16 QAM symbols. These symbols are the inputs of the OFDM data subcarriers. Further on, pilot subcarriers are added to the data subcarriers, as well as zero padded subcarriers. The result, is a total amount of 64 subcarriers which are subsequently OFDM modulated. The last 16 samples of an OFDM symbol are used as a cyclic prefix and are added at the beginning of the OFDM symbol. The next step is the formation of the OFDM frame which involves 20 OFDM symbols, ten short training symbols and two long training symbols. The final step is the upsampling of the baseband signal, pulse shaping and the generation of the bandpass signal through IQ modulation.

2.2.1 Mapping

For the first part of this master thesis, the created data bits will not undergo any coding or other mechanisms that help to recover wrongly received bits. By consequence the first block that is examined is the block that performs the mapping of the data bits into modulated symbols. The information bits that have to be transmitted are in binary format and they can be either 0 or 1. In order to have more efficient communication as far as the number of transmitted bits per time unit is concerned, each information bit is mapped with the use of M-ary modulation into one of 2^M possible symbols. Each symbol is a sinusoid whose amplitude or phase can have have one of M-possible values. Thus, each transmitted symbol, contains $\log_2 M$ information bits.

Since 802.11g can support a variety of modulation types and it can change the modulation during the transmission, 2 different versions of the 802.11g system have been examined. The two versions of the 802.11g system that have been used for the simulations use QPSK modulated data and 16 QAM modulated data. The actual modulation in the simulation model was performed via the Simulink blocks QPSK Modulator Baseband, Rectangular QAM Modulator Baseband. These blocks actually perform the mapping according to the Gray encoding of the input bits into complex numbers. The exact settings for the modulation blocks are presented in Appendix A. Figures 2.5 and 2.6 show the constellations of the QPSK and 16 QAM modulated symbols.

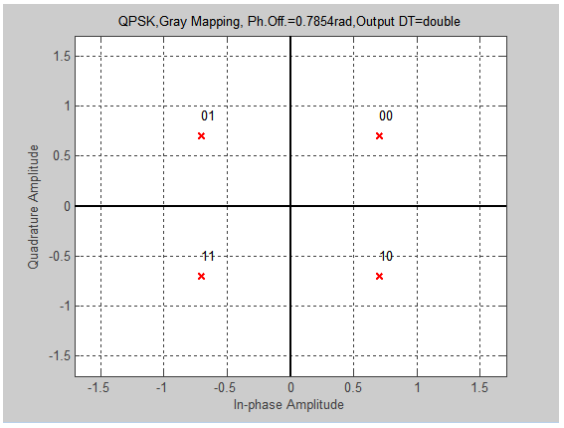


FIGURE 2.5: QPSK Constellation.

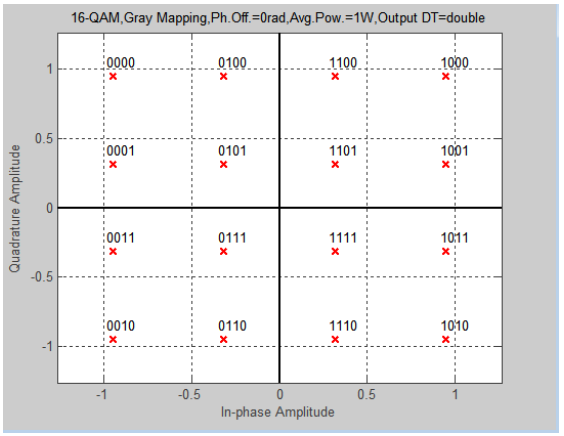


FIGURE 2.6: 16QAM Constellation.

2.2.2 OFDM Modulation

The complex numbers which represent the modulated information bits, have to be assigned to subcarriers and OFDM modulated. This process is performed in the following four stages:

- Formation of the subcarrier inputs
- OFDM Modulation
- Cyclic Prefix insertion
- Formation of the OFDM frame

The process that is followed at each one of these stages is going to be presented in the following subsections. The parameters of OFDM modulation according to the 802.11g standard are illustrated at table 2.1.

Formation of the subcarrier inputs

According to the 802.11g standard, the 64 OFDM subcarriers do not contain only modulated

TABLE 2.1: 802.11g parameters

OFDM Parameter	Symbol	Value
Sampling Frequency	Fs	20 MHz
OFDM Bandwidth	B	20 MHz
OFDM Sampling Period	Ts	50 μs
Total Subcarriers	Ns	64
Data Subcarriers	Nds	48
Pilot Subcarriers	Np	4
Null Subcarriers	Nz	12
Subcarrier Spacing	Δ_f	0.3125 MHz
OFDM Symbol Duration	$Tsymbol$	4μs
Cyclic Prefix Duration	Tcp	0.8μs
OFDM Data Duration	Td	3.2μs

information data. Some of them have modulated data, some are zero padded and some others are modulated by specific sequences. These sequences known as pilot tones, are inserted to specific subcarriers, and give the receiver information about the channel. The pilot tones for the 802.11g system are generated by a generator polynomial. The output of the generator polynomial is unipolar and it contains the values '1' and '0'. These values have to be converted to bipolar representation, where the "1" is replaced by "-1" and the "0" value is replaced by "1". The pilot tone sequence that was used is:

$$p_n = \{1, 1, 1, 1, -1, -1, -1, 1, -1 - 1 - 1 - 1, ..., 1, -1, -1, 1\} \tag{2.2}$$

For each OFDM symbol one value from the p_n sequence was used. The process of formating the subcarrier inputs was performed at two phases. During the first phase a vector with 64 elements, X_t, which consists of data, pilot tones and zeros was formed. An example of such a vector is shown at figure 2.7. The parameters, nd and nz, correspond to the number of data symbols and to the number of zeros. The sequence with which X_t is filled with modulated data is from the left to the right.

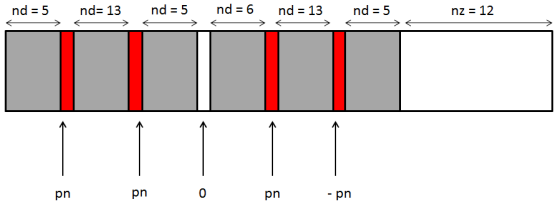


FIGURE 2.7: Format of the X_t vector. The elements of X_t are then rearranged and the last 38 elements are placed at the beginning. The reason for doing that, is that after the IFFT, the zero padding should be situated at the edges of the OFDM spectrum. With that way the edges of the OFDM spectrum act as guard bands.

The 64 elements of the X_t vector have to be rearranged before the Inverse Fast Fourier Transformation (IFFT). The purpose of reshaping the X_t vector is to obtain a double sided spectrum after the IFFT. The result of this reshape is the X vector which is the input vector of the IFFT. The format of the X vector is the following.

$$X = [\ X_t(27),\ ...,\ X_t(64),\ X_t(1),\ X_t(2),\ ...,\ X_t(26)\ ] \tag{2.3}$$

Although the creation of Ns subcarriers with orthogonal frequencies would require Ns oscillators, the actual implementation of OFDM is simplified with the use of IFFT for the OFDM modulation, and of the Fast Fourier Transformation (FFT) for the OFDM demodulation [17]. Normally IFFT accepts as input the frequency domain representation of a signal, and its output is the time domain representation of the signal. The output of the IFFT is given by the following equation[18]:

$$x(l) = \frac{1}{\sqrt{Ns}} \sum_{n=0}^{Ns-1} X(l) e^{j\frac{2k\pi n}{Ns}}, \tag{2.4}$$

where Ns is the number of the subcarriers, $X(l)$ is the input symbol of each subcarrier and l is the subcarrier index. Although the above equation requires that the coefficient $X(l)$ is the representation in the frequency domain, in practice the inputs $X(l)$ are time domain signals. With this mathematical expression, the creation of the subcarrier harmonic frequencies, their modulation and the addition of the signals in the time domain is done with the IFFT algorithm. The length, of the IFFT output is equal to the number of the IFFT points and according to the 802.11g standard the IFFT length is 64. After the OFDM modulation the output of the IFFT has to be also multiplied by the factor $\frac{64}{\sqrt{52}}$ in order to preserve the energy of the data symbols. Since 12 of the 64 subcarriers are zero and the IFFT output should be normalized, the energy of the data subcarriers is also distributed to the zero subcarriers. With this normalization factor, the initial energy of the data subcarriers is preserved.

Cyclic Prefix insertion

The next step towards the creation of the OFDM transmitted symbol is the insertion of the cyclic prefix. The cyclic prefix is the concatenation of OFDM subcarriers which are copied at the beginning of the OFDM symbol [19]. According to the 802.11g standard, if $[x(1), ..., x(64)]$ are the OFDM subcarriers at the output of the IFFT block, the cyclic prefix consists of the OFDM

subcarriers $[x(48),\ x(49)\ , ..., x(64)]$. The transmitted OFDM symbol then has the following form:

$$[x(48),\ x(49)\ , ..., x(64), x(1), ..., x(64)] \tag{2.5}$$

The time duration of the cyclic prefix is $16T_s$=0.8μs, and the total duration of the OFDM symbol is $80T_s$=4μs. Figure 2.8 shows the timing relations inside an OFDM symbol.

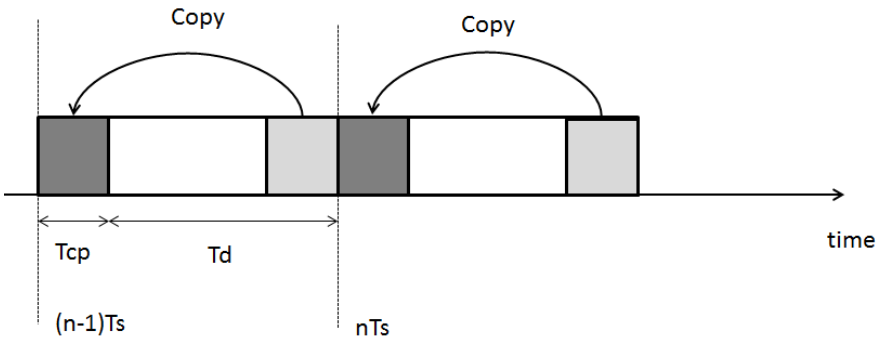


FIGURE 2.8: OFDM symbol with cyclic prefix. The cyclic prefix is a copy of the last 16 samples of each OFDM symbol which is added at the beginning of the OFDM symbol.

Although the use of a cyclic prefix can be seen as a waste of transmitted power, its use has two major benefits. The first benefit is that with the use of the cyclic prefix, the intersymbol interference is minimized. The second purpose is that it allows the effects of a multipath channel to be modelled as a circular convolution instead of a linear convolution. At the receiver, the use of FFT for the OFDM demodulation transforms the data into frequency domain. According to the properties of the DFT, the circular convolution in time domain corresponds to the multiplication signals in the frequency domain. The channel equalization can be performed easily in the frequency domain.

Burst Generation

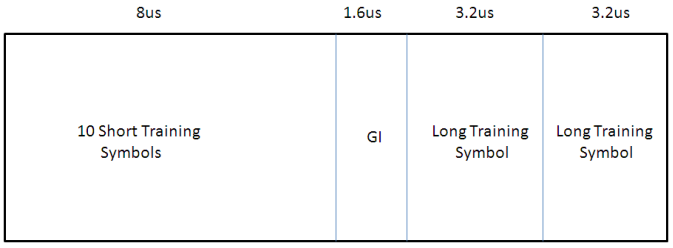


FIGURE 2.9: Transmitted OFDM frame. Each frame consists of 10 short training symbols of 16 samples each. The total duration of the short preamble is 8us. In addition to the short preamble, an OFDM frame consists of a long preamble too. The long preamble consists two long training symbols, each one of which has 52 samples and has 3.2us duration. The last 16 samples of the long training sequence are used as a cyclic prefix. For the formation of the OFDM frame, two cyclic prefixes are inserted between the short preamble and the long preamble. These prefixes have a total duration of 1.6us.

Layers that are higher in the OSI hierarchy than the physical layer, require additional data before the actual transmission of the OFDM symbols. Specifically, the MAC layer requires additional information that helps to synchronize the receiver, estimate the channel and eliminate any transceiver imbalances of the two communicating nodes. Although the purpose of this master thesis is not investigate the behavior of 802.11g system at the MAC layer, some of these additional transmitted data are useful to us. The created OFDM data symbols are transmitted through OFDM frames. An OFDM frame includes several preamble sequences and a number of OFDM data symbols. There are two types of preamble sequences according to the 802.11g standard. The short training sequence and the long training sequence. The short training sequence helps to perform frame detection, carrier frequency offset compensation and automatic gain control.It consists of 10 sequences of 16 symbols each, and the total duration of the short preamble is 8us. The long training sequence has also a duration of 8us and can be used for channel estimation and fine frequency offset estimation. The long training sequence consists of two long training symbols of 3.2us, along with two guard intervals of 0.8us each. The receiver is assumed to have knowledge of these training sequences. For the purpose of this master thesis, the timing duration of the preamble sequence in an OFDM frame is kept, but only the long training sequence is used for channel estimation. Figure 2.9 shows the OFDM frame format that has been used.

The long training symbol includes 52 BPSK modulated subcarriers. These subcarriers are zero padded until the total number of subcarriers is 64 and then they undergo OFDM modulation. The guard interval (GI) of 0.8us is the cyclic prefix that is inserted to each long training symbol after the OFDM modulation. The 8 short training symbols also contain BPSK modulated data patterns.

2.2.3 Bandpass signal generation

The OFDM frame x_f is composed of complex numbers $x_f = x_i + jx_q$. The spectrum of the OFDM frame is situated at the baseband around DC frequency. In order to be able to perform wireless transmission, the baseband bandwidth of the OFDM frames has to be upconverted from baseband, around a carrier frequency f_c. The process of the upconversion is called IQ modulation. The chosen carrier frequency for the simulations was 212 MHz. Although the actual 802.11g system operates at 2.4 GHz, the simulations are performed at a lower frequency in order to reduce the simulation time. The outcome of the simulations will not be affected since the bandwidth of the simulated 802.11g signal is according to the specifications.

Before the IQ modulation, the OFDM frames will undergo pulse shaping and upsampling. Pulse shaping is a very essential process which takes place at the transmitter as well as the receiver. The benefits of pulse shaping are different at the transmitter and at the receiver side. The purpose of performing pulse shaping at the transmitter side is to reduce the transmitted signal bandwidth.

The other benefit of pulse shaping is to cancel Inter Symbol Interference (ISI) at the receiver in case of limited channel bandwidth or multipath environment.

The pulse shaping at the transmitter was performed via the raised cosine filter block from Simulink. The impulse response of the raised cosine filter is given by the following equation[20].

$$H(\omega) = \begin{cases} T_c & \text{if } |\omega| \leq \omega_1, \\ \dfrac{T_c}{2}(1 + \cos\pi\left(\dfrac{|\omega| - \omega_1}{r\omega_c}\right)) & \text{if } \omega_2 < \omega < \omega_1, \\ 0 & \text{if } |\omega| \geq \omega_2. \end{cases} \tag{2.6}$$

Parameter r is the roll-off factor of the filter which indirectly specifies the bandwidth of the filter. For the simulations the value of the roll-off factor was 0.2. Parameters ω_1, ω_2 are the limits of the passband response of the pulse shaping filter, and ω_c is the input symbol sampling frequency. Another parameter that has to be specified is the upsampling ratio of the pulse shaping filter. The upsampling ratio was set to be equal to 50. The reason for such a high sampling frequency comes from the need for superimposing the 802.11g signal to the Bluetooth signal. The initial goal was to evaluate the PR values in a TGn multipath environment. The tap delay of this environment is 10ns, which requires a high sampling rate. In order to superimpose those to signals, the sampling moments in Simulink should be the same. The difficulty occurs, since the Simulink models use the frame based simulation format. Under this format the signals are created and processed in frames. This process enhances the speed of the simulations, but on the other hand requires that the frame time should be constant. Thus the number of samples per frame can increase if the signal is upsampled but the frame time is the same. By consequence, since the two systems had different frame durations, the only sampling time that would allow superimposing the Bluetooth and the 802.11g signals was 1ns, which corresponds to a sampling frequency of 1GHz. Since the OFDM frame has 20MHz sampling frequency, the upsampling ratio was 50.

The bandpass upconversion around a carrier frequency F_c is achieved through the IQ modulation. IQ modulation requires the multiplication of the complex baseband signal with $e^{j\omega t}$ and the extraction of the real part[21]. The outcome of the i-q modulation is a signal whose spectrum is bandpass.

$$x_{rf}(t) = \Re\left[e^{j\omega t\, x_p(t)}\right] = x_i(t)\cos(\omega t) - x_q(t) * \sin(\omega t) \tag{2.7}$$

The bandpass signal is a real signal and it will have a two-sided symmetrical spectrum. This means that the energy of the complex baseband signal is equally distributed to the positive and negative spectrum of the real bandpass signal. For this purpose,the upconverted signal x_{RF} should be multiplied with $\sqrt{2}$ in order to preserve the symbol energy of the baseband signal.

Figure 2.10 shows the spectrum of the baseband signal and the spectrum of the bandpass signal after the upconversion before the power normalization process.

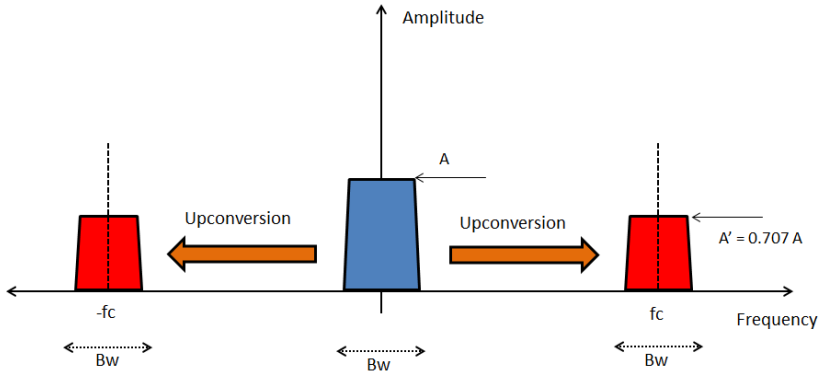


FIGURE 2.10: Baseband and bandpass spectrum. During the upconversion process to bandpass, the energy of the initial baseband signal, which is upconverted to a center frequency fc is half of the initial baseband signal energy. In order to preserve the signal energy, the upconverted signal has to be multiplied with $\sqrt{2}$.

The bandpass signal can now be transmitted after the upconversion process. The spectrum of the transmitted signal is shown in figure 2.11.

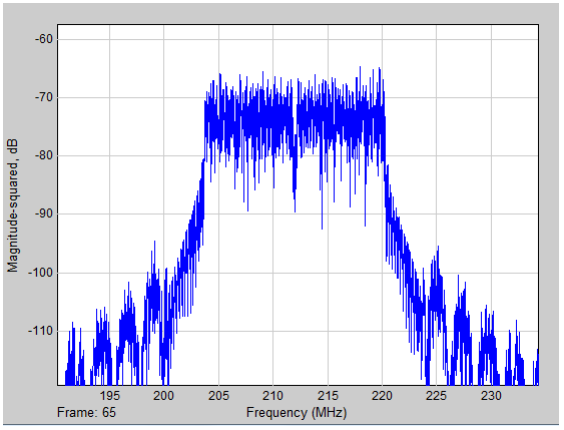


FIGURE 2.11: Spectrum of the transmitted bandpass 802.11g signal. The transmitted signal has a center frequency of 212 MHz and an effective bandwidth of 16.6 MHz.

2.3 802.11g Receiver

This section presents the basic blocks of the 802.11g receiver. The 802.11g receiver similar to the 802.11 transmitter incorporates three basic functional blocks. The first block is responsible for the downconversion of the received bandpass signal to baseband. The second block is responsible for the OFDM demodulation and the extraction of the modulated data. Finally,the third block is responsible for the demapping of the complex symbols to binary format.

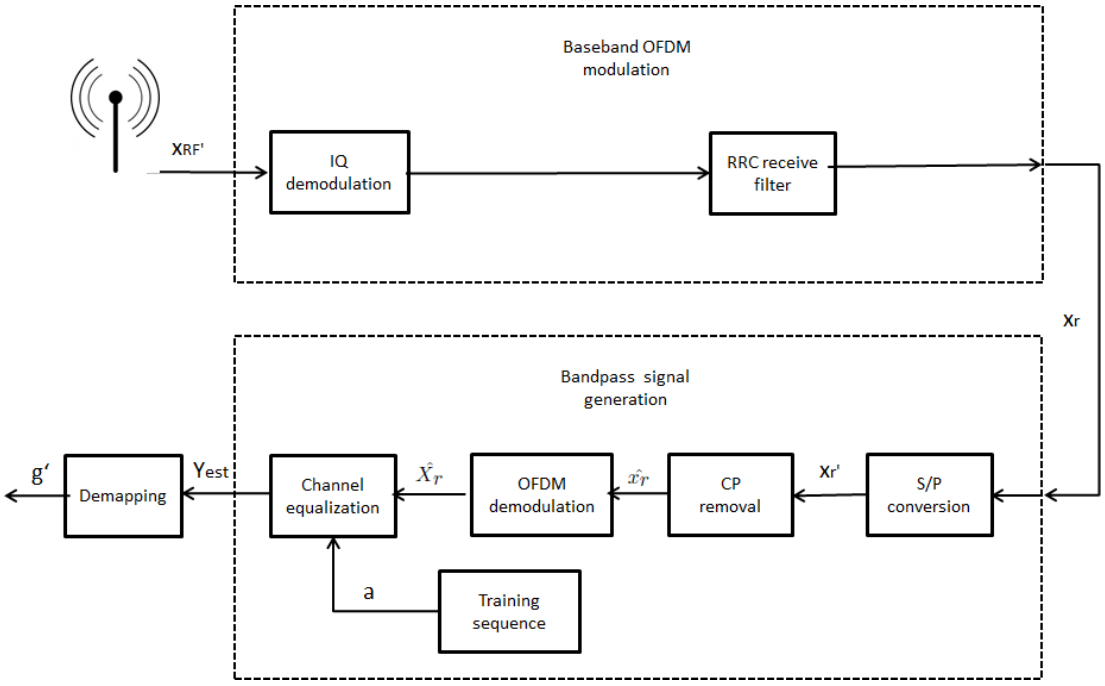


FIGURE 2.12: 802.11g receiver schematic. The first task of the receiver is to perform IQ demodulation, along with downsampling and lowpass filtering of the bandpass signal. As soon as the baseband signal is obtained, the input data are recovered from the OFDM frame, and they are converted from a serial form to a parallel representation. The result of the parallel representation, is a matrix which has at each column an OFDM symbol, with its cyclic prefix at the beginning. The cyclic prefix is removed and the remaining signal is OFDM demodulated through FFT. Moreover, since the training sequence, which is used as the long preamble of each frame is assumed to be known to the receiver, the channel response is estimated, and the OFDM demodulated signal is equalized with the inverse of the estimated channel response. Finally, the equalized symbols are demapped through QPSK and 16 QAM demapping.

2.3.1 IQ Demodulation and Decimation

The first step of the downconversion as it can be seen from figure 2.12 is the IQ demodulation. The IQ demodulation requires the multiplication of the received passband signal $x_{RF'}$ with $e^{-j\omega t}$, where $\omega = 2\pi f_c$, and f_c is the carrier frequency, which is 212 MHz.

$$x_{b'}(t) = \left[e^{-j\omega t}\, x_{RF'}(t)\right] \tag{2.8}$$

The obtained signal $x_{b'}(t)$ has the following low frequency and a high frequency components.

$$x_{hf}(t) = -0.5x_i(t)\sin(2\omega\, t) - 0.5x_q(t)\sin(2\omega\, t) \tag{2.9}$$

$$x_{lpf'}(t) = x_i(t)\cos(\omega\, t)^2 + x_q(t)j\sin(\omega\, t)^2 \;=\; x_i(t) + x_q(t)j \tag{2.10}$$

The downconverted baseband signal, x_r is the low frequency component $x_{lpf'}$. The high frequency component is removed with the use of the raised cosine receive filter. Specifically,Simulink's raised cosine receive filter was used as a reception pulse shaping filter. The parameter values of the reception raised cosine filter were the same as the parameter values of the transmission raised cosine filter. The output of this filter is a complex baseband signal with a sampling rate of 20 MHz.

2.3.2 OFDM Demodulation

As soon as the received signal is converted to baseband the first thing that has to be done is the synchronization of the OFDM frames. The synchronization process is necessary due to the unknown time delay of the reception. As far as the first part of the simulations was concerned, since the transmission was continuous, the time delay was known so there was no need for extra synchronization.

The steps that are necessary for obtaining the mapped symbols are the following:

- Serial to parallel conversion
- Cyclic prefix removal
- OFDM demodulation
- Channel equalization

Serial to parallel conversion

Each received OFDM frame is a vector which consists of 1920 OFDM symbols. These 1920 OFDM symbols have to be converted from serial to parallel representation. If $x_r = [x_r(1), ..., x_r(1920)]^T$ is the vector which represents the received OFDM frame, then the matrix that represents the parallel converted data x'_r which has dimensions $[80, 24]$ is:

TABLE 2.2: The parallel matrix x'_r

Xr(1)	...	Xr(1841)
...	...	...
Xr(80)	...	Xr(1920)

The composition of the x'_r matrix is:

∗ **Columns 1-2** Short Preamble

∗ **Columns 3-4** Long Preamble

∗ **Columns 5-24** OFDM Data

Cyclic prefix removal

Each column of the x'_r matrix is a vector [80, 1] which represents an OFDM symbol along with the corresponding cyclic prefix. The first 16 elements of each column vector which are the cyclic prefix have to be removed. The remaining 64 elements of each OFDM symbol in the time domain have to undergo OFDM demodulation. Figure 2.13 which represents one column of the x'_r matrix, shows an example of the cyclic prefix removal procedure. The extracted OFDM symbol is the $\hat{x_r}$ vector and in the figure it is the part of the column in white.

x'_r(1,1)
...
x'_r(16,1)
x'_r(17,1)
x'_r(18,1)
...
x'_r(79,1)
x'_r(80,1)

FIGURE 2.13: Cyclic prefix removal. The first 16 samples of each column which are depicted in dark color are removed and the remaining 64 samples are the obtained OFDM symbol.

OFDM demodulation

OFDM demodulation is performed via FFT and gives the frequency domain representation of the signal. The FFT algorithm was implemented by the FFT block of Simulink. The output of the FFT, $\hat{X}_r$,is the frequency domain representation of the received signal.

$$\hat{X_r}(l) = \frac{1}{\sqrt{Ns}} \sum_{n=0}^{Ns-1} \hat{x_r}(l) e^{-j\frac{2k\pi n}{Ns}}, \tag{2.11}$$

where $\hat{X}_r(l)$ is the extracted symbol from the l_{th} subcarrier. The extracted frequency domain symbols also have to be multiplied by the factor $\frac{\sqrt{52}}{64}$ because during the IFFT some of their energy was spread to the zero subcarriers.With this multiplication their energy is restored.

Channel equalization

Once the data have been OFDM demodulated, their processing can be done at the frequency domain. The channel estimation and equalization takes place via the long training sequences. The receiver is assumed to have perfect knowledge of the transmitted long training sequence. If a is the BPSK modulated known trainining sequence and $\hat{a}$ is the extracted training sequence from each OFDM frame, then $\hat{b}$ is the ratio of $\hat{a}$ over a, given the fact that the dc component is removed.

$$\hat{b} = \frac{\hat{a}}{a} \tag{2.12}$$

The gain based on which the equalization will take place is given by the following equation:

$$\hat{H}_e = \frac{\overline{\hat{b}^*}_{rms}}{|\hat{b}|^2_{rms}} \tag{2.13}$$

The frequency domain equalization then is:

$$Y_{est} = \hat{X}_r \, \hat{H}_e \tag{2.14}$$

This form of equalization is called zero forcing equalization, and although it is simple it has the drawback of enhancing the noise.

Demapping

The final step in order to obtain the information bits is the demapping of the symbols into bits. The demapping was performed with the use of Simulink's blocks "Rectangular QAM Demodulator Baseband" and "QPSK Demodulator Baseband". The "Rectangular QAM Demodulator Baseband" block was used for the 802.11g system that uses 16 QAM mapping and the "QPSK Demodulator Baseband" block was used for the 802.11g system that uses QPSK mapping.

2.4 Bluetooth Transceiver

The multicarrier 802.11g was examined versus the singlecarrier Bluetooth [3]. Bluetooth is a greatly used singlecarrier system, which operates at the 2.4 GHz ISM band. The range of the ISM frequency band is from 2400 up to 2483.5 MHz for most of the countries with the exception

of France, Spain and Japan. This frequncy band includes two guard bands 2400-2402 MHz and 2480-2483.5 MHz in order to minimize out-of-band interference. Inside the remaining part of the ISM spectrum, 2402-2480 MHz, 79 RF channels are created, with 1 MHz channel spacing. This is the band inside which Bluetooth is operating.

Bluetooth performs frequency hopping among the 79 RF channels of the ISM band in order to minimize interference and fading. The algorithm for the choice of the frequency hopping sequence can be either pseudorandom in its simplest form, or adaptive [22] in order to avoid interference from WLANs. The frequency hops occur every 625us and this time duration is also called time slot. The communication between Bluetooth devices occurs during the time slots. The information bits undergo GFSK modulation and the resulting symbols are grouped into packets with a symbol rate of 1 Msyms/s. The packets are transmitted within these time slots. Normally one packet occupies one time slot but the duration of a packet can be extended up to 5 time slots. Bluetooth communication can be point-to-point or point-to-multipoint. In the point-to-point communication one bluetooth device acts as master and the other as slave.In a similar way,in point-to-multipoint communication one device will be the master device and up to 7 other devices can be the slaves. In order to achieve full-duplex communication,the Time-Division-Duplex (TDD) scheme is applied. With this scheme the master device transmits at the even-numbered time slots and the slave device transmits at the odd-numbered time slots.

2.4.1 Bluetooth transmitter

In this subsection, the basic parts of the used passband Bluetooth transmitter are going to be analyzed. Figure 2.14 depicts the basic blocks of the Bluetooth transmitter and the corresponding signal notation. As it is stated in the introduction chapter, the system architectures that are used for the calculation of the interference matrix do not include coding, thus the used Bluetooth transceiver does not have an error correction mechanism.

GFSK Modulation

Some the key features of the Bluetooth technology are the low cost, the low complexity and the low power consumption of the Bluetooth devices. By consequence, in order to achieve low complexity, the modulation scheme that is used is a shaped binary FM modulation. A common binary FM modulation scheme that is used in the Bluetooth transceivers is the Gaussian Frequency-Shift Keying (GFSK) modulation [23],[24]. The physical representation of the input bit sequence is a sequence of rectangular pulses. The problem of rectangular pulses is the bandwidth that it is required in order to represent the sudden rectangular transitions. Thus, the rectangular pulse sequence is pulse shaped in order to minimize the required bandwidth. As

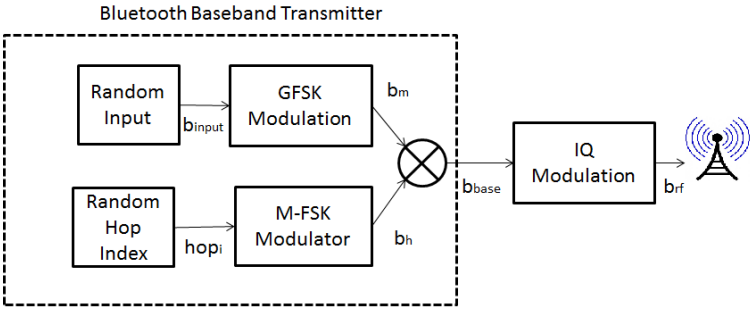


FIGURE 2.14: Bluetooth transmitter schematic. The random input bit sequence is GFSK modulated with BT product 0.5 and modulation index 0.35. The GFSK modulated pulses are then shifted to the appropriate IF channel with the use of a 79-FSK modulator. The input of the 79-FSK modulator block is a number from 0-79 and it is updated every 625us. The last step is the conversion of the Bluetooth signal to bandpass through IQ modulation.

the name of GFSK implies, the pulses undergo Gaussian pulse shaping. The folowing formula provides the mathematical description of GFSK modulation.

$$s(t) = \cos(2\pi\, f_c\, t + 2\pi\, h_m \int_{-\infty}^{t} \sum_{i=1}^{i=n} [I_n * g(t - nT)]\; dt + n_1) \tag{2.15}$$

In the above formula, I_n is the input bit at the moment nT, g(t) is a Gaussian pulse which is convolved with a unit rectangular signal of duration T, h_m is the modulation index, f_c is the carrier frequency and n_1 is a random phase[24]. According to the specifications [23] the modulation index h_m has values between 0.28 and 0.35. The GFSK baseband modulator was implemented in Simulink with the Continuous Phase Modulation (CPM) modulator baseband object. CPM modulated signals do not suffer from phase discontinuities due to the transitions of the input rectangular sequence. This makes them an atractive candidate for Bluetooth systems since the constant phase that they have,leads to high spectral efficiency. Moreover, since the CPM modulation is implemented as constant envelope modulation, the CPM modulated signal is power efficient. The input of the CPM modulator object undergoes pulse shaping. In this case the input rectangular pulses undergo Gaussian pulse shaping with BT product 0.5. Moreover the modulation index was chosen to be 0.35. The modulation index is related to the frequency deviation, f_d, of the GFSK signal according to the following formula.

$$h_m = \frac{2\, f_d}{R} \tag{2.16}$$

In the above equation R is the input bitrate. The spectrum of the GFSK modulated symbols is depicted on figure 2.15.

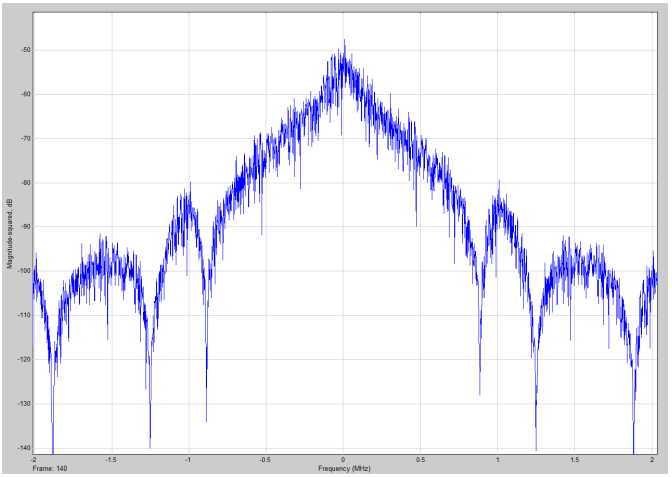


FIGURE 2.15: Bluetooth spectral mask.

Frequency Hopping

As it was stated in the subsection Bluetooth Transceiver, Bluetooth performs frequency hopping through 79 possible frequencies in order to avoid interference and channel fading. The frequency hopping functionality was implemented with a 79-FSK baseband modulator block from Simulink. This block can produce 79 possible waveforms with a different frequency. The possible output frequencies are 1 MHz appart as it is required by the Bluetooth standard[3]. The input of the 79-FSK baseband modulator block is an integer between 0 and 78 which denotes the frequency hop index. The frequency hop indexes are generated by a uniform distribution function every $\frac{1}{1600}$ seconds. The choice of a uniform distribution over an adaptive scheme for frequency hopping can be justified, since an adaptive scheme would give a more optimistic results. Since the interference matrix will be used along with several other parameters in order to define the medium access probability, an average case was chosen.

Bandpass Signal Generation and IQ Modulation

The baseband frequency hopping GFSK modulated signal, which had a sampling rate of 100Msps, had to be upsampled and upconverted in a similar way as the 802.11g system. The process of upsampling and pulse shaping took place again with a raised cosine filter. The used pulse shaping filter was the same as the one used for the 802.11g transceiver, but with different upsampling ratio. Specifically the upsampling ratio is 10. The IQ modulation block that was used for the upconversion of the 802.11g signal, was also used in order to upconvert the baseband Bluetooth signal to passband. The only difference is the oscillation frequency of the local oscIllator. As it was already explained in the previous subsection the frequency hop topology generates 79 possible baseband subcarriers which are distributed around dc. The RF band on which the simulations are going to be performed is the 200 MHz to 300 MHz band. By consequence, the local oscillator frequency is set to 241 MHz. Table depicts the specifications of the used Bluetooth IQ modulator.

TABLE 2.3: IQ Modulator specifications

Oscillation Frequency (MHz)	241
Minimum Output Frequency (MHz)	202
Maximum Output Frequency (MHz)	280

2.4.2 Bluetooth receiver

The used Bluetooth receiver performs the following three tasks in order to extract the transmitted bit sequence from the received bandpass signal. The schematic of the used bluetooth receiver is depicted on figure along with the used signal notation.

- IQ Demodulation and Decimation
- Complex Bandpass Decimation
- FSK Demodulation

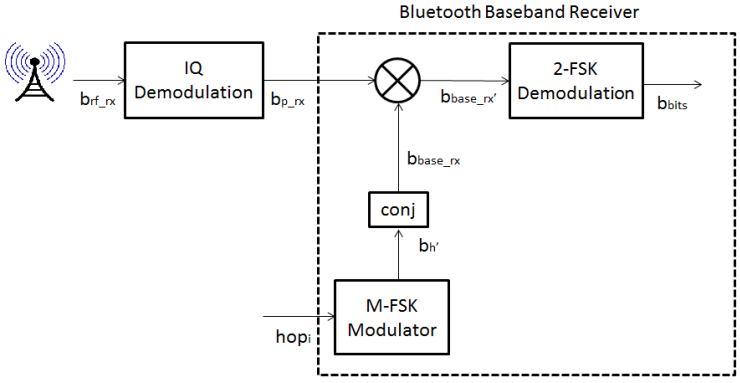


FIGURE 2.16: Bluetooth receiver schematic. The first operation of the Bluetooth receiver is the IQ demodulation of the received signal. Since the receiver has knowledge of the transmitter frequency hops, the equivalent 79-FSK signal is generated at the receiver. The IQ demodulated signal is converted to baseband through multiplication with the complex conjugate of the 79-FSK output. Finally, the baseband signal is demodulated through 2-FSK demodulator, which employs the non-coherent energy detection method.

IQ Demodulation and Decimation

The used IQ demodulation block is similar to the IQ demodulation block that was used at the 802.11g receiver. Again, the only difference is the local oscillation frequency which is set to 241 MHz. The output of the IQ demodulator is the complex baseband signal b_{prx}. The next step is the low pass filtering and the decimation of signal b_{prx}. This process takes place with a raised cosine filter with downsampling ratio equal to 10.

Complex Bandpass Decimation

The complex bandpass signal b_{prx} is a frequency hopping signal and its IF frequency can take values from -39 MHz up to 39MHz. In order to extract the baseband signal, the process of complex bandpass decimation has to be performed according to the following procedure. The receiver is assumed to have perfect knowledge of the hopping sequence. The hopping sequence indexes hop_i which denote the frequency hop_i, are modulating a 79-FSK modulator block. This 79-FSK modulator block is similar to the one that was used for the generation of the IF frequency hopping signal at the transmitter. Since the signal has to be decimated to baseband this time, the complex conjugate of the output of the 79-FSK modulator block has to be taken. The equivalent signal is the b_{baserx} signal, which is multiplied with the IF signal b_{prx}. The result of this process is the extraction of the baseband signal $b_{baserx'}$ which carries the CPM modulated information symbols.

FSK Demodulation

The final task of the Bluetooth receiver is the demodulation of the received baseband signal. The demodulation method that was used was the non-coherent energy detection. This process is implemented with the M-FSK demodulator block of Simulink. Although this is not the most elaborate demodulation method it is a cheap and simple approach. This demodulator achieves a BER of 0.001 for a Signal-to-Noise-Ratio(SNR) of 16.7 dB. Although this behavior is not optimal, it is not much worse than the optimal BER values which are presented at [23]. Further enhancement in the system performance can be achieved with the use of a Maximum a Posteriori (MAP) receiver. This option could further drop the SNR value that is required in order to achieve a BER of 0.001 to 11 dB [25]. Alas, a Viterbi decoder would have been required and at this stage of the simulations, the system should not have any coding mechanisms.

2.5 Raw BER of the 802.11g transceivers in AWGN environment

A first step in order to verify the behavior of the 802.11g transceiver is to calculate the BER in a Additive White Gaussian Noise(AWGN) environment. The term white in the AWGN refers to the constant spectral density N_o (W/Hz) of the AWGN. The amplitude of the AWGN follows a Gaussian distribution, with a zero mean value and standard deviation σ^2. The examination of the system's BER performance under AWGN is of importance. It gives the opportunity to validate the correct behavior of the system. Moreover, it allows to estimate the effect of the used blocks such as the zero-forcing equalizer on the system's BER.

The error probability of the OFDM modulated signals, depends mainly on the error probability of the mapping method. The error probabilities of QPSK and 16 QAM modulations, can be derived from the following equations[26]. In these equations, the term E_b is the energy per

bit. The argument of the Q function is actually the square root of the energy per symbol of the modulation, E_s, over the noise spectral density N_o. Since the created 16 QAM symbols, are already normalized in order to have average power equal to 1, equation 2.18 can be further simplified into equation 2.19.

$$p_{qpsk} = Q(\sqrt{\frac{k\,E_b}{N_o}}) \quad , k = \log_2 4 = 2 \tag{2.17}$$

$$p_{16qam} = \frac{3}{2\,k} Q(\sqrt{\frac{k\,E_b}{10N_o}}) \quad , k = \log_2 16 = 4 \tag{2.18}$$

$$p_{16qam} = \frac{3}{2\,k} Q(\sqrt{\frac{k\,E_b}{N_o}}) \quad , k = \log_2 16 = 4 \tag{2.19}$$

where,

$$Q(z) = \frac{1}{\sqrt{2\,\pi}} \int_z^\infty \exp\,(-\frac{\lambda^2}{2})\,d\lambda \tag{2.20}$$

Since each QPSK modulated symbol corresponds to 2 bits, and each 16 QAM modulated symbol corresponds to 4 bits, the $\frac{E_s}{No}$ ratios are linked to the $\frac{E_b}{No}$ ratios through equations 2.21 and 2.22. In addition to that, $\frac{E_s}{No}$, is related to the SNR of the system according to equation 2.23.

$$\frac{E_s}{N_o} = 2\frac{E_b}{No} \tag{2.21}$$

$$\frac{E_s}{N_o} = 4\frac{E_b}{N_o} \tag{2.22}$$

$$\frac{E_s}{N_o} = \frac{T}{T_s} SNR \tag{2.23}$$

where T is the symbol duration and T_s is the sampling period, in case the modulated symbols are upsampled.

Bit Error Rate of the Baseband 802.11g model

The process of testing the BER values required the addition of noise to the system and the calculation of the BER. As far as the baseband simulations are concerned, a complex vector n_o was created with 1920 elements, as the number of transmitted symbols per OFDM frame. Vector n_o had zero mean value and variance equal to 1. A weighting factor was also applied to

n_o in order to achieve the desired noise level. The value of the weighting factor was given by equation 2.24.

$$w_n = 10^{-\frac{E_{s_{dB}}}{20N_o}} \tag{2.24}$$

where,

$$\frac{E_{s_{dB}}}{N_o} = \frac{E_{b_{dB}}}{N_o} + 10\log(k) \tag{2.25}$$

Since the each modulated symbol was normalized in order to have symbol energy equal to 1, the total noise component, $n_t = w_n n_o$, was scaled to the desired values. Figure, 2.17, shows the BER of the baseband 802.11g system with 16 QAM modulation. Figure 2.18, shows the BER of the baseband 802.11g system with QPSK modulation. In addition to that, figures 2.17 and 2.18, show the BER of the system without equalization in order to show the slight BER degradation due to the zero forcing equalization algorithm.

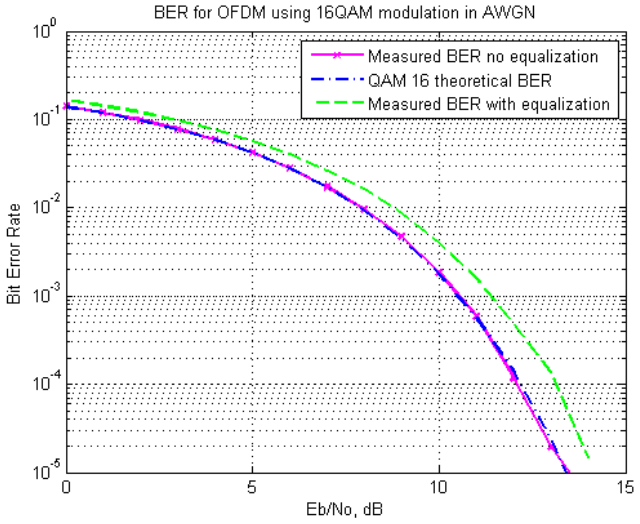


FIGURE 2.17: BER of the 802.11g transceiver with QAM 16 mapping.

2.6 Raw BER of the Bluetooth transceivers in AWGN environment

Since the demodulation of the Bluetooth signals is noncoherent, the error probability is given by formula 2.26 according to [27].

$$P_s = P_b = Q(a,b) - \frac{1}{2}e^{\frac{-(a+b)}{2}} I_0\sqrt{ab} \tag{2.26}$$

$$a = \frac{E_b}{2N_o}(1 - \sqrt{1 - |\rho|2}) \tag{2.27}$$

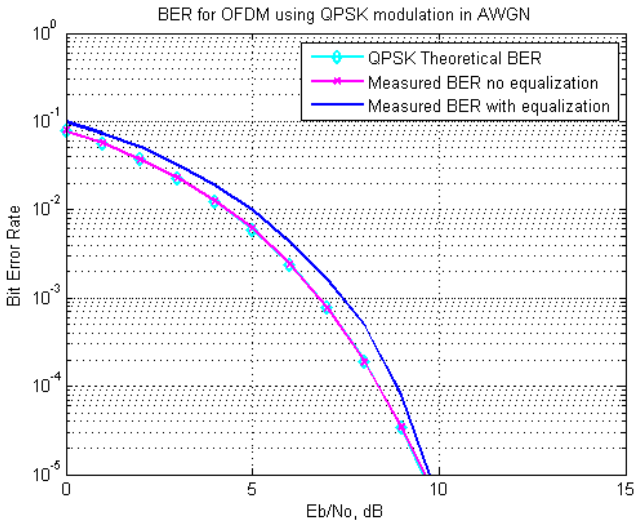


FIGURE 2.18: BER of the 802.11g transceiver with QPSK mapping.

$$a = \frac{E_b}{2N_o}(1 + \sqrt{1 - |\rho|2}) \tag{2.28}$$

$$\rho = sinc(2f_dTs) \tag{2.29}$$

The coefficient ρ, is the correlation coefficient of the modulated signals and it depends on the symbol period, T, and on the frequency deviation f_d. For $d_f = 0.175$, the correlation coefficient is equal to 0.8103. Figure 2.19, shows the bit error rate of the bluetooth system versus the EbNo values.

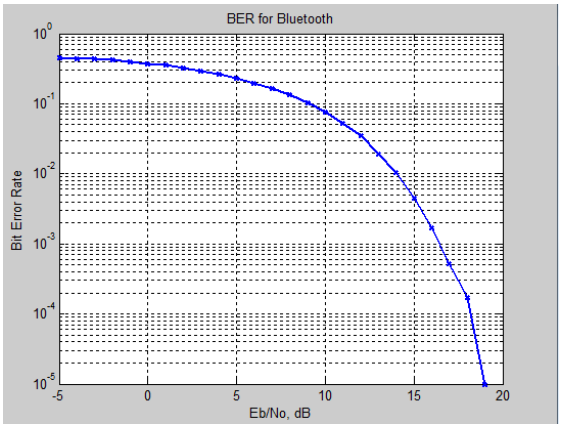


FIGURE 2.19: BER of the Bluetooth system.

Chapter 3

Simulations for the Calculation of the Protection Ratio

3.1 Simulation Procedure

This chapter presents the simulation procedure as well as the simulation measurements for the calculation of the interference matrix. The elements of the interference matrix are the Protection Ratio values. The RF Protection Ratio is the minimum acceptable ratio between the power per MHz, of a wanted wireless signal and any interfering signal at the receiver input, expressed in decibels. Specifically, it is the minimum ratio that leads to a predefined BER value at the receiver. This predefined BER value guarantees that a specific reception quality is achieved at the receiver. The investigated systems have different different bandwidth requirements. By consequence, the average transmitted power by each system corresponds to different bandwidth. The average power of a 802.11g system corresponds to an effective bandwidth of 16.6MHz [28], while the average power of a Bluetooth system corresponds to 1MHz bandwidth. In order to be able to find the relationship between the transmitted powers that provide a predefined BER of 0.001, a common bandwidth has to be defined. In this case, the nominal frequency bandwidth is set to be the 1MHz bandwidth. Thus, the received power from a 802.11g system has to be normalized to represent the power that corresponds to 1MHz.

3.1.1 Simulation Topology

Figure 3.1 depicts the topology that was used in order to obtain the Protection Ratio values. In figure 3.1, system A is the desired transceiver system and system B is the interfering system. The type A transmitter starts transmitting at a constant data rate. At one moment, type B transmitter starts transmitting at a constant data rate too. For the sake of simplicity, the notation system A and system B will be preserved for this chapter. The superposition of the signals A and B passes through the channel and is received by the receiver A. The transmitted power of each system changes with the use of the variable gain factors that are inserted after each system. For each variable gain pair, a number of simulations is required in order to estimate the average BER value. The purpose is to estimate the transmitted power levels of systems A and B in order to obtain an average BER = 0.001. The average power of the signals, $P = \frac{1}{M}\sum_{i=1}^{i=M}|x_i|^2$, is used as a power level measure. The time of the interferer transmission, is not constant but it is assumed to follow auniform distribution. Details about the transmission times are given in the following subsection.

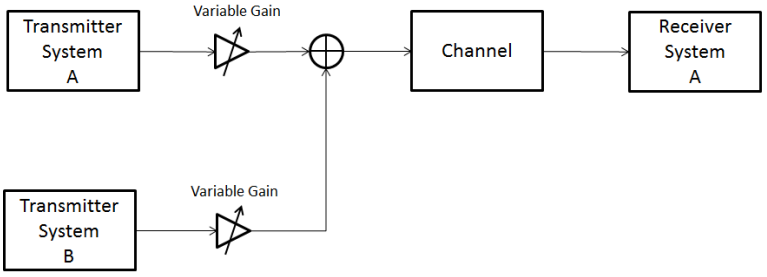


FIGURE 3.1: Schematic of the simulated topology.

3.1.2 Transmission Timing

Figure 3.2, shows the timing of the transmissions. The arrows which are directed upwards denote the beginning of the transmission, while the arrows that are directed downwards denote the end of a transmission. System A, begins transmission at time t_0 and keeps transmitting at a constant data rate until time t_2. The choice of time t_2 corresponds to the time that is required in order to obtain 200000 received bits. Depending on the examined system A, time t_2, is different due to the different throughput of each system. The interferer system, system B, initiates transmission at time $t_1 = t_0 + d_t$. The delay of the interferer transmission, d_t, follows a uniform distribution. The obtained value of d_t is confined between t_0 and t_2.

For each value of the variable gain factors, a number of simulations is performed. In each simulation d_t has a different value which follows a uniform distribution as it was stated above. This happens in order to minimize any correlation that might exist between fixed transmission

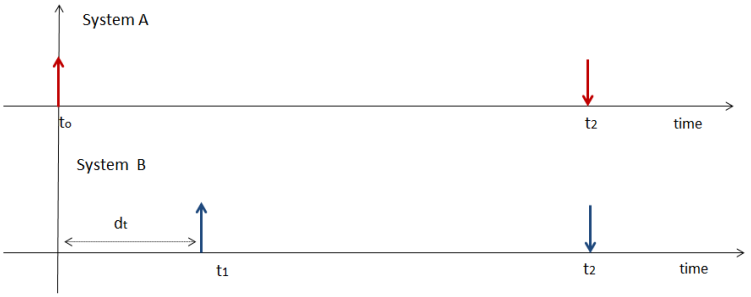


FIGURE 3.2: Transmissions duration.

times for the interferer and the system under evaluation. The obtain BER value corresponds to the average BER of n_i iterations. The number of iterations that was used is 200. Once the average BER 10% difference from the desired value 0.001, the simulations stop.

3.1.3 Simulated Data Rates

The data rates of the simulated systems are presented in this subsection. The effective data rates depend on multiple parameters. Table 3.1 summerizes these parameters.

TABLE 3.1: Simulated Data Rates

Modulation	Baud Rate (Bd) (MBaud/s)	Bits / Baud (nb)	Throughput (Th) (Mbps)
802.11g-QPSK	12	2	24
802.11g-16QAM	12	4	48
Bluetooth GFSK	1	1	1

The baud rate is the modulated symbol rate of each system. In the case of the 802.11g system, the calculated baud rate corresponds to the total symbol rate of 48 data subcarriers. Moreover, since we assume that the systems in these simulations do not include any coding, the bits per baud rate, corresponds to the number of the bits that can be represented by each modulated symbol. In case of QPSK modulation, each QPSK modulated symbol corresponds to two bits and for this reason the bits per baud rate is 2. As far as 16 QAM modulation is concerned, the bits per baud rate is 4 corresponds to 4 bits. Parameter MAC throughput which is also calculated depends on the number of OFDM data subcarriers inside a MAC frame. For the simulations, the OFDM subcarriers that have data inside a MAC frame occupy 50% of the frame due to the cyclic prefix as well as the preamble. Table 3.2 shows the effective MAC data rate, Th, which has been used for the simulations. The effective MAC data rate is analogous to the baud rate (Bd) and to the number of bits per baud (nb).

TABLE 3.2: MAC Throughout

Modulation	MAC Throughput (Mbps)
802.11g-QPSK	12
802.11g-16QAM	24
Bluetooth GFSK	1

3.1.4 RF Frequencies

The simulations that have been performed in order to determine the maximum interference power levels that can be tolerated can be distinguished into two categories. The first category takes into account adjacent channel interference and the second co-channel interference. The corresponding carrier frequencies of the evaluated systems are presented in this section.

Co-Channel Interference

The co-channel interference simulations examine the behavior of the 802.11g and Bluetooth systems versus interference from a 802.11g or a Bluetooth system. The 802.11g system is up-converted to 212 MHz and the Bluetooth system can perform hops between 202 MHz and 279 MHz.

When the Bluetooth system is evaluated versus a 802.11g system, the Bluetooth hops are confined inside the 20 MHz spectrum of the 802.11g system. If the Bluetooth was allowed to perform hops in the full hopping range, since the interferer transmission time is random, the simulation time would have to increase significantly in order to have an accurate estimation of the total number of hops that fall inside the 802.11g band spectrum. The accurate estimation of hops, allows us to have a more accurate estimation of the interference time versus the total transmission time. Unfortunately, due to the implementation of the blocks in passband, increasing the simulation time significantly causes the computer memory to overload. When systems A and B where both 802.11g systems, the center frequency of the system B was chosen to be 212.05 MHz instead of 212 MHz in order to avoid coherent interference.

Adjacent Channel Interference

For the adjacent channel interference simulations, an 802.11g system which is upconverted to 212 MHz is system A of figure 3.1. System B of figure 3.1 is another 802.11g system, whose spectrum partially overlaps the spectrum of system A. Table 3.3 shows the center frequencies of the adjacent channels.

TABLE 3.3: Center Frequencies for Adjacent Interference Scenario

System	Case1	Case2	Case3	Case4
A	212 MHz	212 MHz	212 MHz	212 MHz
B	212.05 MHz	217.05 MHz	222.05 MHz	227.05 MHz

3.2 Simulation Results

Flat Fading - Adjacent Channel Interference

The simulated environment was an ideal environment and the first results correspond to adjacent channel interference of the 802.11g systems. The results, for the 16 QAM and the QPSK implementation are shown in figure 3.3. The central frequency of the interferer signals is depicted on axis-x of figure 3.3 while the Protection Ratio values are depicted on axis-y. The upper line corresponds to the PR values of the 802.11g - 16 QAM system, while the lower to the 802.11g - QPSK system. The additional measurements with AWGN as interferer, aim to verify the convergence of the measured PR values with the theoretical ones. As far as the noise power is concerned, the average noise power which corresponds to the effective bandwidth of 16.6 MHz is taken into account.

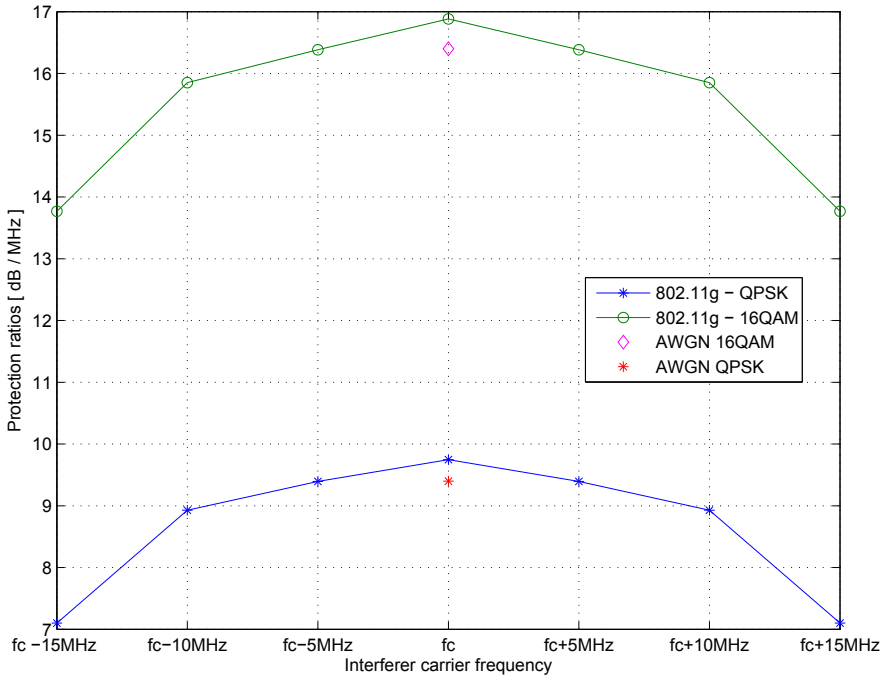


FIGURE 3.3: PR [dB/MHz] in the case of 802.11g co-channel and adjacent channel interference by a 802.11g system and by AWGN.

A first examination of figure 3.3 shows that the measured PR values for 802.11g co-channel interferers is really close to the PR value in case of AWGN. This remark denotes that the methodology for the computation of the PR values is correct. As it was expected, the overall protection

ratios for the 16 QAM system are higher than the ones of the QPSK system. This is logical, since the error probability of 16 QAM modulation is higher than the error probability of QPSK modulation.

In detail, when the interferer system occupies the same channel as the investigated system, the required PR for the 16 QAM system is approximately 6.67 dB larger than the PR of the QPSK system. According to theory, the required $\frac{E_b}{N_o}$ values in order to achieve BER = 0.001 for 16 QAM and QPSK modulations are 10.5 dB and 6.75 dB respectively. In case of systems that occupy the same bandwidth, the PR is equal to the $\frac{E_s}{N_o}$ ratio. The relation between $\frac{E_b}{N_o}$ and $\frac{E_s}{N_o}$, has been presented in equation 2.25.

Since the modulation index of 16 QAM is twice the modulation index of QPSK, according to equation 2.25:

$$\frac{E_{s_1}}{N_o} = \frac{E_{b_1}}{N_o} + 10\log(k_1) \quad \text{for } QPSK. \tag{3.1}$$

$$\begin{aligned} \frac{E_{s_2}}{N_o} &= \frac{E_{b_2}}{N_o} + 10\log(k_2) \quad \text{for } 16\ QAM. \\ &= \frac{E_{b_1}}{N_o} + 3.75 + 10\log(k_2) \\ &= \frac{E_{s_1}}{N_o} + 6.75 \end{aligned} \tag{3.2}$$

Equation 3.2 shows that the required theoretical $\frac{E_s}{N_o}$ ratio for 16 QAM is 6.75 dB higher than the equivalent ratio for QPSK. Since both systems occupy the same effective bandwidth and have the same sampling rate, the PR is equal to the $\frac{E_s}{N_o}$ ratio which is equal to 6.75 dB. The theoretical value is close to the measured value which is 6.67[dB/MHz]. According to figure 3.3 and table 3.4, which shows the exact PR values, as the interferer bandwidth decreases, the protection ratio value also decreases. Since the spectral overlap of the systems is smaller, the PSD of the interfering system should be bigger in order to obtain the same BER.

Table 3.4: PR in the case of adjacent channel 201.11g interference.

System	fc	fc+5	fc+10	fc+15
802.11g - 16QAM	16.88	16.38	15.85	13.77
802.11g - QPSK	9.75	9.4	8.93	7.1

Flat Fading - Co-Channel Interference

The second set of measurements on the flat fading environment involved the co-channel interference of the systems. The result of this set of measurements will provide the PR values which are going to be used for the creation of the Interference Matrix. Each modulation system is interfered by other modulation systems, and by AWGN. The PR of each investigated system are first depicted versus the theoretical BER - $\frac{E_s}{N_o}$ curve . As it was already mentioned the $\frac{E_s}{N_o}$ ratio of each modulation scheme is equal to the PR value. For this reason the theoretical BER curve can be also plotted versus the PR values. Figure 3.4 shows the PR value of a 802.11g 16 QAM system, versus different types of interferers. Table 3.5 shows the exact PR values. Before explaining the results, it is necessary to specify to which values the parameters of the table do correspond.

Description of the used parameters in the tables:

- System Power: Average power of the interfered system in the whole system bandwidth
- Interferer Power:Average interferer power in the whole interferer bandwidth
- Power Ratio: Ratio of average power levels without bandwidth normalization
- PR: Ratio of average power levels per MHz

TABLE 3.5: 802.11g 16 QAM system versus co-channel interference measurements.

Interferer	System Power [W]	Interferer Power [W]	Power Ratio [dB]	PR[dB/MHz]
802.11g	1	0.0205	16.88	16.88
Bluetooth	1	0.0033	24.75	12.53
AWGN	1	0.019	17.2	16.4

According to the figure and table 3.5, the PR of 802.11g 16 QAM versus AWGN as an interferer is close to the theoretical ratio $\frac{E_s}{N_o}$ = 16.75 dB. The PR of the 802.11g 16 QAM system against another 802.11g system that uses OFDM modulation is 16.88 dB/MHz, which is also close to the theoretical value. This was expected since the 802.11g with OFDM modulation system has a behavior which is similar to AWGN. For this reason, in future simulations instead of the complete system, a noise source with similar spectral mask can be used instead. Moreover, there is no difference in the PR value against 802.11g systems with different types of symbol mapping. The reason for that is that the output of the mapping constellation is normalized in order to have energy equal to one, and by consequence the effect is exactly the same. The last PR measurement involves interference from a Bluetooth system that performs frequency hopping inside the 802.11g channel. The equivalent PR value corresponds to Bluetooth interference at the data subcarriers of the OFDM symbol. By consequence, a Bluetooth system that would

like to perform frequency hopping in the equivalent frequencies has to limit its output power to -12.53 dB below the 802.11g -16 QAM power spectral density.

In case the frequency hops are performed in the spectrum part which is occupied by the pilot subcarriers, the PR should be larger. The pilot subcarriers are used for phase offset compensation and in some cases also for channel estimation and equalization. In that case, the impact of interference is more significant on these subcarriers and the PR should increase significantly. An example of the effect of Bluetooth interference on specific OFDM subcarriers can be found in [29]. The tested system of that case is a 802.11g 64 QAM system and according to the results, the PR increases more than 10dB when the Bluetooth system affects the pilot subcarriers. The difference between the PR of [29] in case of interference on the data subcarriers, arises from the fact that the hopping rate inside the 802.11g bandwidth is higher in our simulations.

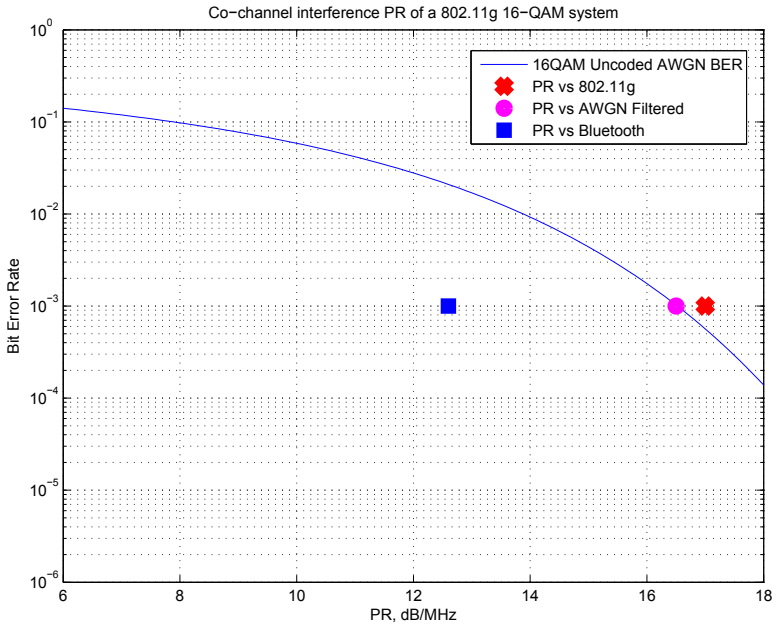


FIGURE 3.4: PR of 802.11g 16 QAM system against co-channel interference.

Figure 3.5 shows the PR value of a 802.11g QPSK system, versus different types of interferers. Table 3.6 shows the obtained PR values. The PR against filtered AWGN and 802.11g interferer systems is also really close to the theoretical $\frac{E_s}{N_o}$ = 9.75 dB/MHz. The PR against a Bluetooth system, that has the same hopping pattern as the one used against the 16 QAM system, is 7.4 dB and it corresponds to the required PR against interference in the data subcarriers of the OFDM symbol.

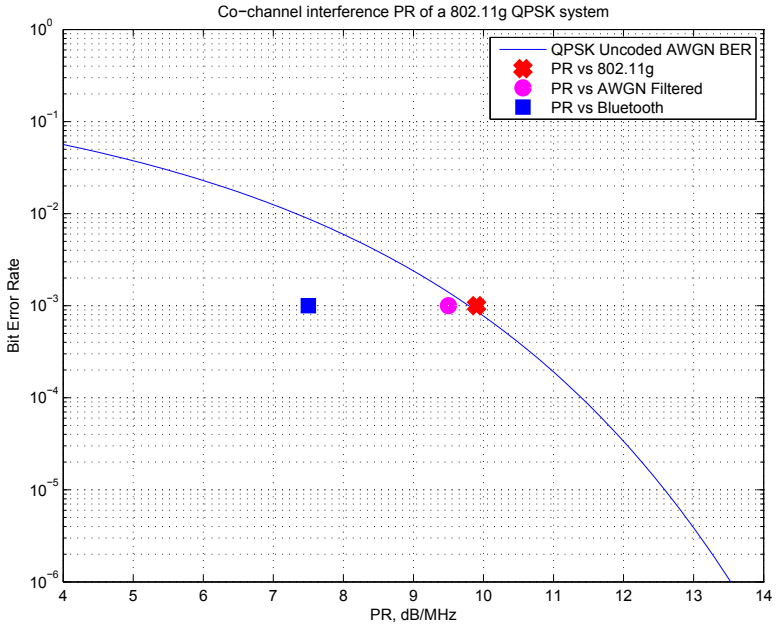


FIGURE 3.5: PR of 802.11g QPSK system against co-channel interference.

TABLE 3.6: 802.11g QPSK system versus co-channel interference measurements.

Interferer	System Power [W]	Interferer Power [W]	Power Ratio[dB]	PR[dB/MHz]
802.11g	1	0.106	9.75	9.75
Bluetooth	1	0.011	19.6	7.38
AWGN	1	0.095	10.2	9.4

Finally, figure 3.6 shows the PR value of a Bluetooth system, versus different types of interferers. Table 3.7 shows the obtained PR values. From the equivalent figure we can see that the PR of a Bluetooth system against a 802.11g interferer is close to the PR against a AWGN interferer. Normally, due to the frequency hopping ability of the Bluetooth system, the PR value should have been smaller. Bluetooth can avoid constant 802.11g interference. This PR value can be explained since the scenario which was investigated, involved full occupation of the spectrum by 802.11g systems which should coexist with a Bluetooth system. In that case, a large percentage of the Bluetooth hops will be interfered by a 802.11g system. For this reason, the required Bluetooth PR against 802.11g interference is significantly larger than the one which is computed in [29]. Specifically, the computed PR is 14.94 dB/MHz while the simulation results of [29] estimate the required PR to approximately 7 dB.

The PR of a Bluetooth system against co-channel interference from another Bluetooth system was evaluated with the following procedure. The power spectral density was tuned to be 10 dBm higher than the Bluetooth receiver sensitivity [3] which is -70 dBm and the required PR was computed. In order to avoid possible constructive interference, the interferer had a frequency

shift of 100 kHz at the carrier frequency. In addition to that, extra noise that corresponded to a BER of 10^{-5} was added to the system, in order to minimize the correlation between the two Bluetooth systems. According to the Bluetooth standard [3], the desired C/I ratio that provides a raw BER of 0.001 is 11 dB. This is the desired value for the optimal Bluetooth receiver. Since our receiver is not optimal from a BER point of view, the desired PR should have been higher. Specifically, it should be close to the $\frac{E_s}{N_o}$ value that leads to 0.001 BER. The obtained PR is significantly smaller than this value, most probably due to constructive interference. Although many measures have been taken in order to minimize the correlation of the Bluetooth systems, a complete isolation could not be achieved. Nevertheless, the Bluetooth transceiver achieves the theoretical BER value and has the appropriate spectral mask.

TABLE 3.7: Bluetooth system versus co-channel interference measurements.

Interferer	System Power	Interferer Power	Power Ratio [dB]	PR[dB/MHz]
802.11g	1	0.53	2.75	14.94
Bluetooth	1	0.18	7.5	7.5
AWGN	1	0.021	16.7	16.7

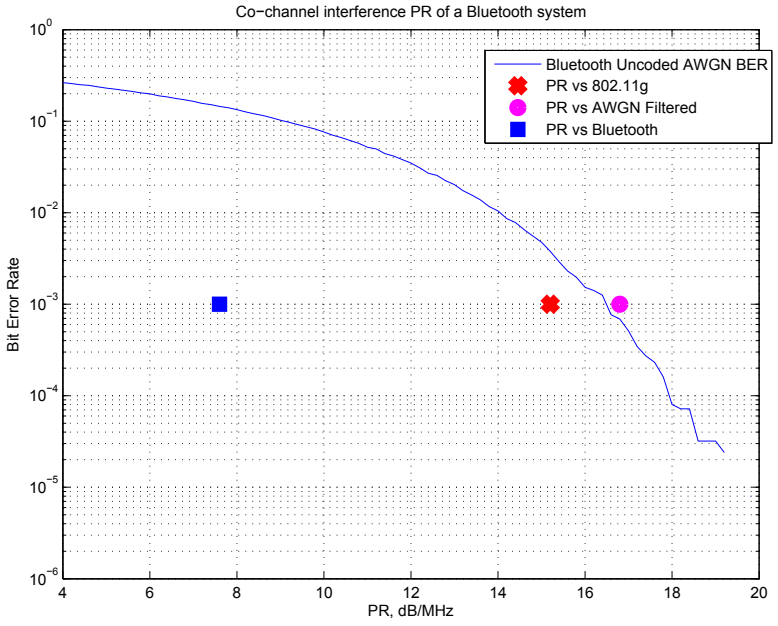


FIGURE 3.6: PR of Bluetooth system against co-channel interference.

3.3 Interference Matrix and Possible Applications

The obtained PR values against co-channel interference are depicted on the interference matrix of table 3.8. The interference matrix is a method of depicting the sensitivity of different modulation schemes, versus interference. It can provide significant information on the interference power levels that each system can tolerate, and it can show whether interference between two systems is symmetrical or not.

TABLE 3.8: Co-channel Interference Matrix

Interf. System / Victim System	802.11g 16 QAM	802.11g QPSK	Bluetooth
802.11g 16 QAM	16.88	16.88	12.53
802.11g QPSK	9.75	9.75	7.38
Bluetooth	14.94	14.94	7.5

Since the investigated modulation schemes are 3, the specific interference matrix has 4 columns and 4 rows. The first column of the matrix specifies the type of modulation scheme that is going to be interfered. The first row of the matrix, specifies the type of the interfering system. The elements of rows 2-4 and columns 2-4 are the corresponding PR values for each case. The interferer matrix is read in the following way. The diagonal elements correspond to the sensitivity of each modulation scheme when it is interfered by the same modulation scheme. The elements of row i and column j denote the sensitivity of modulation scheme i, when interfered by system j.

An argument against the use of such a matrix with the equivalent PR values, is that most systems have predefined maximum output power levels that cannot be adjusted. For example, the Bluetooth standard, [3] defines the following three power classes, of which only one supports power control:

Class	Max Power(dBm)	Power Control
1	20	Mandatory
2	4	Optional
3	0	Optional

In a scenario where system A is interfered by a Bluetooth class 2 system, the knowledge of the PR of system A versus Bluetooth cannot affect the Bluetooth transmitted power. Nevertheless, these measurements do not aim to be used standalone. The purpose is to use these measurements

in spectrum access mechanism as the one proposed in [11]. In such a mechanism, the modulation sensitivity of each system is only one parameter that affects the sharing of the spectrum. The fair sharing of the spectrum can be performed with the use of other techniques. Such techniques employ the reduction of the spectral overlap, reduction of the spatial overlap and reduction of the time overlap.

Chapter 4

Physical layer of the coded 802.11g transceivers.

The second part of this thesis, focuses on comparing the Opportunistic Error Correction scheme and the Forward Error Correction with interleaving scheme under narrowband interference and under the same the effective throughput criterion. For the remaining part of this thesis the FEC scheme with Hard Decoding(HD) and interleaving will be referred to as scheme *A*. The OEC scheme will be referred to as scheme *B*. The transceiver which has been used for the simulations is the baseband 802.11g transceiver of chapter 2.

The input bit streams are encoded according to schemes *A* and *B*, before being mapped to QPSK symbols. For this part of this master thesis, only one type of modulation has been used, the QPSK modulation. The purpose for doing that was the need to focus mainly on the encoding and decoding schemes *A* and *B* under narrowband interference. Other types of modulations such as 16 QAM have larger error probability than QPSK. An adaptive equalization scheme with better performance than the ZF equalization algorithm, could have enabled the investigation of other modulation types which are more error prone.

4.1 Implementation of the encoding Scheme *A*

Figure 4.1 depicts a schematic of scheme A encoding. The randomly generated bit stream is first encoded with $\frac{1}{2}$ code rate, convolutional codes. The convolutionally coded data, are then rearranged through interleaving, with the purpose of avoiding bursts of errors due to the narrowband interferer. This section includes a short theoretical description of each technique, along with details on its implementation.

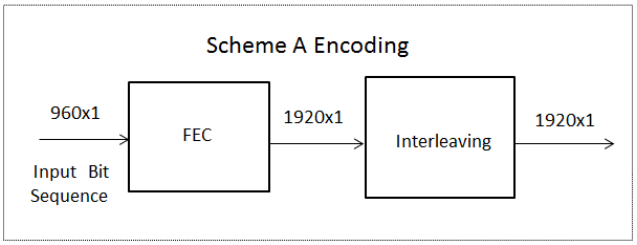


FIGURE 4.1: Block diagram of the scheme *A* encoding.

4.1.1 FEC

Forward Error Correction is a technique that is widely used in communications systems. It involves inserting redundancy to the input bitstream by an Error-Correcting-Code (ECC). The purpose of FEC is to reduce the transmission errors due to noise or interference. FEC encoding is divided into two categories, the block codes and the convolutional codes. The output of block encoders is the input message along with the parity bits. On the other hand, the outputs of convolutional encoders are the parity bits of the encoding procedure [30]. Convolutional coding is the FEC method which is used by the 802.11 systems and specifically by the investigated 802.11g system[31]. Convolutional codes are characterized by the following set of parameters, (k, n, L).

- Parameter k denotes the number of the input bits.
- Parameter n denotes the number of the output bits.
- Parameter L is the constraint length of the code.

The investigated convolutional code had a code rate $Rc = \frac{1}{2}$, and it was implemented by a trellis encoder. The specific trellis encoder was a built-in function by Matlab and accepted as inputs, the generator polynomials [133, 171] in octal form, and the value of the constraint length which was equal to 7. Figure 4.2 shows the topology of the used feedforward trellis convolutional encoder. Since the constraint length is 7, the encoder has 6 memory registers, each one of which introduces delay equal to one sample time. In order to achieve the $\frac{1}{2}$ code rate, for every input bit the encoder should produce two output bits. The output bits are created by two summation nodes whose content is defined by the generator polynomials. Specifically, the binary representation of 133 and 171 is ”001011011” and ”001111001”. Table 4.1, shows the correspondance of the generator polynomial values with the shift register content.

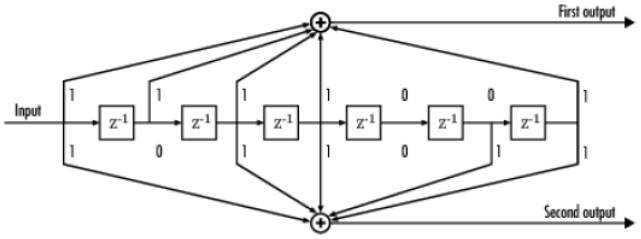


FIGURE 4.2: Trellis convolutional encoder topology.

TABLE 4.1: Correspondance of the summed shift register content to the generator polynomial.

Input sequence a[n]	Generator Polynomial Bit
a[n]	7
a[n-1]	6
a[n-2]	5
a[n-3]	4
a[n-4]	3
a[n-5]	2
a[n-6]	1

4.1.2 Data Interleaving

Data interleaving is another technique in order to combat the effect of burst errors and it involves rearrangement of the input bit stream before transmission. The source of burst errors can be noise in the channel, transmission through a channel with deep fades or interference. As far as this thesis is concerned, interleaving was mainly used in order to minimize the effects of narrowband interference.

Data interleaving for the 802.11g system is performed in two permutations. The first permutation ensures that adjacent coded bits are mapped onto nonadjacent subcarriers. The index of the coded bits after the first permutation is given by equation [33]. Parameter N_{CBPS} is the number of coded bits per OFDM symbol. Since QPSK modulation is used, $N_{CBPS} = 96$.

$$i = \frac{N_{CBPS}}{16}(d \, mod_{16}) + floor(\frac{d}{16}) \tag{4.1}$$

with $d = 0,1,2,...,N_{CBPS} - 1$.

The second permutation ensures that adjacent coded bits are mapped alternately onto less and more significant bits of the constellation. The goal of this permutation is to avoid long runs of low reliability bits. The index after the second permutation is given by equation 4.2. Parameter N_{BPSC} corresponds to the number of coded bits per subcarrier and it is equal to 2.

$$j = s\,floor(\frac{i}{s}) + \left[i + N_{CBPS} - floor(\frac{16i}{N_{CBPS}})\right] mod(s) \tag{4.2}$$

with i = 0,1,2,...,$N_{CBPS} - 1$ and $s = max\left(\frac{N_{BPSC}}{2}, 1\right)$.

Table 4.2, shows an example of the rearranged bit index after interleaving. The presented matrix is read row-by-row. The input of this interleaver example, is a sequence of numbers from 1 up to 96.

Table 4.2: Example of the used interleaver operation for an input vector numbers from 1 up to 96.

1	7	13	19	25	31	37	43	49	55	61	67	73	79	85	91
2	8	14	20	26	32	38	44	50	56	62	68	74	80	86	92
3	9	15	21	27	33	39	45	51	57	63	69	75	81	87	93
4	10	16	22	28	34	40	46	52	58	64	70	76	82	88	94
5	11	17	23	29	35	41	47	53	59	65	71	77	83	89	95
6	12	18	24	30	36	42	48	54	60	66	72	78	84	90	96

4.1.3 Data Deinterleaving

The deinterleaver, which performs the inverse rotation, is also peformed via two permutations. The first permutation reverses the second permutation of the interleaver. The index after the first deinterleaver permutation is given by equation 4.3. The second permutation of the deinterleaver reverses the first permutation of the interleaver. The index after the second permutation of the deinterleaver is given by equation 4.4.

$$i = s\,floor(\frac{j}{s}) + \left[j + floor(\frac{16j}{N_{CBPS}})\right] mod(s) \tag{4.3}$$

$$k = 16i - (N_{CBPS} - 1) floor(\frac{i}{16}) \tag{4.4}$$

4.1.4 Viterbi Decoding

Viterbi decoding of convolutional codes, can be implemented with two methods[30]. The first method is hard decision decoding and the second method is soft decision decoding. In hard decision decoding which is the chosen decoding method, the received bit stream from the interleaver is considered to be the input of the Viterbi decoder.

The Viterbi decoding algorithm can be implemented in the following steps.

1. Branch Metrix (BM) calculation.

2. Path Metric (PM) accumulation through add-compare-select recursion.

The branch metric is a measure of the "distance" between what was transmitted and what was received, and is defined for each state in the trellis diagram. A HDD sets as branch metric the Hamming distance between the expected bits and the received bits. The Hamming distance of two codewords is the number of positions at which the corresponding codewords are different. If r_K is the received binary codeword and s_K is a state of the trellis diagram, the Viterbi decoder calculates at each step the Hamming distance $d_H = |r_K - s_K|$. The path metric is a value associated with a state in the trellis diagram. In the case of HDD, the PM of each state is actually the minimum Hamming distance from the initial state to the current state. Since the code rate is $\frac{1}{2}$, state s at step i+1 can be reached by two predecessor states. If a and b are the predecessor states of s, and the PM of a and b at the time step i are known, then the calculation of the PM of s at time step i+1, where $i \geq 2$, is made accoding to equation 4.5.

$$PM\left[s, i+1\right] = min\left(PM\left[a, i\right] + BM\left[a \mapsto s\right], PM\left[b, i\right] + BM\left[b \mapsto s\right]\right) \tag{4.5}$$

4.2 Implementation of the encoding Scheme *B*

Narrowband interference, affects the subcarriers of the OFDM symbol. Scheme A tries to reproduce the original data, using all of the received data and makes no distinction between subcarriers that undergo interference and the others. Scheme B, which is also known as OEC, employs LDPC and CRC encoding to protect each subcarrier, and discards the subcarriers that undergo strong interference. Then Fountain encoding, which is an erasure channel coding scheme, is used to reconstruct the original data from the remaining packets. Figure 4.3, presents the implementation of scheme B. The randomly generated bits, are first encoded by a Fountain encoder. The Fountain accepts 500 source bits and produces N output bits. This process is repeated 168 times. A matrix is then created with dimensions $N \times 168$. Each row of this matrix will be first CRC encoded and then LDPC encoded, row-by-row. Once the encoding procedure has finished, the following step is the mapping of the symbols and the OFDM modulation. The main characteristic of the OEC scheme is, that each row of the $N \times 168$ matrix will be assigned to a specific OFDM subcarrier.

The used receiver topology differs on one point from the original OEC topology. In the original topology, channel estimation is implemented before the LDPC-CRC decoding scheme. The energy of each subcarrier is compared to a threshold value and the equivalent packet is either rejected or is allowed to continue to the LDPC-CRC decoder. This functionality was not implemented in this Master Thesis since the main purpose is to evaluate the performance of the algorithms under interference. Nevertheless, the absence of such a scheme could allow the existence of undetected error bits at the LDPC-CRC decoder output. In order to solve the problem

of undetected error bits passing to the Fountain decoder, the transmitted LDPC-CRC encoded sequence is known to the receiver and the BER is computed for each 168 bit CRC output. Once there are errors that could not be detected by the CRC decoder, the whole CRC output which corresponds to a specific Fountain packet is discarded. Since the CRC output is now error free, it is further on processed by the Fountain decoder which tries to reconstruct the original data from the N' received packets. The basic parts of the OEC scheme are presented in the following subsections.

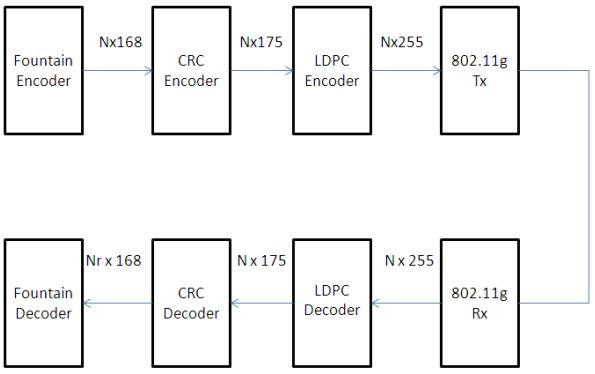


FIGURE 4.3: Block diagram of scheme *B* transmitter. The Fountain encoder produces an output matrix with $N = 732$ lines and 168 columns. Each line of the matrix corresponds to a specific Fountain packet and it is CRC encoded. After the CRC encoding, the matrix has dimensions $N \times 175$. Further on, each line is LDPC encoded and the final matrix $N \times 255$ is the input of the 802.11g transmitter. The $N \times 255$ matrix is split into submatrices with dimensions 48×255 and each submatrix is padded with an extra column with zeros. After the padding, the 256 elements of each row are QPSK mapped into 128 symbols. The result is a 48×128 matrix, which fills the data subcarriers of 128 OFDM symbols. The inverse process is followed at the receiver. The demapped bits are grouped into a $N \times 255$ matrix. The first procedure is the LDPC decoding of each line with the purpose of correcting error bits. The next step is the CRC decoding of each 1×175 LDPC output. As soon as errors are detected the packet is discarded, thus the whole line of the matrix is discarded. Finally, each column of the remaining matrix is Fountain decoded through Message Passing and Gaussian Elimination.

4.2.1 Fountain Encoding

Michael Luby created in 2002 another category of erasure codes, called the Fountain codes [34]. Fountain codes differentiate from other erasure coding schemes in two aspects. The first aspect is that they are rateless codes. The characterization rateless means, that from an initial aggregation of K packets, the number of encoded packets that can be generated is limitless. Moreover, fountain codes are near-optimal for every erasure channel. This means that we can send as many encoded packets as are needed in order to be able to decode them at the receiver, regardless of the statistics of the erasure channel. As long as the number of received packets is slightly larger than the initial number of packets, the decoder will be able to recover the initial packets. Finally, the third characteristic of the Fountain codes is that they require relatively simple encoding and decoding mechanisms.

The Fountain codes that have been used in this thesis are Luby Transform (LT) codes. The decoding process of the Fountain codes is performed through Message Passing (MP) and Gaussian Elimination (GE) [1]. The input of the Fountain encoder is a block of K source packets. On every sampling moment, a random number of packets is chosen and undergoes an XOR operation. The result of the XOR operation on the randomly chosen input packets, is the output of the encoder.

In order to be able to succesfully decode the K source packets, the Fountain encoder should create at least Nov output packets. The number of the output packets Nov is slightly bigger than the source packets, $Nov = (1+\varepsilon)K$ where ε is called the Fountain encoding overhead. According to the literature [13], it is possible to decode Fountain encoded packets with 3% overhead, with the use of MP and GE. Since some of the packets will be lost, if the Fountain encoder generates only Nov output packets the system will fail. In order to avoid retransmissions, the encoder should generate N packets where $N > Nov$. Parameter N is equal to $(1 + ov)K$, where ov is the total overhead of the Fountain encoder. The total overhead value was 46,5%, according to the requirement that both systems should have the same effective throughput. The procedure for the calculation of the total overhead is presented below.

The calculation of the effective throughput of scheme A is based on the need for a feedback channel and retransmissions of packets if the PER exceeds 10%. In scheme A we assume that a packet has 54 bytes [35], thus a PER of 10% corresponds to a BER of $2 \cdot 10^{-4}$. If the BER of the decoded data according to scheme A, under $R_A = \frac{1}{2}$ code rate, exceeds $2 \cdot 10^{-4}$ then there has to be a retransmission. The effective throughput threshold which is imposed by scheme A is:

$$\begin{aligned} Te_a &= (1 - \text{PER})\ R_A\ T_h \\ &= 10.8\text{Mbps} \end{aligned} \tag{4.6}$$

The effective throughput of scheme B, Te_b, should be equal to the effective throughput of scheme A. Te_b depends on the code rate of the scheme R_{OEC}, and on the actual bit rate T_h which is the same as in scheme A. If R_{OEC} is the code rate of the opportunistic error scheme, then in order to have the same effective throughput, R_{OEC} should be equal to $(1 - \text{PER})\ R_A = 0.45$. The R_{OEC} is given by the following equation:

$$R_{OEC} = \frac{k_p\, K}{n_p\, N} \tag{4.7}$$

where k_p is the number of data bits per packet (168), K is the number of input packets for the Fountain encoder (500), n_p is the number of bits with error correction per packet (255) and N is the number of transmitted packets. From the above equation it can be concluded that in order to have such a R_{OEC}, for 500 initial packets, the fountain encoder should produce 732 packets.

The extra 232 packets correspond to 46.5% of the 500 input packets. The encoding process is repeated 168 times since each packet is assumed to have 168 bits.

The result of the encoding process is an output vector T[1, N], and the generator matrix G, with dimensions [K, N]. The number of the rows of matrix G is equal to the number of the initial packets that have to be coded. The number of the columns N, is equal to the number of the total packets that are transmitted. In this case $K = 500$ and $N = 732$. An example of a generator matrix G is depicted on figure 4.4. The receiver is assumed to have knowledge of the generator matrix G. The pseudo-algorithm of the Fountain encoding process is also presented below. The used Fountain encoder was based on Roee Diamant's Fountain encoder[36].

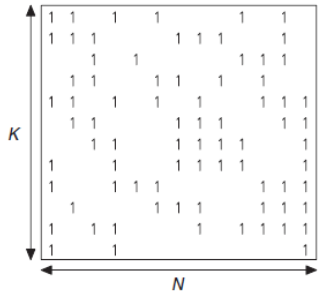


FIGURE 4.4: Example of a generator matrix G[K, N][1]. The elements of each column which are 1 denote the input packets which have to be added modulo-2, in order to produce one output.

Require: Degree distribution ρ, initialized generator matrix G[K, N] to zero.

1: **for** $i = 1, 2, ..., N$ **do**
2: Initialize uniform generator function, $u_i = 0$.
3: Initialize generator matrix column, $G(:, i)$ to zero.
4: Generate degree ρ_i.
5: Choose, uniformly at random ρ_i, distinct input blocks.
6: Perform modulo-2 addition of the ρ_i input packets with indexes which are calculated in the previous step.
7: The output t_i is the result of the modulo-2 addition.
8: Set to one, the elements of $G(:, i)$, with indexes equal to the indexes of the input packets that have been modulo-2 added.
9: **end for**
10: **return** Fountain encoded output vector T[1, N], and the equivalent generator matrix G[K, N].

Algorithm 1: LT encoding

4.2.2 CRC Encoding

The output of the Fountain encoder is a matrix with N rows and 168 columns as it can be seen in the block diagram of figure 4.3. Each one of the rows corresponds to a Fountain packet and each Fountain packet will be assigned to a specific OFDM subcarrier. Before that, the 168 bits of each row have to be CRC encoded. CRC encoding involves inserting some parity bits to the transmitted sequence. With the use of these parity bits which are also called CRC checksum, the receiver can detect the existence of some types of errors at the received message. The CRC checksum which is appended to the transmitted frame, is computed by the division of the input bit sequence with the CRC generator polynomial. The input of the CRC encoder has 168 bits and it corresponds to a polynomial with degree equal to 167 while the CRC generator polynomial has degree equal to 7. The pseudo-algorithm of the CRC checksum generation is presented below. The implementation of the CRC encoding was performed with the Matlab object comm.CRCGenerator.

Require: Input frame I 168×1, CRC generator polynomial 8×1.
1: Left shift I by 8 bits and create I'.
2: Divide I' by the CRC generator polynomial.
3: Obtain the remainder of the division, R.
4: Convert R to the corresponding parity bits, C.
5: Append C to I and create I_{crc}.
6: **return** CRC encoded output vector I_{crc}, 175 x 1.

Algorithm 2: CRC encoding

4.2.3 LDPC Encoding

Low Density Parity Check (LDPC) encoding was first introduced by Gallagher in his PhD thesis [37]. LDPC codes are linear block codes. As linear block codes, each k-bit message vector m is mapped into an n-bit codeword c. This process can be viewed as a mapping from k-space to n-space by a generator matrix G through the relation c = mG. The generator matrix G can be derived from a parity matrix H with dimensions $(n - k) \times n$. The parity matrix H is a sparse matrix as the name low density parity check codes implies. Depending on the number of ones per row and per column the parity matrices can be distinguished into two categories, the regular and the irregular. This characterization is based on the number of non-zero elements per row and per column. The used LDPC code maps a 175 bit input message to a 255 bit codeword and has a regular parity matrix H, with 16 non zero elements per row and per column. The parity matrix H has to reduced to a systematic matrix $H_{sys} = [I_{n-k}|P]$,where I is the identity matrix, through Gaussian elimination and column reordering. Since $GH^T = 0$, the equivalent systematic generator matrix is $G_{sys} = \left[P^T|I_k\right]$. Since the systematic generator matrix has been calculated, the valid codewords c, are given by the relation c = mG_{sys}. Examples of H_{sys} and G_{sys}

are depicted below. The LDPC encoding was performed with the use of Matlab's ldpcencoder object.

$$H_{sys} = \begin{pmatrix} 1 & 0 & 0 & 0\cdots & 0 & p_{1,1} & p_{1,2} & \cdots & p_{1,k} \\ 0 & 1 & 0 & 0\cdots & 0 & p_{2,1} & p_{2,2} & \cdots & p_{2,k} \\ 0 & 0 & 1 & 0\cdots & 0 & p_{3,1} & p_{3,2} & \cdots & p_{3,k} \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & 1 & p_{n-k,1} & p_{n-k,2} & \cdots & p_{n-k,k} \end{pmatrix}$$

$$G_{sys} = \begin{pmatrix} p_{1,1} & p_{1,2} & \cdots & p_{1,n-k} & 1 & 0 & 0 & 0\cdots & 0 \\ p_{2,1} & p_{2,2} & \cdots & p_{2,n-k} & 0 & 1 & 0 & 0\cdots & 0 \\ p_{3,1} & p_{3,2} & \cdots & p_{3,n-k} & 0 & 0 & 1 & 0\cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ p_{k,1} & p_{k,2} & \cdots & p_{k,n-k} & 0 & 0 & 0 & 0\cdots & 1 \end{pmatrix}$$

4.2.4 LDPC Decoding

The LDPC decoding of the input frames of 255 bits, is performed via Matlab's object, ldpcdecoder. The ldpc decoder object implements ldpc decoding through message passing. The number of iterations of the LDPC decoder is 50. In order to be able to perform the LDPC decoding, the previous block which performs the demapping has to convert the demapped symbols into Log-Likelihood Ratio. In order to be able to perform the Log-Likelihood Ratio conversion of the demodulated symbols, it is also necessary to have an estimation of the noise or the interference variance. For this purpose, when the interference signal has bandwidth bigger than 1MHz, the variance of the interference signal is used. If the interference signal has smaller bandwidth, the variance of the added noise is used.

4.2.5 CRC Decoding

Each LDPC decoder output of 175 bits is afterwards CRC decoded with the purpose of detecting LDPC errors. If the CRC decoder detects errors then the equivalent package is discarded. The decoding is also performed via Matlab object comm.CRCDetector. The employed method for detecting transmission errors involves a division. If the polynomial representation of the 175 bit output of the LDPC decoder is P(x), and the CRC generator polynomial is represented as C(x), then the packet does not have any errors if the residual of the division $\frac{P(x)}{C(x)}$ is 0. Depending on the type of the CRC generator polynomial different types of errors can be identified. As far as this thesis is concerned, the CRC generator polynomial is 10001001 and it corresponds to the polynomial $C(x) = x^7 + x^3 + 1$. Table 4.3 summerizes the types of errors that can be detected by the selected CRC generator polynomial [30].

TABLE 4.3: Types of errors that can be detected by $C(x) = x^7 + x^3 + 1$

Error Type	Requirement	Fulfilled
Single Bit Errors	C(x) has at least 2 terms	YES
Two Bit Errors $x^i + x^j$	C(x) does not divide $x^i(x^{j-1} + 1)$	YES
Odd Bit Errors	C(x) = a(x)(x+1)	NO
All Bursts of Errors ≤ 7	C(x)has a constant term	YES

4.2.6 Fountain Decoding

Once the decoder has obtained the vector $T'[1, N']$ where $N' < N$ due to erasures, the purpose is to obtain the initial vector $s[1, K]$, from the relation $T' = sG$. An efficient way to obtain s is through message passing. Algorithm 3, shows the pseudocode for the message passing decoding of LT encoded messages. The received packets T'_i will be refered to as check nodes.

Require: Received vector T', initialized generator matrix G$[K, N]$.
1: Set the variable $finished$ to zero.
2: Create the vector $s[1, K]$ and assign the value 5 to all of its elements.
3: **while** $finished \neq 1$ **do**
4: Locate check node $T'i$ that is connected to only one source packet s_k. Practically this is done by checking if there are any columns of G with only one element equal to 1.
5: **if** Number of check nodes that are connected to only one source packet is zero **then**
6: BREAK
7: **end if**
8: Set s_k equal to $T'i$.
9: Add s_k to the nodes $T'n$ that are connected to s_k.
10: Remove all the edges connected to the source packet s_k. This is done by setting line k of G to zero.
11: Find the packets s_i that haven't been decoded.
12: **end while**
13: **return** Vector $s[1, K]$.

Algorithm 3: Message Passing Decoding

When the number of source packets is small and the requirements of the system impose a small overhead, the message passing algorithm might not be able to fully decode the received packets. In that case, the packets that have not been decoded yet, undergo Gaussian elimination.

Gaussian elimination is a method of solving systems of linear equations Ax = b. If we represent the matrix product Ax as C, then the system of the linear equations can be represented as $[C \mid b]$. Then, with the help of Gaussian elimination with partial pivoting, the matrix $[C \mid b]$ is transformed into the matrix $[C' \mid b']$. Matrix C' is an upper diagonal matrix whose elements $C_{i,j}$ are the elements $a_{i,j}x_{i,j}$ of the initial linear system. The values $a_{i,j}$ can be then calculated

with back-subsitution. The algorithm of Gaussian elimination with partial pivoting is depicted in algorithm 4.

Require: Augmented matrix $D = [C \mid b]$ with dimensions $M \times M + 1$.
1: **for** the first column, $j = 1$ **do**
2: Find the maximum element of the column.
3: Pivot the row with the maximum element to row 1.
4: **end for**
5: **for** every column, $j = 1, 2, ..., M$ **do**
6: **for** every row, $i = j + 1, j + 2, ..., M$ **do**
7: Rearrange rows below the diagonal so according to the criterion $d_{i,j} \geq d_{i+1,j}$....
8: **end for**
9: **if** $D_{j,i} = 0, j = i$ **then**
10: No unique solutions.
11: **end if**
12: **for** $i = j + 1, j + 2, ..., M$ **do**
13: Zero the $D_{i,j}$ element.
14: **end for**
15: **end for**
16: Perform backward subsitution.
17: **return** Solutions of the linear equation system.

Algorithm 4: Gaussian elimination with partial pivoting

4.2.7 Evaluation of the Fountain Code Rate

A really important parameter, that characterizes the Fountain encoder and decoder scheme is the Fountain code rate. The Fountain code rate R_{FC} is :

$$R_{FC} = \frac{K}{Nov} \tag{4.8}$$

This ratio denotes the Fountain overhead which is required, in order to achieve successful decoding of the input packets. The optimal ratio is equal to 0.97, since the minimum overhead that can guarantee successful decoding is 3% when the input packets are 500. The optimal value for the ratio depends mainly on the creation of the Generator Matrix G. Since the optimal Generator Matrix G was not available, it is necessary to evaluate the Fountain code rate of our system. The following procedure was followed for this purpose.

The created packets were transmitted through an erasure channel with 20% erasure probability. When only the message passing algorithm was used for the decoding, the overhead values

correspond to the required overhead which could provide decoding with failure probability 1% according to the specifications [34]. The overhead values for the combination of message passing with gaussian elimination, correspond to the required overhead in order to achieve BER equal to zero. Figure 4.5 shows the overhead threshold, over which the Fountain decoder of scheme B can guarantee successful decoding. The marker in the figure emphasizes that the required overhead for 500 packets is 10%. The conclusion of this procedure is, that the used Fountain decoder can successfully decode the input packets if more than 550 packets survive. The calculated overhead is the worst case overhead of the Fountain decoder, since the channel erasures are random. In case the erasures are not completely random, as in the case of interference at a constant frequency, the specified Fountain scheme can also reconstruct the original data sequence for slightly smaller overhead values, 7.5%. As far as the simulations are concerned, the worst case overhead value of 10% is used.

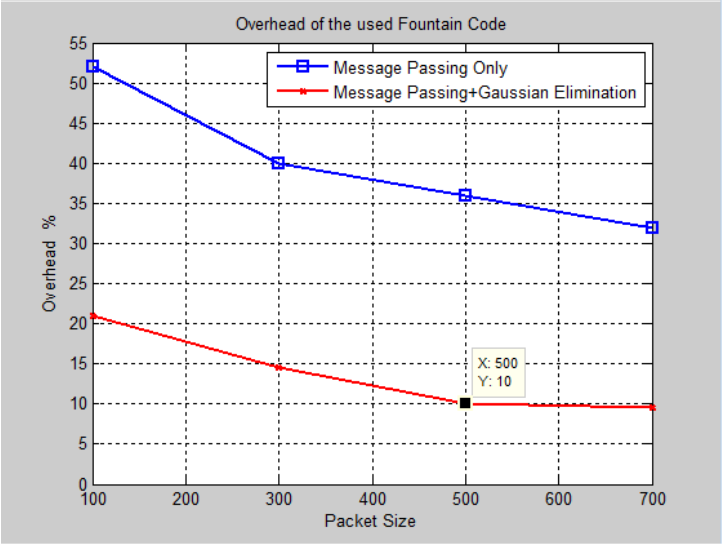


FIGURE 4.5: Scheme B overhead threshold for successful decoding. The combination of message passing and gaussian elimination decoding can successfully decode Fountain packets with 10% overhead. The use of message passing only requires approximately 35% overhead for 500 encoded Fountain packets.

4.3 Simulations

4.3.1 Partial Band Jammer Assumptions

The total bandwidth B_i occupied by the interferer signal, corresponds to a ratio γ of the total occupied bandwidth which is 20 MHz. The upper bound of ratio γ according to [14] is:

$$\gamma \leqq \alpha \left(1 - \frac{R_{OEC}}{R_{FC} R_{EC}}\right) \tag{4.9}$$

In the above equation α is the percentage of the total bandwidth which is used by the user to transmit the data and it is equal to $\frac{48}{64} = 0.75$. Since the Fountain code which was used required a larger overhead than the nominal value of 3%, the equivalent Fountain code rate is $R_{FC} = \frac{500}{550} = 0.909$. Thus, $0 \leq \gamma \leq 0.193$ and the jammer bandwidth is bounded between $0 \leq B_i \leq 3.86$ MHz. The procedure for creating the Partial Band Jammer (PBJ) involves lowpass filtering, of a complex sequence with variance equal to 1. Then, the filtered data, are shifted to the appropriate IF frequency. As far as the simulations are concerned, the PBJ bandwidth is defined by the -3 dB cut-off frequency of the lowpass filter. In the following simulation section, the term PBJ bandwidth corresponds to that definition. Figure 4.6 shows the spectrum of the 802.11g signal interfered by a PBJ signal with 1 MHz bandwidth and a Signal to Interference Ratio (SIR) ratio at the jammed spectrum equal to zero.

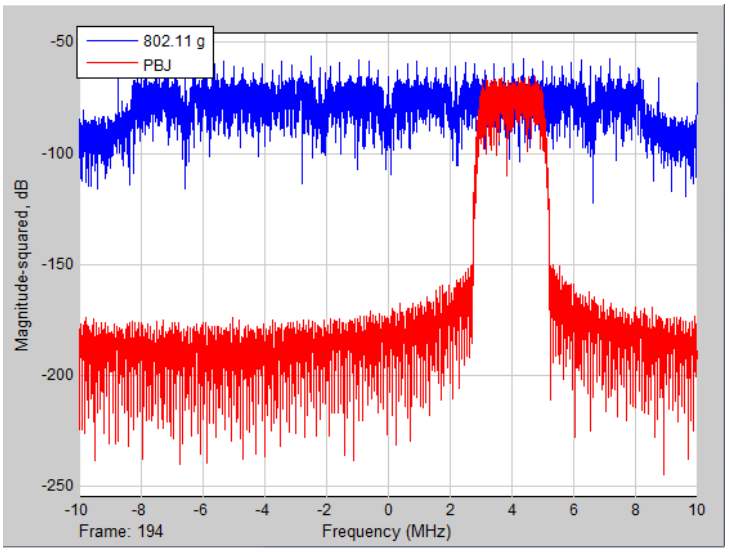


FIGURE 4.6: Example of PBJ with 1 MHz bandwidth and SIR=0. The PBJ signal interferes a 802.11g signal under OFDM modulation, which occupies a 20MHz channel bandwidth.

4.3.2 Partial Band Jammer Frequency

The carrier frequency of the PBJ signal was chosen to be constant, in order to reduce possible errors that occur from the LDPC decoder when frequency hopping is performed. As it was already stated, each row of the 732×168 Fountain output matrix is LDPC and CRC encoded. The reverse process is followed at the receiver, and the LDPC-CRC decoder discards subcarriers that are affected by interference. If frequency hopping is applied, some subcarriers will be partially affected by the PBJ signal. By consequence, the total number of discarded subcarriers does not depend only on the PBJ bandwidth and SIR but also on the format of the OFDM frame, as well as on the timing of the frequency hops versus the beginning of the OFDM frame. This case was not investigated in this Master Thesis but it would be a good idea for future work.

The problem that occurs will be clarified with the use of figure 4.7. Figure 4.7 shows the first 48 rows of the Fountain encoded data. The data of these rows coincide with the subcarrier data of 168 OFDM symbols. Two cases can be distinguished in the matrix of the corresponding figure. The first case represents the effect of PBJ with constant frequency, that affects subcarrier 46. The second case, corresponds to a PBJ which performs frequency hopping during the transmission of the OFDM symbols. In the first case, subcarrier 46 is discarded by the LDPC-CRC decoding scheme. In the second case, subcarriers 2 and 48 are discarded.

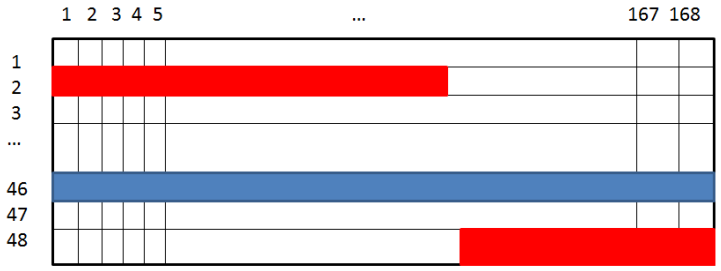


FIGURE 4.7: PBJ effect on the OFDM subcarriers, with and without frequency hopping. Without frequency hopping the interference affects one subcarrier, while with frequency hopping the interference can cause the discarding of two subcarriers.

4.3.3 Simulation assumptions

Summerizing the simulation assumptions, a 802.11g system with encoded data from schemes A and B was partially interfered by a jammer. For every measurement point in the graphs, 50 blocks of data passed through the channel, where each block had $500 \cdot 168 = 84000$ source bits. Thus for each simulation point $500 \cdot 168 \cdot 50 =\approx 4.2$ million source bits were transmitted.

The result of the simulations are the BER plots of schemes A and B versus different bandwidth interferer signals. The evaluation of the BER values will lead to conclusions on the system that achieves throughput optimization. If the BER of scheme A is more than $2 \cdot 10^{-4}$ the equivalent data are not accepted and the transmitter has to send them again. If scheme B succeeds then the BER value is zero. If it fails, the BER value is 0.21 and the transmitter has to send some additional packets. The BER of scheme B is 0.21, because on average if the Gaussian elimination fails, the number of the packets that cannot be decoded is around 42% of the initial packets. Since the encoded data have an equal ratio of 1's and 0's, the BER in this case is around 0.21.

4.3.4 AWGN environment measurements

The first set of measurements was performed in AWGN environment with 20dB SNR. The noise was inserted to the channel before the receiver. Figure 4.8 shows the behavior of these systems for different PBJ bandwidths and SIR equal to 0.

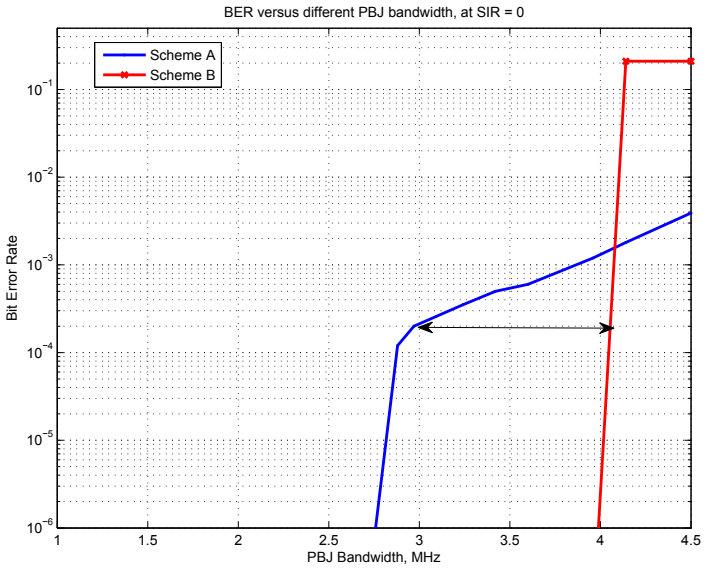


FIGURE 4.8: Performance comparison of schemes A and B over an AWGN environment with 20dB SNR, under the presence of PBJ with variable bandwidth B_i and SIR=0. Scheme B achieves error free communication for interferer bandwidth 4.14MHz, while scheme A can provide error free communication for 3MHz interferer bandwidth. From this figure we can conclude that scheme B outperforms scheme A, for SIR = 0, since it can withstand bigger interferer bandwidth. Alternatively, scheme B can withstand one extra Bluetooth signal with 1MHz bandwidth.

Figure 4.8 denotes, that scheme B has better performance in comparison to scheme A. For a PBJ signal with bandwidth 3 MHz, scheme A exhibits a BER equal to $2.1 \cdot 10^{-4}$ while scheme A has a BER equal to zero. Thus scheme A would require retransmission of the packets, for jammers with bandwidth greater than 3 MHz. On the other hand, scheme B can withstand jammers with bandwidth equal to 4.14 MHz without errors. It is interesting to notice that scheme B achieves successful communication for interferer bandwidth which is bigger than the imposed limit of 3.86 MHz.

The reason for that, is that in this case the lost packets do not have a random distribution. The loss of packets is due to interference only. In that case, the used Fountain scheme can operate with a Fountain code rate of 7.5%. For this code rate, the gamma factor of equation 4.9 is 0.274 and the equivalent PBJ bandwidth is 4.1 MHz. When the PBJ bandwidth is bigger than 4.1 MHz, then the number of packets that arrive at the fountain decoder is smaller than the number it can successfully decode, and this has as a result the failure of Gaussian elimination.

4.3.5 Multipath measurements

The second set of measurements was performed in a 12 tap multipath channel. Both systems are evaluated under the same channel realizations. The mean amplitude of each tap is shown in

figure 4.9 and the distribution of the coefficient amplitude is presented in Appendix A. Each tap introduces a delay of 50ns and the coherence time of the channel is 10ms.

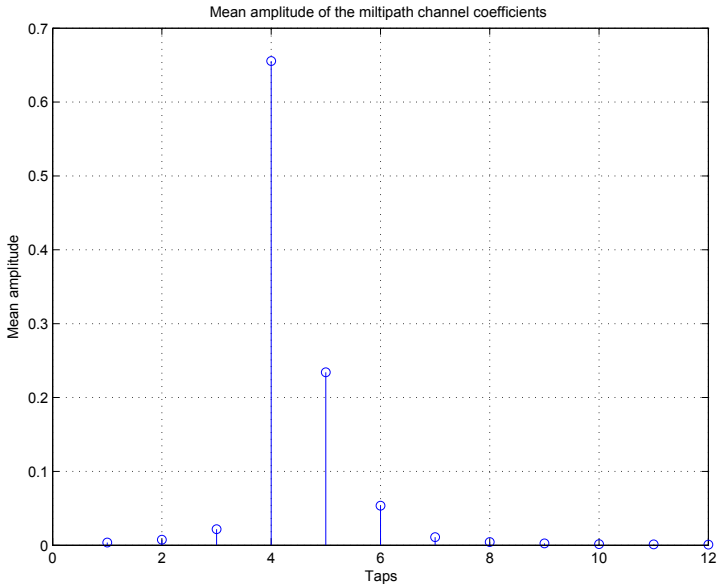


FIGURE 4.9: Mean amplitude of the channel coefficients.

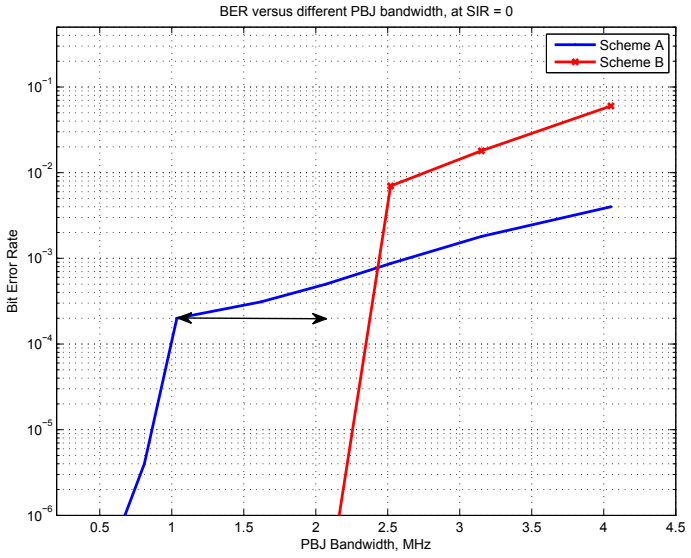


FIGURE 4.10: Performance comparison of schemes A and B over an multipath environment with 20dB SNR, under the presence of PBJ with variable bandwidth B_i and SIR=0. Scheme B achieves error free communication for interferer bandwidth 2MHz, while scheme A can provide error free communication for 1MHz interferer bandwidth. From this figure we can conclude that scheme B outperforms scheme A, for SIR = 0, since it can withstand bigger interferer bandwidth. Alternatively, scheme B can withstand one extra Bluetooth signal with 1MHz bandwidth.

Figure 4.10 shows that scheme B outperforms scheme A in a multipath environment since it can provide successful communication for bigger bandwidth interferers. The actual bandwidth gain depends on the used channel and for this reason a precise value for the bandwidth gain cannot be extracted. In this specific multipath channel, scheme A can support PBJ with bandwidth up

to 1 MHz while scheme B can support PBJ with bandwidth up to 2 MHz. A logical assumption would have been that the bandwidth gain of scheme B would have been larger in the multipath channel than in AWGN environment. The reason for that is that scheme A processes all of the received data, and errors due to multipath fading are present at the decoder output. On the other hand, scheme B would discard the subcarriers that undergo fading and assuming the number of received packets is over the threshold, no errors would be present. Nevertheless, the existence of deep fadings can lead to a significant number of packets to be discarded, and in that case the PBJ bandwidth that can be supported is smaller. In this set of simulations 50 blocks of OEC encoded data were transmitted for each simulation point. Since the channel coherence time was 10ms, each burst went through a different channel realization. The failure of the decoding of one burst could increase the average BER and this could also affect the maximum bandwidth that it can be supported.

4.3.6 Conclusions

It has been shown through simulations, that scheme B which is the OEC scheme, outperforms scheme A in terms of throughput efficiency under PBJ interference at a constant frequency, with SIR=0. The improvement on the throughput efficiency of scheme B derives from the fact that it can withstand PBJ with the same bandwidth as scheme A without the need for packet retransmissions. When packet retransmission are required, the effective throughput of scheme A falls below the imposed limit of 10.8 Mbps.

Nevertheless, although scheme B can avoid packet retransmissions in comparison to scheme A, this does not mean that the use of a feedback channel and the packet retransmission scheme is not useful. A feedback channel can further improve the performance of the OEC scheme. The key factor in the utilization of a feedback channel is the number of required retransmissions. This number can be kept to a minimum with the use of LDPC-CRC decoding, Fountain decoding and a channel estimation block in comparison to scheme A.

Chapter 5

Measurements

5.1 Measurements equipment

5.1.1 Transmitter setup

The 802.11g bursts were created offline with Matlab. The generated data is stored in a file, and with the use of a server in the transmit PC, they were uploaded to the Adlink PCI-7300Aboard5 [38]. The PCI-7300Aboard5 transmitted the data sequence to the AD9761 DAC [39] with a sampling rate of 20 Ms/s. The size of the input file which was uploaded to the Adlink PCI-7300Aboard5 was 16 MB, while the actual size of the 802.11g data sequence was around 12 MB. The remaining space at the data file, was automatically filled with zeros by the server software. The process of uploading the data from the server software was executed continuously in a loop. Figure 5.1, depicts the upload process. In order to be able to assume that the channel is quasi-static, one burst of scheme A encoded data was followed by a burst of scheme B encoded data. Thus, in figure 5.1, burst B1 corresponds to a scheme A encoded burst, burst B2 corresponds to a scheme B encoded burst, etc.

After the DAC, the baseband signal was upconverted to 2.37 GHz with the aid of AD8346 Quadrature Modulator [40] and then it was connected to the transmit antenna. The choice of the specific RF frequency was based on two factors. First of all, it had to be below the 2.4 GHz band in order to avoid interference from other WiFi networks. Moreover, the matching of the used antennas was optimal at 2.37 GHz, thus the power loss at the antenna connections due to reflections was minimum. The transmit and receive antennas were $\frac{\lambda}{4}$ vertical monopole antennas. This choice was based on the possibility of modifying the characteristic resistance of the antenna, by modifying the position of the $\frac{\lambda}{4}$ sized skirts. Figure 5.2 shows a picture of the transmitting antenna and figure 5.4 shows the complete transmitter setup. The numbers in the circles indicate the components of the transmitter setup. At the bottom off the trolley, marked with

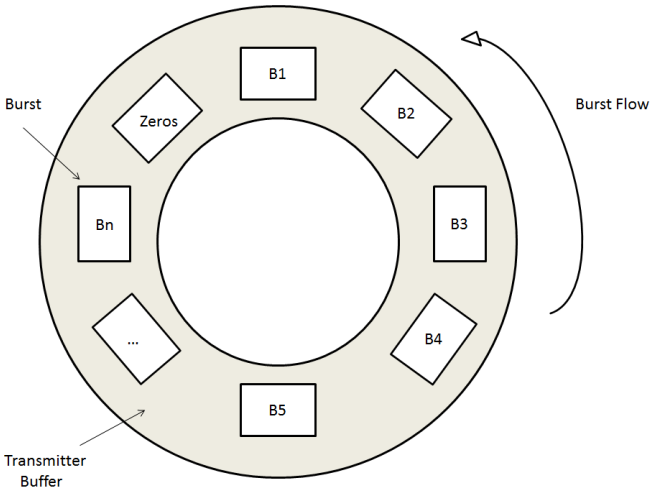


FIGURE 5.1: Schematic of the bursts under transmission sequence.

number 1, is the transmitter PC which has the server software. The component with number 2 is the Adlink PCI-7300Aboard5 and it is connected to the AD8346 Quadrature Modulator(number 3). The RF output of the AD8346 Quadrature Modulator passes through an amplifier (number 4) with 30 dB amplification and the output of the amplifier is driven to the antenna. Figure 5.3 shows the spectrum of the RF 802.11g system before the power amplifier, which was obtained with *Rohde&Schwartz* spectrum analyzer [41]. The whole transmitter topology, as well as the receiver topology was placed on trolleys in order to avoid connection with the antenna through a coaxial cable. In addition to that, both the transmit and receive antennas were placed 1m above the ground.

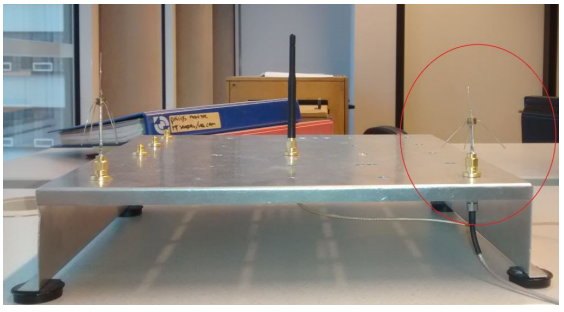


FIGURE 5.2: $\frac{\lambda}{4}$ monopole antenna with grounding plane.

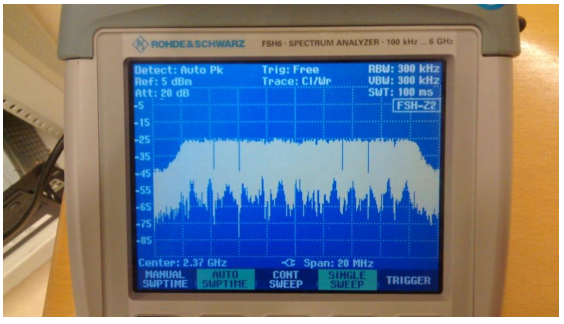


FIGURE 5.3: Spectrum of the RF 802.11g system after the AD8346 Quadrature Modulator.

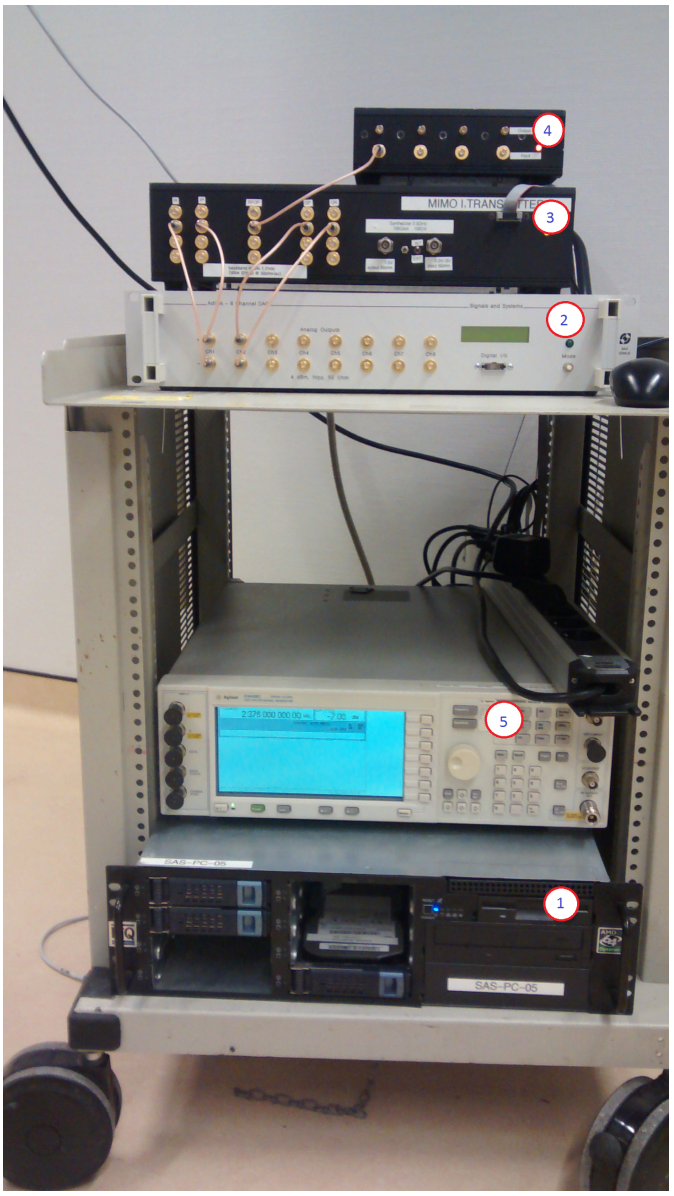


FIGURE 5.4: Transmitter topology for the hardware measurements. Component number 1 is the transmitter PC with the server software. The digital output is driven to component number 2, Adlink PCI-7300Aboard5, which performs DAC and it is upconverted to 2.37GHz by component 3, AD8346 Quadrature Modulator. Before the connection with the antenna, the transmitted signal is further amplified by 30dB, by component 4. Component number 5 is Agilent E4438C ESG Vector Signal Generator which generated the interferer signal.

5.1.2 Receiver setup

The structure of the receiver is similar to the structure of the transmitter. The signal is received from a $\frac{\lambda}{4}$ vertical monopole antenna and it is amplified, before being downconverted by a AD8347 Quadrature Demodulator [**?**]. The quadrature demodulation component, is component number 1 in figure 5.5. The baseband output of the quadrature demodulator is driven to Adlink PCI-7300Aboard5, which is component number 2. It is important to notice that the amplitude

of the baseband input of Adlink PCI-7300Aboard5 can be tuned by a potentiometer which is situated at the input of the board. In figure 5.5 and specifically on component 2, another red circle has been drawn in order to emphasize on 2 led components. These led lights can be either green, orange or red. The color of the led lights denotes the amplitude of the input signal. If the color is green the input signal is around 50% of the maximum input level. If the color is orange, the input signal is around 75% of the maximum input level and if it is red it is over 90% of the maximum input level. The potentiometer gain is modified for every measurement in order to have approximately the same SNR level. The desired input amplitude is around 50% of the maximum input level in order to avoid possible clipping phenomena.

Furthermore, the baseband signal is filtered by a Butterworth filter before the quantization process which is performed through a 12-bit ADC operating at 20Ms/s [42]. As soon as the server software at the receiver PC is triggered, it starts saving the data in a buffer of 32 MB. The size of the buffer at the receiver PC is twice the size of the buffer at the transmitter PC in order to ensure that at least one complete burst sequence will be present at the receiver.

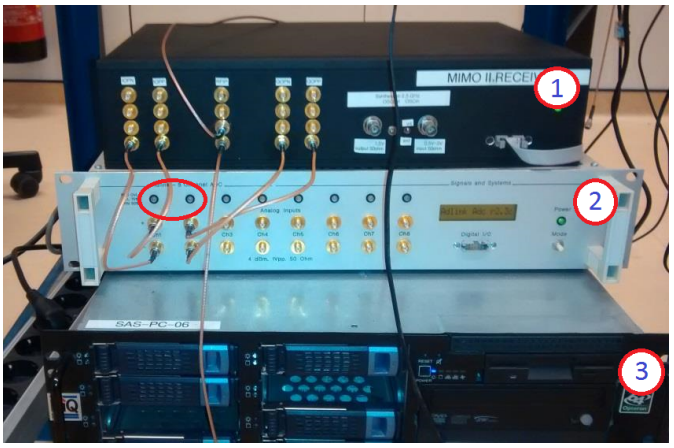


FIGURE 5.5: Receiver topology for the measurements. Component number 1 is AD8347 Quadrature Demodulator. The baseband output of component number 1 is driven to component number 2, Adlink PCI-7300Aboard5, which performs the lowpass filtering and the ADC conversion. Finally, the digital signal is driven to the receiver PC, which is component number 3 for offline processing of the data.

5.1.3 Interferer setup

The Bluetooth interferer signal was generated with the use of Agilent E4438C ESG Vector Signal Generator [43]. The equivalent signal generator is the component with number 5 in figure 5.4. The RF frequency of the interferer signal was 2.376 GHz and it was kept constant. The steps for the generation of the interferer signal through Agilent's E4438C ESG Vector Signal Generator are presented on Appendix B. Figure 5.6 shows the RF spectrum of the transmitted interferer signal.

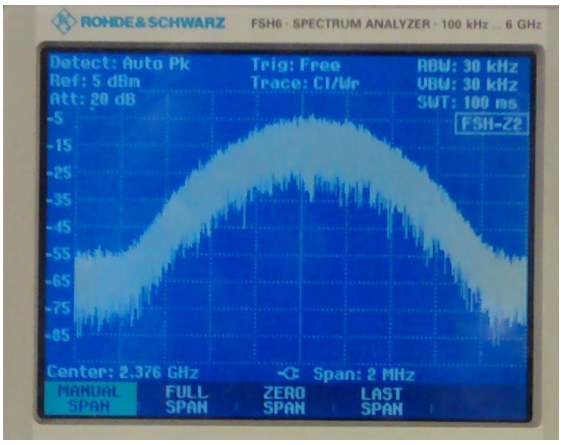


FIGURE 5.6: Spectrum of the interferer signal created by Agilent's ESG Vector Signal Generator.

5.2 802.11g offline receiver topology

The processing of the received data was performed offline as it was already stated. Nevertheless, the schematic of the 802.11g receiver which was presented in chapter 4 had to be modified in order to be able to compensate for the channel distortions and perform accurate timing estimation of the bursts. Figure 5.7 shows the schematic of the modified 802.11g offline receiver. The blocks for OFDM Demodulation, Demapping, Equalization and Scheme Decoding are the same as the ones used in chapter 4. The implementation of the rest of the blocks follows in this section.

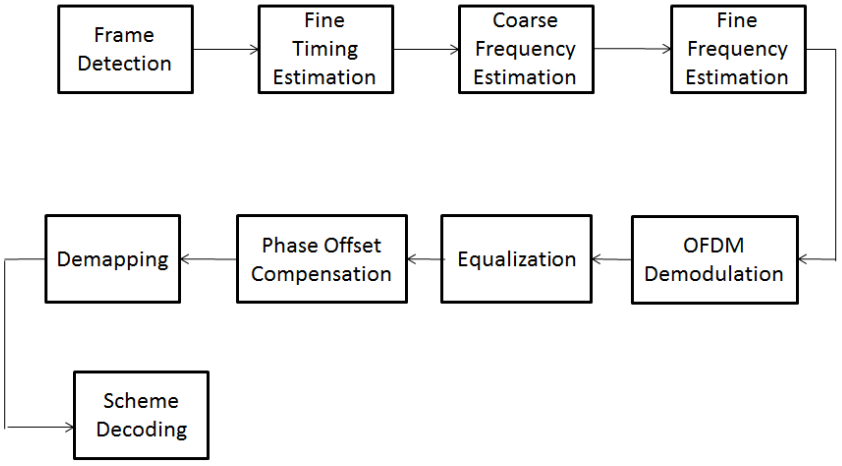


FIGURE 5.7: Schematic of the offline 802.11g receiver. The first task is the detection of a transmitted frame. As soon as a frame is detected, the exact timing of the frame has to be evaluated through fine timing estimation. Since the beginning of the frame has been estimated, the introduced frequency offset is estimated and compensated through coarse and fine frequency estimation processes. Further on, the OFDM symbols are extracted from the OFDM frames, they are OFDM demodulated and the effect of the channel is cancelled through channel equalization. The remaining phase offset of the equalized symbols is corrected and the obtained symbols are QPSK demapped to a bit sequence. Finally, the obtained bits are decoded with the correspondind decoding scheme.

5.2.1 Frame Detection

The first objective of an actual 802.11g receiver is to detect the existence of a transmitted frame. There are multiple ways to detect the beginning of an OFDM frame, such as power detection, power detection using a window and correlation detections. Power detection and power detection through a moving window, compare the obtained power value with a specific threshold. The arrival of a frame can be concluded depending on the value of the power. The correlation method, uses the autocorrelation of the input samples and tries to detect the short preamble of the OFDM frame. When the short preamble is detected, the value of the autocorrelation starts increasing until it reaches a plateau. As far as this thesis is concerned, the detection of the frames was done manually given the fact that the functionalities of the 802.11g receiver are performed done offline. According to figure 5.1, the bursts are transmitted in a loop. As soon as the receiver is turned on, it saves the input data sequences in a buffer. Figure 5.8 shows an example of a received burst sequence. The position of a burst inside the received sequence can be easily found by approximation. Since the amplitude of the short preamble is much smaller compared to the amplitude of the remaining burst, the beginning of the burst is assumed to be 1000 samples before than the observed maximum value. The purpose for performing the frame detection in this way, was the simplicity of the approach as well as the minimization of the computation time.

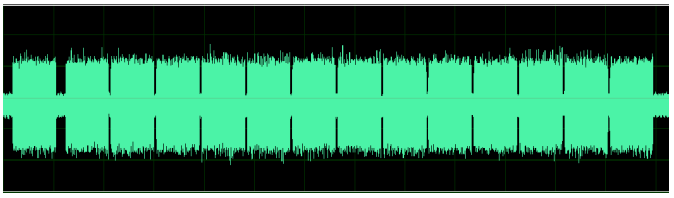


FIGURE 5.8: Received burst sequence. This figure shows the 14 bursts which where transmitted in a loop by the 802.11g transmitter.

5.2.2 Time synchronization

As soon as an OFDM frame has been detected, a sync flag is generated , and the receiver proceeds to the phase of timing estimation. Timing estimation or else timing acquisition is established with the purpose of synchronizing the OFDM FFT window with the received frames. The chosen timing acquisition approach took advantage of the correlation between the CP and the last part of each OFDM symbol [44]. If the received signal has time delay θ and CFO ε, the calculated time delay and the CFO are derived from the maximization of the log-likelihood function. The input of the log-likelihood function is vector r, which consists of a number of input samples of the serial signal x'.

$$\Lambda(\theta, \varepsilon) = |\gamma(\theta)| \cos(2\pi\varepsilon + \angle\gamma(\theta)) - \rho\Phi(\theta) \tag{5.1}$$

$$\gamma(m) = \sum_{k=m}^{m+L-1} r(k) r^*(k + Ns) \tag{5.2}$$

$$\Phi(m) = \frac{1}{2} \sum_{k=m}^{m+L-1} |r(k)|^2 + |r(k + Ns)|^2 \tag{5.3}$$

where Ns is the IFFT length for the OFDM modulation, L is the number of samples of the CP and ρ_{ML} is the magnitude of the correlation coefficient between $r(k)$ and $r(k + Ns)$.

$$\begin{aligned} \rho_{ML} &= \sum_{k=m}^{m+L-1} \frac{E\{r(k) r^*(k + Ns)\}}{\sqrt{E\{|r(k)^2|\} E\{|r(k + Ns)^2|\}}} \\ &= \frac{\sigma_s^2}{\sigma_s^2 + \sigma_n^2} \\ &= \frac{SNR}{SNR + 1} \end{aligned} \tag{5.4}$$

Maximization of the log-likelihood ratio of equation 5.1 requires that the cosine is 1. This leads to the following ML estimation of the CFO:

$$\hat{\varepsilon}_{ML}(\theta) = -\frac{1}{2}\angle\gamma(\theta) + n \tag{5.5}$$

According to [44], the effect of noise can be considered neglectible, $n = 0$, so the CFO formula can be simplified to the following one:

$$\Lambda(\theta, \hat{\varepsilon}_{ML}) = |\gamma(\theta)| - \rho_{ML}\Phi(\theta) \tag{5.6}$$

The joint estimation of θ and ε becomes:

$$\hat{\theta}_{ML} = \arg\left[max_\theta \{|\gamma(\theta)| - \rho\Phi(\theta)\}\right] \tag{5.7}$$

$$\hat{\varepsilon}_{ML} = -\frac{1}{2}\angle\gamma(\hat{\theta}_{ML}) \tag{5.8}$$

The quantities that affect the joint estimation of the time delay and the CFO are the number of CP samples L, and the SNR. The number of the CP samples is already known to the receiver and the SNR can be estimated. Figure 5.9 shows an example of the chosen timing estimation performance. The input signal is an OFDM frame with a preamble which consists of 320 samples and is followed by 20 OFDM symbols. For visibility purposes, only the preamble and the first 7 OFDM symbols are shown. The environment is AWGN with SNR = 20dB and the signal is delayed 20 samples. In addition to that, an additional constant CFO, df = $0.1\Delta_f$ is introduced. The peaks in the plot at samples 322, 420, etc show the estimated beginning of the OFDM symbols. Figure 5.10 shows the estimated CFO at the corresponding estimated sampling moments.

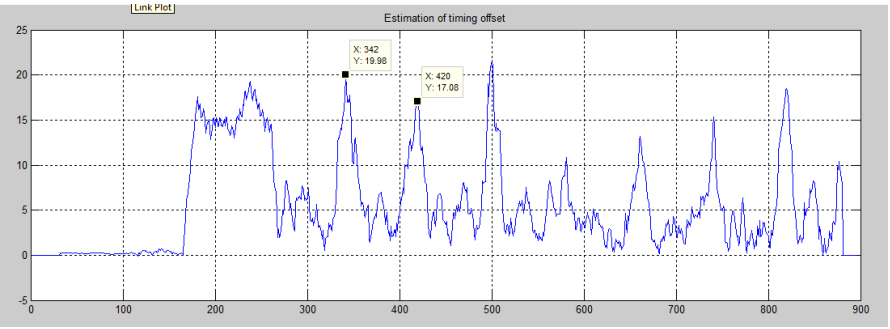


FIGURE 5.9: Timing offset estimation. The distinct peaks correspond to an estimation of the beginning of the cyclic prefix. The first peak occurs at sample 342, the second at sample 420 etc. As far as the measurements were concerned the beginning of each cyclic prefix was set to a threshold value below the peak.

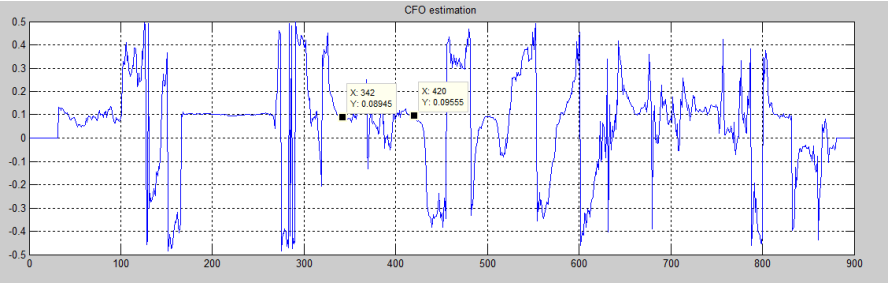


FIGURE 5.10: Constant frequency offset estimation. The estimated frequency offset through the ML algorithm at the estimated times is approximately $0.1\Delta_f$.

As far as the measurements are concerned, the beginning of the OFDM symbol did not occur at the peaks of figure 5.9 but at a threshold value before the peak. This value was first estimated through cable measurements and during the offline processing it was corrected appropriately. The final estimation was based on the variance of the BER of the bursts. The value that lead to the smallest variance was identified as the beginning of the frame.

5.2.3 Coarse frequency offset estimation

The received signal might have a frequency offset due to the local oscillators of the RF transceivers and due to Doppler effects of the channel. The effect of the frequency offset is that, the transmitted energy of subcarrier C_γ is spread to the neighboring subcarriers. The value of the frequency offset defines the amount of energy that is spread to the neighboring subcarriers. This effect can have a detrimental impact on the behavior of our system. The compensation of the frequency offset is done in two stages. At the first stage, a coarse estimation of the frequency offset is performed. As soon as the frequency offset $\hat{d_f}$ has been estimated during the coarse frequency estimation process, the received signal is multiplied with the following term in order to compensate for that.

$$w = e^{-j\frac{2\pi\hat{d_f}t}{64}} \tag{5.9}$$

Although the ML estimation algorithm which was used for the fine timing estimation can provide a coarse frequency estimation, this approach was not not selected. The autocorrelation of short training symbols from the received preamble was chosen instead.

The preamble consists of 10 identical short training symbols. If we assume that the frequency offset is constant, each one of these symbols should exhibit the same frequency offset. The correlation of two short training symbols can provide the total phase shift that they exhibit. In order to do that, a new vector $t = [seq_1\ seq_2]$ is created which consists of two identical short training symbols and which has total length D. The correlation of t is computed according to the following equation:

$$\hat{\mathrm{C}}_{\frac{D}{2}} = \sum_{i=0}^{\frac{D}{2}-1} t_i t^*_{i+16} \tag{5.10}$$

If $C_{\frac{D}{2}}$ is the correlation of the transmitted training symbols, the expected value of the received training symbol's correlation is:

$$E\left\{\hat{\mathrm{C}}_{\frac{D}{2}}\right\} = e^{-j\frac{\pi dfD}{Ns}} C_{\frac{D}{2}} \tag{5.11}$$

Since the transmitted training symbol sequences do not have any phase offset, the argument of the expected value of the correlation can be used for the estimation of the frequency offset $\hat{d_f}$. In that case, the estimated frequency offset is :

$$\hat{d}_f = -arg\left\{\hat{C}_{\frac{D}{2}}\right\}\frac{Ns}{\pi D} \tag{5.12}$$

For the purpose of coarse frequency estimation, 9 correlations were performed and the averaqe value of the phase shift was used for the estimation of the frequency offset. Each correlation occurred between two short training symbols of 16 samples. Figure 5.11 shows the correlation sequence. At each step of the correlation sequence, the receiver performs the following correlation:

$$\hat{C}_{coarse} = \sum_{i=1}^{16} t_i t_{i+16}^* \tag{5.13}$$

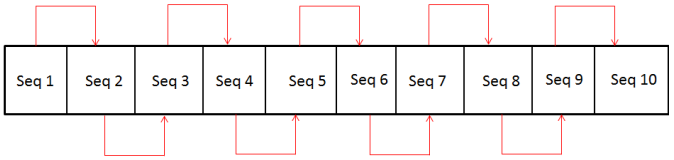


FIGURE 5.11: Correlations between training sequences of 16 samples. The total estimated frequency offset is the average value of the frequency offsets of the 9 correlations.

The coarse estimated frequency offset in this case is

$$\hat{d}_f = -arg\left\{\hat{C}_{coarse}\right\}\frac{64}{\pi 32} \tag{5.14}$$

As soon as the coarse frequency estimation is performed, the correction factor w_1 is applied to the corresponding frame:

$$w_1 = e^{-j\frac{2\pi\hat{d}_f t}{64}} \tag{5.15}$$

5.2.4 Fine frequency offset estimation

The coarse frequency offset estimation process has an estimation error. For this purpose, fine estimation is also required in order to compensate for the residual frequency offset. The training symbols from the long preamble are used for the estimation of the residual frequency offset. The correlation approach is also followed, but this time the correlated symbols have 64 samples length and L becomes 128. The computed correlation at the receiver is:

$$\hat{C}_{fine} = \sum_{i=1}^{64} t_i t_{i+64}^* \tag{5.16}$$

The fine estimated frequency offset in this case is equal to:

$$\hat{d}_f = -arg\left\{\hat{C}_{fine}\right\}\frac{64}{\pi 128} \tag{5.17}$$

After the fine estimation process is finished, correction factor w_2 is applied to the corresponding frame:

$$w_2 = e^{-j\frac{2\pi\hat{d}_f t}{64}} \tag{5.18}$$

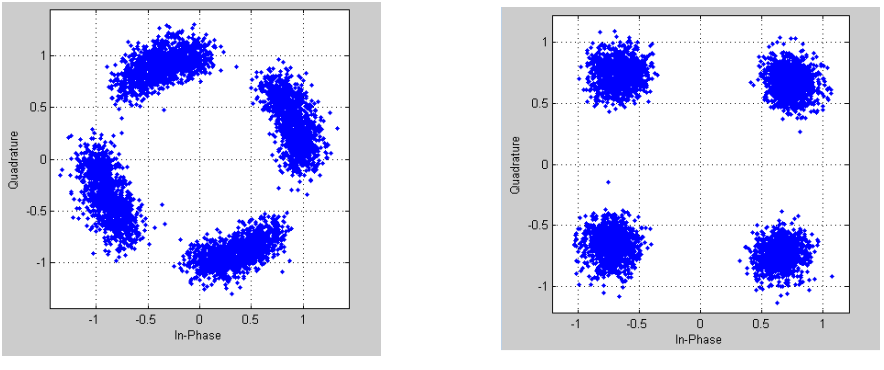
5.2.5 Phase offset compensation

The introduced CFO might not be able to be removed completely by the coarse and the fine estimation processes. This can lead to a residual CFO, which causes a rotation of the subcarriers. The remaining CFO causes a degradation of the BER, especially if a high-order modulation scheme is used, such as 16 QAM, 64 QAM, etc. A simple scheme for the phase offset compensation was used which took advantage of the pilot symbols. If $K_p = \{-21, -7, +7, +21\}$ is the set of the pilot subcarriers which are known to the receiver, the phase distortion is [45]:

$$\gamma_p(t) = \frac{1}{4}\sum_{k\in K_p} \arg\left(S(k), Y(t,k)\right) \tag{5.19}$$

In the above equation $S(k)$ is the transmitted pilot subcarrier $k, \left(k \in K_p\right)$ which is known to the receiver and $Y(t,k)$ is the received pilot subcarrier $k, \left(k \in K_p\right)$ after equalization of symbol t. The phase compensation is then executed according to the following formula:

$$D(t,k) = Y(t,k)e^{-j\,\gamma_p(t)} \tag{5.20}$$



(A) Without frequency compensation (B) With frequency compensation

FIGURE 5.12: Effect of CFO and phase offset on the IQ constellations of the equalized OFDM symbols. The effect of CFO and phse compensation is evident on the second subfigure. The scatter plot of the symbols shows that they have smaller deviation from the mean value.

Figure 5.12 shows the IQ constellations of the equalized symbols, in the case where the RF output of the transmitter is connected via a cable to the RF input of the receiver. The first figure does not include any frequency - phase offset compensation, while the second case involves the implemented frequency and phase compensation algorithms.

5.3 Measurement Procedure

The measurement procedure took place in the RAM laboratory, CR-3.522. In total 18 measurements were performed. Figure 5.13, shows a schematic of the laboratory. The transmitter topology as well as the interferer were placed 2m from the entrance of the lab. The direction of the antennas was modified in order to obtain maximum radiation pattern towards the entrance of lab. The interferer power was tuned, in order to achieve a SIR at position 3 approximately equal to zero. Since the antennas did not have the same characteristic impedance, this way was chosen to tune the interferer power level. The evaluation of the SIR at position 3, was performed with the use of the receive antenna and the spectrum analyzer. The places were the receiver was positioned correspond to the numbered positions in the schematic. The transmission of both signals was constant and upon the trigger of the receiver server software, the input data were saved to the buffer.

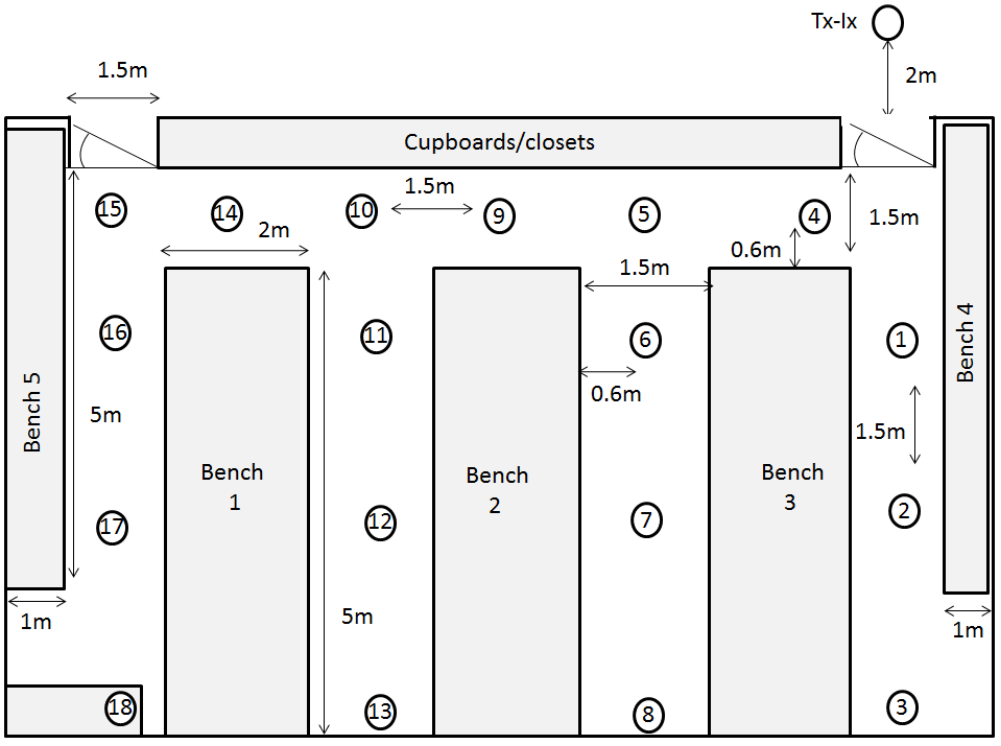


FIGURE 5.13: Schematic of the measurement positions. The 802.11g transmitter along with the interferer were placed 2m away from the lab entrance. The receiver was positioned at the numbered places of the figure. The minimum distance of each measurement position was 1.5m.

5.4 Data Analysis Procedure

Figure 5.8 shows an example of a received burst sequence. As it was already stated, the bursts with odd number correspond to data which are encoded with encoding scheme A. Bursts with even number correspond to data which are encoded with coding scheme B. The bursts with odd number will be referred to as type A bursts, and the bursts with even number as type B bursts. In total, there are 18 received burst sequences, and each burst sequence has 7 type A and 7 type B bursts. As far as type A bursts are concerned, the goal is to obtain the average BER from the 7 bursts. As far as type B bursts are concerned, the goal is to obtain the average number of discarded subcarriers per OFDM symbol from the 7 bursts. Since each Fountain packet encodes one subcarrier the average number of discarded subcarriers per OFDM symbol is equal to the average number of discarded packets per OFDM symbol.

Before obtaining the average BER and the average number of discarded packets per OFDM symbol, it is essential to identify the outlier values. According to [46], an outlier is ”an observation (or subset of observations) which appears to be inconsistent with the remainder of that set of data”. An outlier in this Master Thesis context could exist, if one type A burst has a BER which differs a lot from the BER value of the remaining 6 type A bursts. If this outlier is not excluded, then the average BER could potentially differentiate a lot from the expected value. By consequence, the characterization of the communication as successful or unsuccessful according to the predefined criteria would be wrong.

The criterion which was used for detecting outlier measurements was the modified Thompson τ criterion [47]. According to the Thompson τ criterion, a sample n_i is an outlier if $n_i > \tau\sigma$. Parameter σ is the standard deviation of n samples. For the scope of this thesis the samples correspond either to the BER of type A, or to the number of discarded packets per OFDM sumbol of type B. Parameter τ for n samples, is obtained from the following expression [48]:

$$\tau = \frac{t_{a/2}(n-1)}{\sqrt{n}\sqrt{n-2+t_{a/2}^2}} \tag{5.21}$$

Figure 5.14 shows the procedure for computing the average BER or the average number of discarded packets. Initially the average BER of each burst is computed. The next step is to detect any outliers. In order to have more accurate results, the number of bursts and thus the number of the Modified Thompson algorithm inputs should have been bigger. Alas, the computation time that was required for the data type conversion was more than 1,5 hour. Thus, only 7 bursts were used. These bursts correspond to approximately 560.000 source bits, and around 1.2 million transmitted symbols in the channel. The outlier values are detected with the Modified Thompson algorithm and further on they are removed. In this case the BER value of the 5th

burst is an outlier and it is removed. The average BER which will be used for extracting the conclusions is calculated from the remaining BER values. The same procedure is also followed for evaluating the average number of discarded packets per OFDM symbol.

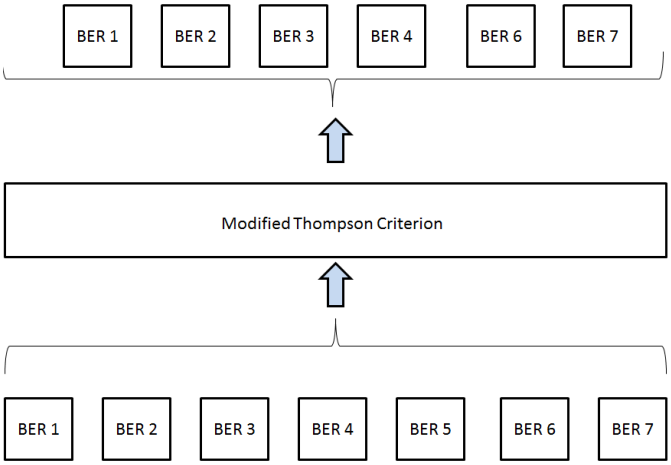


FIGURE 5.14: Outlier removal for the computation of the average BER or average number of discarded packets.

After removing the outliers, the average BER and average number of discarded packets is obtained. If the average BER of type A bursts is smaller than $2\,10^{-4}$ then the communication under scheme A is characterized as successful. As far as scheme B is concerned, a first measure of the success of scheme B is the number of discarded packets per OFDM symbol. According to [14], in order to characterize a transmission under coding scheme B as successful, the maximum number of subcarriers that can be discarded in each OFDM symbol while keeping the same effective throughput requirement is:

$$\begin{aligned} Nd &\leq \left(1 - \frac{R_{OEC}}{R_{EC}R_{FC}}\right) N_{ds} \\ &\leq 0.26 N_{ds} \\ &\leq 12 \end{aligned} \tag{5.22}$$

So, if the average number of discarded packets of type B bursts is below 12, then communication under scheme B is characterized as successful. The number of successful communications of each type, will be used for extracting the conclusions versus different Signal-to-Noise and Distortion (SINAD) ratios. The SINAD ratio is:

$$SINAD = 10 \log \frac{S + Np + I}{Np + I} \tag{5.23}$$

The numerator at the SINAD term corresponds to the combined signal and noise and interference power level ($S + Np + I$). The denominator corresponds to the combined noise and interference power level ($Np + I$). The combined noise and interference level can be obtained by taking the average power of the noise and interference. The measurement of the noise and interference levels occurs after the transmission of the bursts, exploiting the time when the transmitter is transmitting a zero sequence while the interferer is constantly transmitting. The combined signal and noise and interference level is obtained by taking the average power value of each received burst. Figure 5.15 depicts the parts of the received sequence which are used for the calculation of SINAD. The reason for using the SINAD as a parameter for the evaluation of the communication, is the difficulty of estimating the actual SNR of the measurements. The SNR could have been used, if there was an adaptive channel information scheme. This scheme could allow the estimation of the SNR range with and without interference. Unfortunately such a scheme was not implemented, thus an accurate estimation of the SNR could not be provided.

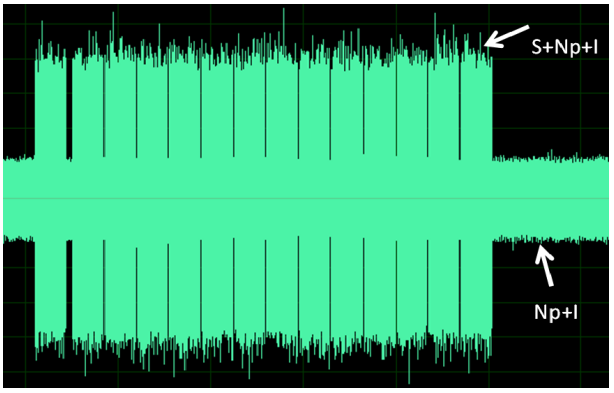


FIGURE 5.15: Data which are used for the calculation of SINAD. The average noise and interference power level was obtained from the samples after the transmission of 14 802.11g bursts. For that reason, the end of the transmitted sequence which was transmitted in a loop, was set to zero. The average power level of the received signal, corresponded to the average signal and noise and interference power level.

The measurement procedure has two goals. The first goal is to evaluate the overall performance of coding schemes A and B over a Bluetooth jammer in terms of throughput optimization. The result of this analysis will be a conclusion on which system is better. The histogram of the relative frequency of successful measurements, versus different SINAD ranges will be used for extracting the conclusions. All of the measurements are taken into account and there is no discrimination between measurements with strong interference or weak interference. The relative frequency of successful communications is defined as the ratio of successful measurements over the total number of measurements which belong to the specific SINAD range.

Figure 5.16 shows the SINAD distribution of the measurements without any additional noise. There are 4 measurements which have a SINAD between $[6, 6.5)$ dB. In that case, the relative frequency of successful communication for a SINAD interval between $[6, 6.5)$ dB is the ratio of successful measurements over 4.

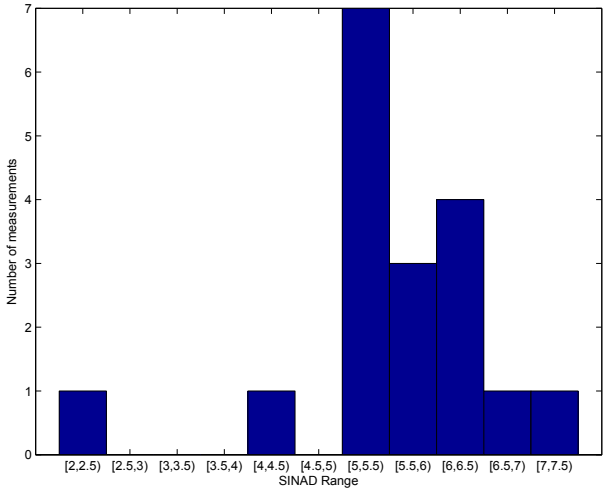


FIGURE 5.16: Distribution of the measurement SINAD.

The second goal of the measurement procedure is to evaluate the relative frequency of successful measurements for different SIR intervals. In total four SIR have been identified, SIR$\in$ $\{[-7,-2.5),[-2.5,0),[0,2.5),[2.5,7)\}$. The relative frequency of successful measurements, versus different SINAD intervals is evaluated for each SIR.

In addition to the distribution of the relative frequency of successful measurements, a curve is also provided, which shows the expected SINAD value, for achieving successful communication on each SIR case. The equivalent curve is produced with the following procedure. The RF output of the transmitter is connected via a cable to the RF input of the receiver. The obtained data via cable are processed offline and a narrowband signal with the characteristics of the interferer signal is added in Matlab. The result of this analysis is to obtain the expected SINAD value which corresponds to a BER of 2×10^{-4} for scheme A. As far as scheme B is concerned the purpose is to find the SINAD value which corresponds to 12 discarded subcarriers per OFDM symbol. The use of these plots can help to evaluate whether the measurement results are cohesive.

5.5 Results

Figure 5.17 shows the overall distribution of the relative frequency of successful communication. As it was already mentioned in the previous section, all of the measurements are used for this calculation. It can be seen from this figure, that the OEC scheme achieves successful communication in a larger SINAD range, since it succeeds for SINAD values bigger than 4.5 dB. On the other hand, the Convolutional Coding scheme with Interleaving requires a SINAD of at least 6.5 dB in order to achieve successful communication. The relative frequency at the SINAD interval $[2,2.5)$ dB is zero since both coding schemes fail. The overall frequency of successful

measurements gives an indication that the OEC scheme outperforms the convolutional coding scheme with interleaving.

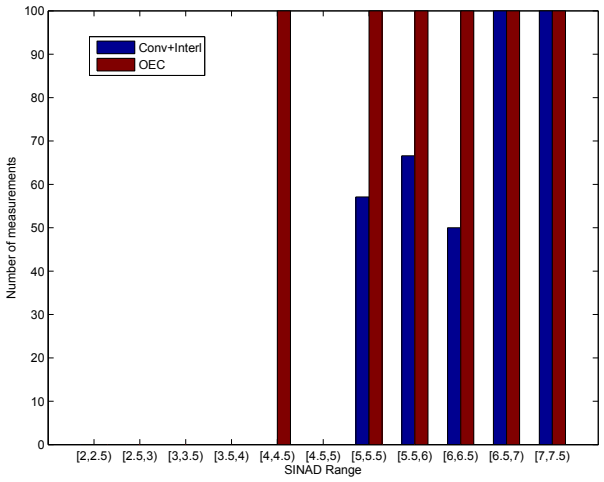


FIGURE 5.17: Overall relative frequency of successful measurements.

The second part of the result analysis, involves evaluating the performance of the coding schemes for each SIR range. Figure 5.18 shows the relative frequency of successful measurements for the cases in which SIR $\in [2.5, 7)$ dB. Figures 5.19 and 5.20 show the calculated threshold values. The measurements which have the equivalent SIR value belong to a SINAD range from $[5, 7)$ dB. Both schemes manage to provide successful communication. The investigation of the threshold values from figure 5.19, shows that in order to achieve successful communication with the Convolutional Coding scheme, the SINAD should be greater than 5.5 dB. Figure 5.20 shows that the SINAD should be greater than 4.7 dB in the OEC scheme.

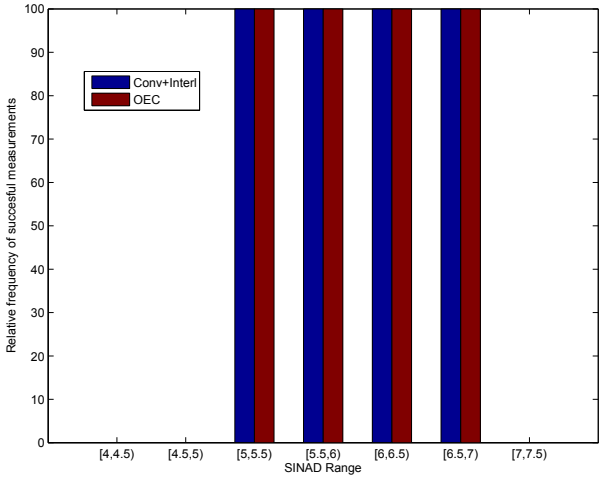


FIGURE 5.18: Relative frequency of successful measurements for SIR $\in [2.5, 7)$.

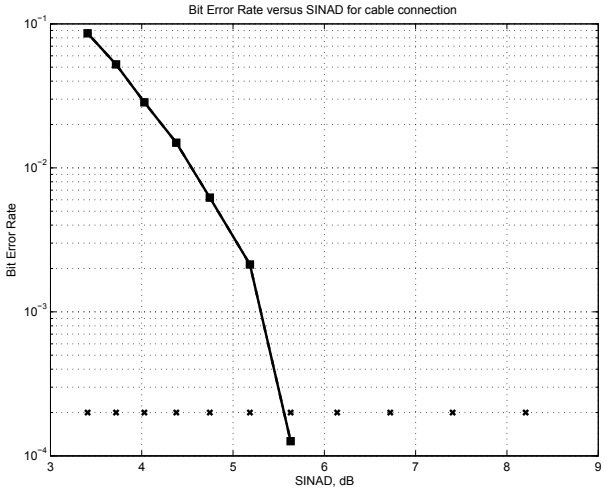


FIGURE 5.19: Evaluation of the threshold value for successful communication, for scheme A and SIR $\in [2.5, 7)$.

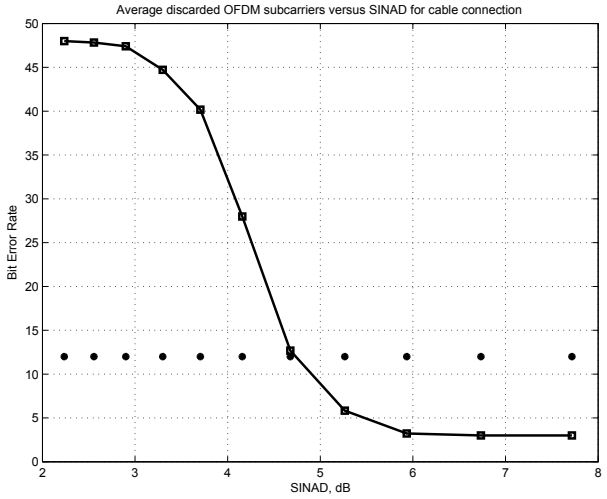


FIGURE 5.20: Evaluation of the threshold value for successful communication, for scheme B and SIR $\in [2.5, 7)$.

Figure 5.21 shows the relative frequency of successful measurements for the cases in which SIR$\in [0, 2.5)$ dB. Figures 5.22 and 5.23 show the calculated threshold values. The measurements which have the equivalent SIR value belong to a SINAD range from $[5.5, 6.5)$ dB. Scheme B, which is the OEC scheme, achieves successful communication in all of the measurements. On the other hand, scheme A achieves successful communication in the SINAD interval $[5.5, 6)$ dB while fails at the intervals $[5, 5.5)$ dB and $[6, 6.5)$ dB.

The behavior of coding scheme A is not as expected. The main reason that can justify such a result, is the small number of measurements. The success of scheme A in the $[5.5, 6)$ SINAD range corresponds to only one measurement. Figure 5.22 also shows, that the Convolutional

Coding scheme requires a SINAD ratio of at least 6.5 dB in order to achieve successful communication. There seems to be an incompatibility between the estimated threshold value and the relative frequency of successful measurements.

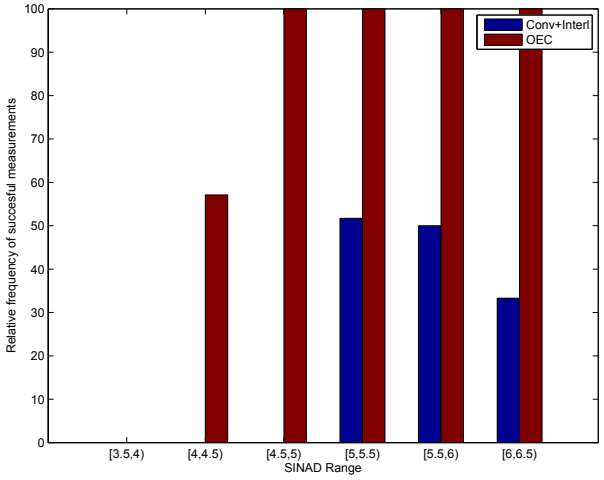


FIGURE 5.21: Relative frequency of successful measurements for SIR $\in$ [0.2.5).

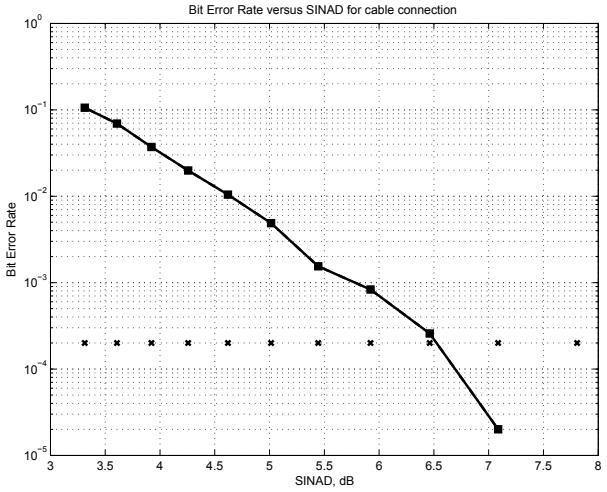


FIGURE 5.22: Evaluation of the threshold value for successful communication, for scheme A and SIR $\in [0, 2.5)$.

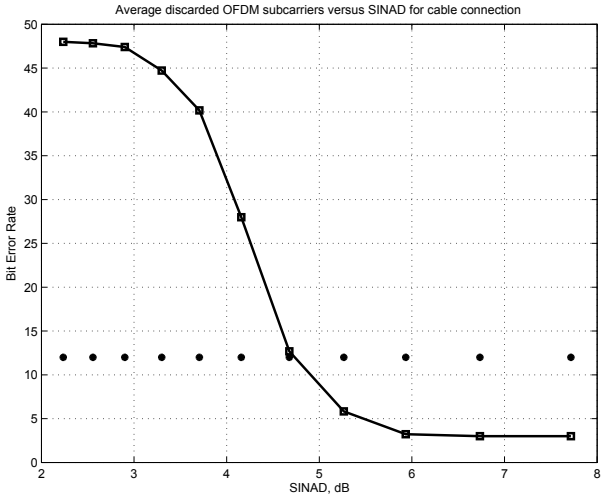


FIGURE 5.23: Evaluation of the threshold value for successful communication, for scheme B and SIR $\in [0, 2.5)$.

Figure 5.24 shows the relative frequency of successful measurements for the cases in which SIR $\in [-2.5, 0)$ dB. Figures 5.25 and 5.26 show the calculated threshold values. The measurements which have the equivalent SIR value belong to a SINAD range from $[4, 5.5)$ dB. Scheme B, which is the OEC scheme, achieves successful communication in all of the measurements. Again, it is interesting to notice that scheme B succeeds in the SINAD interval $[4, 4.5)$. According to figure 5.26, the estimated SINAD value that leads to successful communication is bigger than 4.5 dB. One possible reason for that, is that there is only one measurement in the SINAD interval $[4, 4.5)$.

On the other hand, scheme A fails to achieve successful communication. The reason for the failure of scheme A is the low SINAD of the measured values. According to figure 5.25, scheme A requires a SINAD value bigger than 7.5 dB in order to achieve successful communication, when SIR $\in [-2.5, 0)$ dB. In this case the measured values have a SINAD less than 5.5 dB.

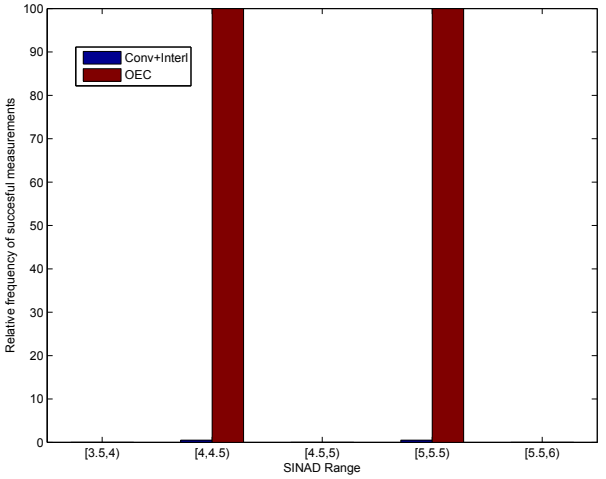


FIGURE 5.24: Relative frequency of successful measurements for SIR $\in [-2.5, 0)$.

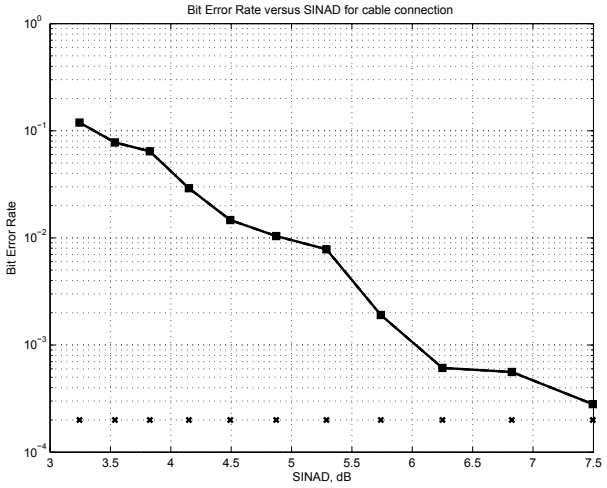


FIGURE 5.25: Evaluation of the threshold value for successful communication, for scheme A and SIR $\in [-2.5, 0)$.

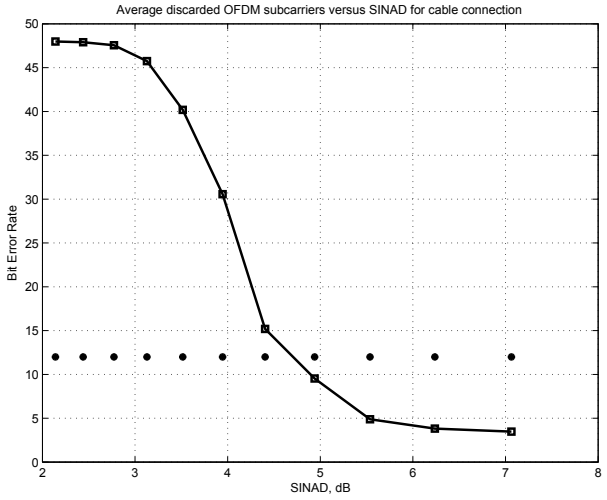


FIGURE 5.26: Evaluation of the threshold value for successful communication, for scheme B and SIR $\in [-2.5, 0)$.

Finally, both coding schemes fail, when the SIR value is approximately -7 dB. This SIR value corresponds to only one measurement and this specific measurement had a SINAD of 2.5 dB. Figures 5.27 and 5.28, depict the estimated behavior of both schemes when SIR -7 dB.

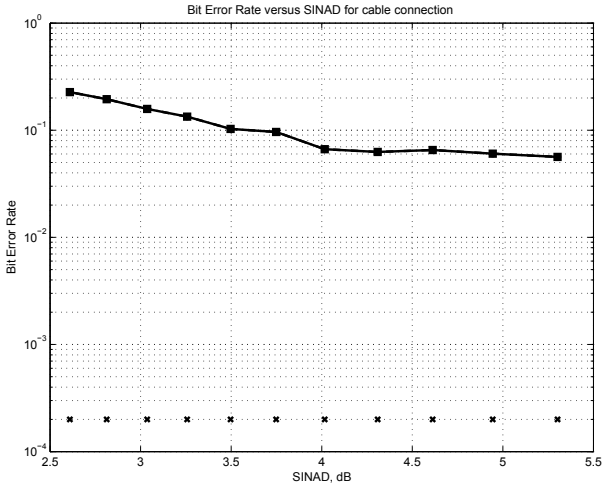


FIGURE 5.27: Evaluation of the threshold value for successful communication, for scheme A and SIR $\in [-7, -2.5)$.

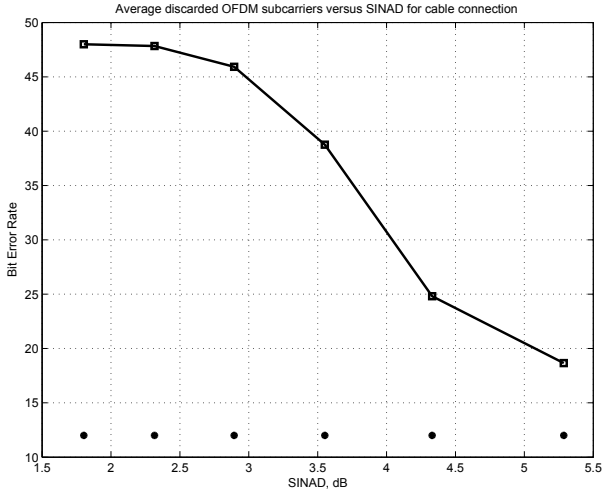


FIGURE 5.28: Evaluation of the threshold value for successful communication, for scheme B and SIR $\in [-7, -2.5)$.

Overall, the OEC scheme managed to provide successful communication under constant frequency Bluetooth interference for the LOS and NLOS measurements. This behavior is according to the expectations and the simulations of Chapter 4. On the other hand, Convolutional Coding and Interleaving scheme did not perform as it was expected. According to the simulations of chapter 4 the specific scheme manages to guarantee successful communication against a PBJ with bandwidth 1MHz in case of flat fading, which is the LOS measurement scenario. Specifically, it failed at 2 out of 3 LOS measurements of figure 5.13. The reason for the failure does not have to do with errors in the timing estimation, frequency estimation and phase offset

compensation blocks or to the IQ imbalance of the measurements, since the same procedure was followed for the OEC scheme and it performed as expected. The problem was finally situated at the interleaver of the scheme. A mistake had been made which deteriorated the performance of the scheme. Figures 5.29 and 5.30 show the effect of this intearleaver mistake. A PBJ signal at a constant frequency with SIR = 0 interferes the 802.11g system and AWGN with various levels is added. Figure 5.29 shows the expected behavior with the correct interleaver of Chapter 4. Figure 5.30 shows the behavior of the system with the measurements interleaver. It can be seen that the behavior of the used interleaver deteriorates the system performance under a LOS scenario.

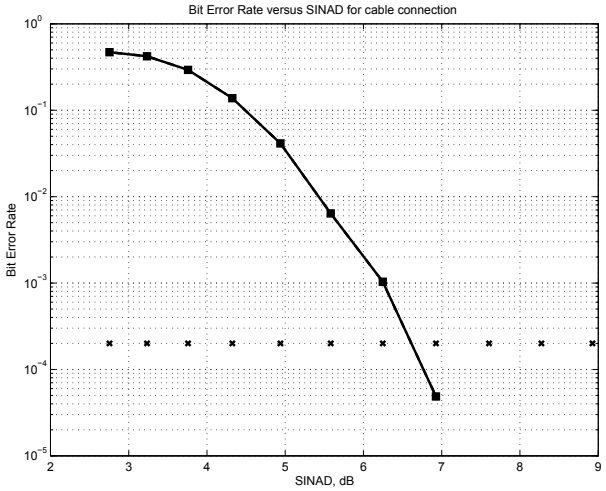


FIGURE 5.29: BER of scheme B for a PBJ at a constant frequency, with SIR = 0 and AWGN.

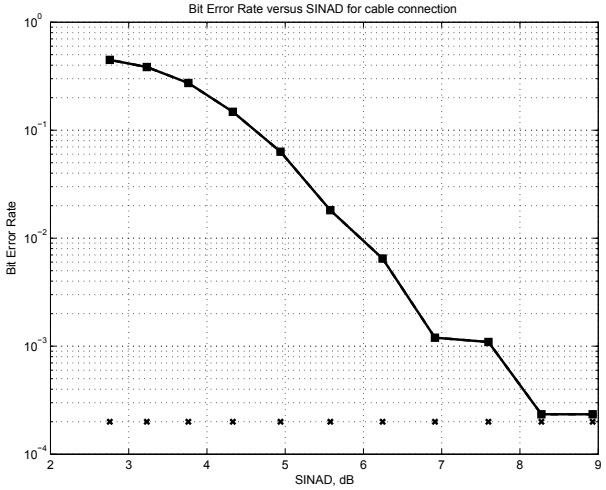


FIGURE 5.30: BER of scheme B for a PBJ at a constant frequency, with SIR = 0 and AWGN. The interleaver of this figure was the interleaver of the measurements on which there was a mistake. The behavior of scheme A is worst than the expected one. The optimal behavior is depicted on figure 5.29.

5.6 Conclusions

A first conclusion of the measurements procedure is that the used OEC scheme manages to provide successful communication to a 802.11g system, which is interfered by a Bluetooth signal transmitting at a constant frequency. The OEC scheme had a 100% successful communication rate, excluding one measurement which could not be processed. In detail, the OEC scheme managed to guarantee the communication under the following SIR levels:

$\text{SIR} \in \{[-2.5, 0), [0, 2.5), [2.5, 7)\}$.

The results concerning the success of the OEC scheme although they were successful, they are just an indication. In order to be able to prove through measurements, that the OEC scheme indeed manages to deal with a Bluetooth signal transmitting at a constant frequency, a larger number of measurements is required. A larger number of measurements would allow a more accurate statistical analysis of the received data.

As far as the FEC scheme with interleaving is concerned, the mistake that was made does not allow to perform a fair comparison of the two schemes. The interleaver problem affects the behavior of the scheme under interference and the specific implementation has worst results than expected. The timing, frequency offset estimation and phase compensation of the samples is performed in a similar way for both schemes and it did not affect the behavior of the FEC with interleaving scheme under interference.

Chapter 6

Conclusions

6.1 Conclusions

This thesis has as main topic the investigation of methods that will help wireless transmitting systems improve their throughput under wireless interference. The main question can be answered through the investigation of the secondary questions.

The first secondary question involves the evaluation of the modulation sensitivity towards interference, defined as Protection Ratio, which is required in order achieve communication with BER = 0.001. The equivalent ratio values were obtained for the case of 802.11g system with 16 QAM modulation, 802.11g system with QPSK modulation, and Bluetooth system, through Simulink simulations. Each system was evaluated versus the other systems and versus the same system. In addition to that, the Protection Ratio was also evaluated for the case of adjacent channel interference from a 802.11g system. The Protection Ratio values were obtained, at the bandpass range of 200 MHz for the case of an ideal environment.

The result of these measurements is the sensitivity of each investigated modulation system against narrowband or broadband interference. The sensitivity results are depicted in a matrix which is called the Interference Matrix. If the modulation scheme of the interfered system can be detected, then with the help of the sensitivity matrix, the interfering system can make the appropriate decisions in order reduce its interference footprint. The reduction of the interference footprint will lead to the reduction of the spatial overlap between two systems.

The sensitivity matrix, is only one factor that affects the interferer's behavior and in general, the coexistence of wireless systems in the LE spectrum according to [11]. Other parameters are also evaluated in order to provide fair sharing of the spectrum. By consequence, these measurements can be considered as one input to a multiple input process, which aims to help at a fair sharing

of the LE spectrum. Under this scope, the throughput of wireless systems under interference can be improved.

The second subquestion aims to investigate the performance of two channel coding techniques versus narrowband interference. The investigated coding schemes are the Opportunistic Error Correction scheme and the Convolutional Coding with Interleaving scheme. The Opportunistic Error Correction scheme operates requires a 10% overhead in order to successfully decode the packets. The Convolutional Coder used the generator polynomials [133, 171] and the decoding was performed via a Viterbi hard decoder. The previous schemes are compared on the basis of the same effective throughput. The OEC scheme outperformed the Convolutional Coding with Interleaving scheme under constant PBJ carrier frequency. Specifically, the OEC scheme can withstand Partial Band Jammers with larger bandwidth, at a SIR =0 dB, for the cases of flat and multipath fading. The actual bandwidth gain of the OEC scheme can be further improved since the optimal Fountain overhead value is smaller.

Finally, the last subquestion examines the behavior of the two coding schemes under a Bluetooth interferer which has continuous transmission at a constant frequency. Again the purpose is to identify the system which performs better under the same effective throughput requirement. In total 18 measurements were performed in the laboratory of the RAM group. A conclusion cannot be made about the superiority of one scheme since there was a mistake in the interleaver of scheme A and time was not enough to repeat the measurements. Nevertheless a conclusion can be made about the OEC scheme. As it was expected, the OEC managed to provide successful communication under a Bluetooth jammer signal, which was transmitting constantly and at a fixed frequency. If we exclude one measurement that could not be processed for both schemes, the OEC scheme managed to provide successful communication in all of the other measurements. The average number of lost packets per OFDM symbol was below 12 for all of the measurements.

Overall, the answer to the main question, which is how we can improve the throughput of a wireless system under interference, was answered through the results of the subquestions. If we follow a spectrum management approach, the specification of the interferer matrix can be an input to the algorithm that aims to provide fair sharing of the LE spectrum. If we follow the information theory approach, the use of the OEC scheme is better than the CC scheme with hard decoding, when dealing with narrowband interference at a constant frequency. With this way, the need for retransmissions is minimized, and the throughput of the system is improved.

6.2 Recommendations

6.2.1 Recommendations on the Protection Ratio Simulations

As far as the modeling of the Bluetooth and 802.11g systems in Simulink is concerned, the chosen implementation demanded too much computation time. This was the bottleneck of this part of the research. A simpler approach could be used instead, to model the interferer. Narrowband or broadband noise with the appropriate spectral characteristics can be used instead of an OFDM system or a Bluetooth system. Since the measured value is the average power of the interferer signal, the results will not be affected. Another alternative which could minimize the computation time, would be to implement the systems in a programming language, such as C++, which can achieve fast computation time. Moreover, since the 802.11g and Bluetooth systems are already developed, it is easy to investigate the sensitivity of other types of modulation.

As far as the complexity of the chosen systems is concerned, the complexity of the simulated 802.11g transceiver could be extended, adding other functionalities of the MAC layer, such as timing estimation, frequency and phase compensation. It would be useful to evaluate the impact of interference on the frame detection and fine timing estimation of the 802.11g receiver. Specifically, it is interesting to investigate the levels of interference power that can cause the failure of the time estimation at the receiver.

Moreover, the impact of interference on the phase compensation of the receiver, in the case the interference affects the pilot tones can be also simulated. This particular research would be really useful when a high order modulation scheme is employed such as 64 QAM. Mistakes in the phase estimation and correction could have a detrimental effect on the BER value of the high order modulation scheme.

From the Bluetooth system perspective, it would be also interesting to investigate the scenario in which the Bluetooth transceiver employs adaptive frequency hopping.

6.2.2 Recommendations on the Channel Coding Simulations and Measurements

The complexity of the specific OEC scheme can be extended with a channel state information block. The channel estimation scheme is already used in [13], but its role can be further enhanced. In [13] a channel estimation scheme is used in order to reject the subcarriers with low SNR. When the goal is to combat deep fading and to mitigate interference, the information of the channel estimation block can be also used to discard subcarriers with high SIR.

In addition to that, the performance of the above coding schemes could be also evaluated versus frequency hopping PBJ. This research requires the implementation of a MAC frame according to

the 802.11g specifications. The extracted results correspond to specific MAC frame dimensions. This process can be repeated for multiple PBJ interferers which perform the frequency hopping at different times.

In addition to that, the success of the Convolutional Coding and Interleaving scheme can be also evaluated for larger interleaver matrix dimensions. In this implementation, the size of the interleaver is 96. Changing the size of the interleaver also has an impact on the interleaver gain over narrowband interference. Apart from that, adding an extra permutation to the interleaver, which would introduce time interleaving would be also interesting.

And finally, the behavior of the system under soft Viterbi decoding could be also investigated. The expectations are, that the soft decoding scheme will outperform the hard decoding scheme. The only difficulty in soft decoding, is the estimation of the noise variance along with the interference variance, which is required in order to obtain the Log-Likelihood ratio representation of the demodulated symbols. This problem exists also at the OEC scheme since the LDPC decoder expects the input data to be in Log-Likelihood format. In this case an assumption was made on the variance of the interference. When the PBJ signal had a bandwidth bigger than 1 MHz, the variance of the noise was used for the Log-Likelihood estimation. When the PBJ signal had a bandwidth which was bigger than 1 MHz, the variance of the interfering signal was used for the estimation.

Since there was a mistake in the interleaver which was used for the measurements, and this mistake deteriorated the performance of the FEC with interleaving scheme it is not possible to extract conclusions about the superiority of a scheme. Thus, a certain recommendation would be to repeat the measurement procedure. Given the fact that the space where the measurements were performed allows a small number of measurements, two suggestions can be made. The first suggestion is to repeat the measurement procedure in a space where it will be possible to obtain at least 40 measurements. A bigger number of measurements would give more accurate estimation of the statistical properties of the measurements. In case the space is again limited, different sets of measurements can be performed at the same positions, but with variable interferer power this time. This approach could provide a more accurate distribution of the received SINAD levels. This could allow us apart from verifying the superiority of one system, also to quantify the system gain.

Bibliography

[1] D. MacKay, "Fountain codes," *IEEE Proceedings on Capacity Approaching Codes Design And Implementation Special Section*, 2005.

[2] IEEE, *IEEE Std 802.11g*, 2003.

[3] ——, *IEEE Std 802.15.1*, 2003.

[4] A. S. Tanenbaum, *Computer Networks*. Prentice Hall, 2003.

[5] M. Buddhikot and K. Ryan, "Spectrum management in coordinated dynamic spectrum access based cellular networks," *IEEE New Frontiers in Dynamic Spectrum Access Networks*, 2005.

[6] J. Motola and al, "Cognitive radios- making software radios more personal," *IEEE Pers. Comm*, vol. 6, no. 4, Aug 1999.

[7] D. J. MacKay, *Information Theory, Inference and Learning Algorithms*. Cambridge University Press, 2013.

[8] C. Berrou, A. Glavieux, and P. Thitimajshima, "Near shannon limit error-correcting coding and decoding: Turbo-codes," *Proceedings of IEEE International Communications Conference*, 1993.

[9] K. S. Zigangirov, *Low-Density Parity-Check Codes*. Wiley-IEEE Press, 2004.

[10] B. D. V. Veen and K. M. Buckley, "Beamforming a versatile approach to spatial filtering," *IEEE ASSP Magazine*, April 1988.

[11] B. A. Witvliet, M. J. Bentum, R. Schiphorst, and C. H. Slump, "Medium usage model for the design of dynamic spectrum management in ISM bands," *IEEE International Conference on Communications*, 2012.

[12] B. Witvliet, M. J. Bentum, R. Schiphorst, and C. H. Slump, "Dynamic spectrum management: Social behavior for radio systems," *CTIT Symposium: ICT: The Innovation Highway*, 2012.

[13] X. Shao, R. Schiphorst, and C. H. Slump, "An opportunistic error correction layer for OFDM systems," *EURASIP Journal on Wireless Communications and Networking*, 2009.

[14] X. Shao and C. H. Slump, "Co-channel interference cancelation : Cross coding vs beamforming," *IEEE International Conference on Communications (ICC)*, pp. 5–9, 2011.

[15] J. G. Proakis and M. Salehi, *Digital Communications*, 5th ed. McGraw-Hill, 2008.

[16] R. Gallager., *Theory of Code Division Multiple Access Communication*. MIT, 1963.

[17] S. Weinstein and P. Ebert, "Data transmission by frequency-division multiplexing using the discrete fourier transform," *IEEE*, vol. 19, pp. 628–634, Oct 1971.

[18] J. G. Proakis and D. G. Manolakis, *Digital Signal Processing*, 4th ed. Pearson International, 2007.

[19] A. Peled and Ruiz, "Frequency domain data transmission using reduced computational complexity algorithms," *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, vol. 5, p. 964 –967, Apr 1980.

[20] M. Joost, *Theory of Root Raised Cosine Filter*. Krefeld DE, 2010.

[21] S. Hayikn, *An introduction to analog and digital communications*. John Wiley and Sons, 1989.

[22] Golmie.N, Rebala.O, and Chevrollier.N, "Bluetooth adaptive frequency hopping and scheduling," *IEEE Military Communications Conference*, Oct 2003.

[23] R. Schiphorst, F. Hoeksema, and K. Slump, *Bluetooth demodulation algorithms and their performance*. 2nd Karlsruhe Workshop on Software Radios, 2002.

[24] M. Nafie, A. Gatherer, and A. Dabak, *Decision Feedback Equalization For Bluetooth Systems*. Texas Instruments.

[25] R. Schiphorst, F. Hoeksema, and K. Slump, "A (simplified) bluetooth maximum a posteriori receiver," *IEEE Workshop on Signal Processing Advances in Wireless Communications*, 2004.

[26] F. Xiong, *Digital Modulation Techniques*. Artech House Telecommunication Library, 2010.

[27] M. K. Simon and M. S. Alouini, *Digital Communication over Fading Channels – A Unified Approach to Performance Analysis*, 1st ed. Wiley, 2000.

[28] *IEEE Std 802.11.a : High-Speed Physical Layer in the 5 GHz band*, 2003.

[29] S. Selby, A. Amini, and C. Edelman, *Simulating Interference Issues between Bluetooth PANs and 802.11 b and 802.11g WLANs*. Agilent Technologies.

[30] MIT, *MIT 6.02 Lecture Notes*, 2010.

[31] J. Mikulka and S. Hanus, "Bluetooth and IEEE 802.11b/g coexistence simulation," *Radioengineering*, vol. 17, no. 3, 2008.

[32] Matlab, "Error detection and correction." [Online]. Available: http://www.mathworks.nl/help/comm/ug/error-detection-and-correction.html

[33] Z. Zhang, B. Wu, Y. Zhou, and X. Zhang, "Low-complexity hardware interleaver/deinterleaver for ieee 802.11a/g/n wlan," *VLSI Design*, vol. 2012, 2012.

[34] M. Luby, "Lt codes," *Proc. 43rd Ann. IEEE Symp. on Foundations of Computer Science*, p. 271–282, 2002.

[35] A. Doufexi, S. Armour, M. Butler, A. Nix, D. Bull, and J. McGeehan, "A comparison of the hiperlan/2 and ieee 802.11 a wireless lan standards," *IEEE Communications Magazine*, vol. 17, no. 3, p. 172, July 2002.

[36] R. Diamant, *Raptor and Fountain Codes*. University of British Columbia.

[37] R. G. Gallager, *Low-Density Parity-Check Codes*. MIT Press, 1963.

[38] Adlink, *Adlink, 80 MB/s High-Speed 32-CH Digital I/O PCI Card*.

[39] AnalogDevices, *AnalogDevices, 10-Bit, 40 MSPS, dual Transmit D/A Converter*.

[40] Analog-Devices, *AnalogDevices , 800 MHz to 2.7 GHz RF/IF Quadrature Demodulator*.

[41] Rohde-Schwartz, *Rohde-Schwartz, FSH6 Spectrum Analyzer*.

[42] AnalogDevices, *AnalogDevices, Dual 12-Bit, 20/40/65 MSPS, 3V A/D Converter*.

[43] Agilent, *Agilent Technologies E4428C/38C ESG Vector Signal Generators, User's Guide*.

[44] J. van de Beek, M. Sandell, and P. BÄorjesson, "ML estimation of time and frequency offset in OFDM systems," *IEEE Transactions on Signal Processing*, vol. 45, no. 7, July 1997.

[45] K. Akita, R. Sakata, and K. Sato, "A phase compensation scheme using feedback control for ieee 802.11a receiver," *IEEE 60th Vehicular Technology Conference*, vol. 7, no. 0, pp. 4789–4793, 2004.

[46] V. Barnett and T. Lewis, *Outliers in Statistical Data*.

[47] R. Dieck, *Measurement Uncertainty, Methods and Applications*.

[48] M. Anbarasi, S. Ghaayathri, R. K. , and I. Abirami, "Outlier detection for multidimensional medical data," *International Journal of Computer Science and Information Technologies*, vol. 2, no. 1, pp. 512–516, 2011.

Appendix A

Data

A.1 Software Measurements against variable bandwidth PBJ

TABLE A.1: Coding schemes BER in case of jammer with SIR = 0 and variable bandwidth in a flat fading environment with 20dB SNR.

SIR	Scheme A BER	Scheme B BER	Bw Jammer (MHz)
0	0	0	0.9
0	0	0	1.15
0	0	0	1.8
0	0	0	2.7
0	$1.2e^{-4}$	0	2.9
0	$2.0e^{-4}$	0	2.97
0	$3.5e^{-4}$	0	3.24
0	$5.0e^{-4}$	0	3.42
0	$6.0e^{-4}$	0	3.6
0	$1.2e^{-3}$	0	3.96
0	$1.8e^{-3}$	0	4.14
0	$4.0e^{-3}$	0.21	4.5

TABLE A.2: Coding schemes BER in case of jammer with SIR = 0 and variable bandwidth in a multipath fading environment with 20dB SNR.

SIR	Scheme A BER	Scheme B BER	Bw Jammer (MHz)
0	0	0	0.45
0	0	0	0.81
0	$2.0e^{-4}$	0	1.035
0	$3.1e^{-4}$	0	1.62
0	$5.0e^{-4}$	0	2.07
0	$8.7e^{-3}$	$7.0e^{-3}$	2.52
0	$1.8e^{-3}$	$18.0e^{-2}$	3.15
0	$4.0e^{-3}$	$60.0e^{-2}$	4.05

A.1.1 Distribution of the multipath coefficients amplitude

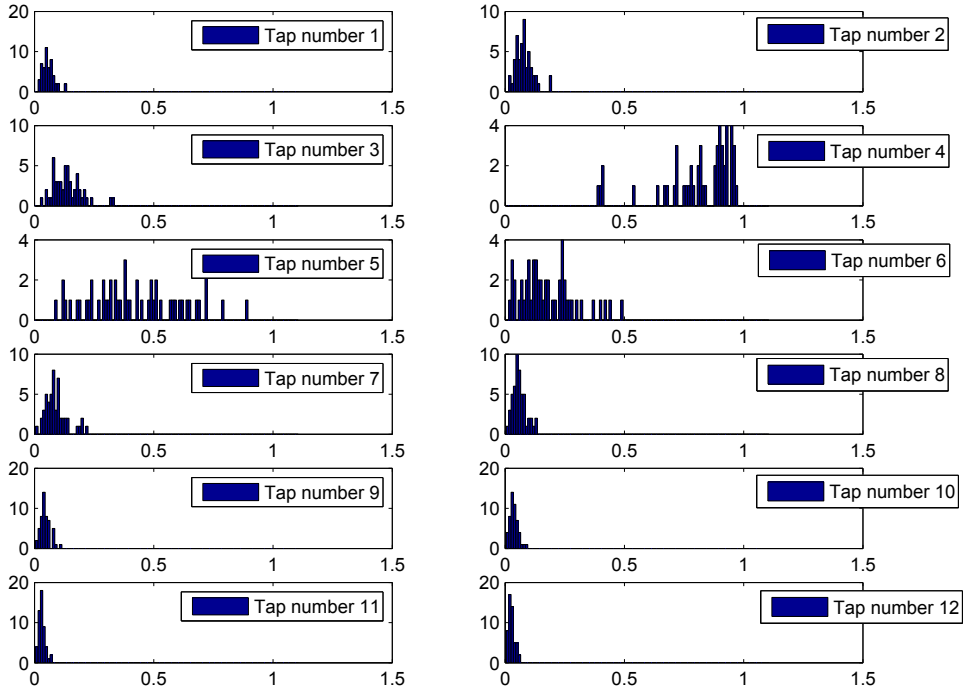


FIGURE A.1: Amplitude distribution of the channel coefficients of chapter 4.

A.2 Hardware Measurements

TABLE A.3: Hardware measurements with CC, interleaving and HD.

Measurement	Mean BER	Standard Dev	SIR	SINAD
1	0	0	2.2	6.3
2	$1.1e^{-3}$	0	2.5	5.1
3	0	0	2.4	5.9
4	$0.3e^{-3}$	$0.6e^{-3}$	2.5	6.1
5	$0.7e^{-3}$	$0.3e^{-3}$	2.5	6.3
6	$0.1e^{-3}$	0	1.7	5.2
7	0	0	7.3	6.1
8	0	0	7	5.6
9	0	0	7.5	6.7
10	0	0	7	5
11	0	0	7	7.5
12	$0.3e^{-3}$	0	3.7	5.7
13	$0.1e^{-3}$	$0.3e^{-3}$	5	5.5
14	$1.1e^{-3}$	$0.5e^{-3}$	−1	5.4
15	$6.6e^{-3}$	0	−1	4.4
16	$3.1e^{-3}$	0	−0.8	5.4
17	$0.3e^{-3}$	0	−0.5	5

Table A.4: Hardware measurements with OEC scheme.

Measurement	Mean Dis. Pack.	Standard Dev	SIR	SINAD
1	2.2	1.3	2.2	6.3
2	2.3	1.35	2.5	5.1
3	2.02	0.04	2.4	5.9
4	2	0	2.5	6.1
5	2	0.06	2.5	6.3
6	2.02	0.03	1.7	5.2
7	0.09	0.9	7.3	6.1
8	1.05	0.33	7	5.6
9	1.45	0.16	7.5	6.7
10	1.8	0.09	7	5
11	0.04	0.04	7	7.5
12	2.02	0.03	3.7	5.7
13	2.1	0.07	5	5.5
14	2.8	2.46	−1	5.4
15	7.3	2.6	−1	4.4
16	3.4	1.3	−0.8	5.4
17	2.1	1.3	−0.5	5

Appendix B

Creation of the narrowband interferer via Agilent E4438C ESG Vector Signal Generator

The procedure that was followed in order to generate the narrowband interference signal, is presented in the following steps. Before depicting these steps, it is necessary to specify the basic units of the Agilent E4438C ESG Vector Signal Generator, through which we can control the output signal. The goal is to allow a reader of this thesis understand how the narrowband signal can be generated. In the following subsection, each step of the process, refers to a specific option of one of the specified units. A detailed description of the specified signal generator can be found in [43]:

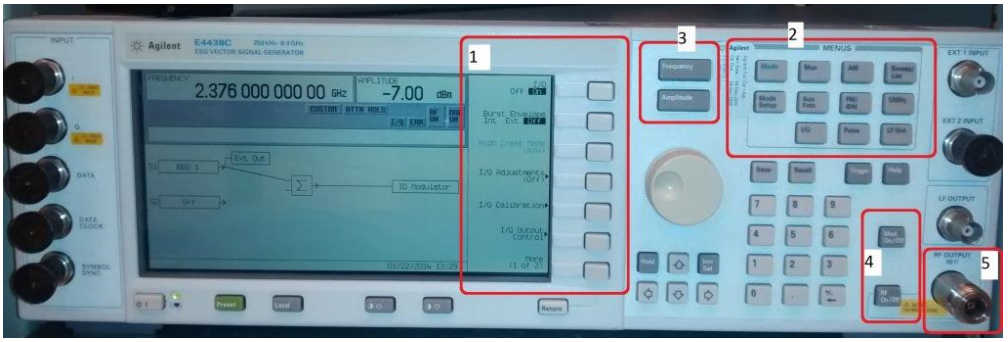


FIGURE B.1: Basic control parts of Agilent E4438C ESG Vector Signal Generator.

1. On the left side of unit 1 are depicted some options for the processing of the signal. The user can choose between the values of each option, by pressing the white buttons which are situated at the same level but on the right side of unit 1.

2. The menu of unit 2 gives the user the ability to choose between different types of baseband modulation.

3. Unit 3 allows the user to make changes in the amplitude and the frequency of the RF output signal.

4. The user can decide whether the output data will be modulated or not and whether it will be upconverted to RF with unit 4.

5. Finally, the output of E4438C ESG Vector Signal Generator is obtained through unit 5. If the output is chosen to be an RF signal, the connection should be made through a connector with 50 Ohm resistance.

B.0.1 Narrowband interference generation steps

1. Press Mode from unit 2. Then choose Real Time I/Q baseband mode through unit 1.

2. Choose Mode Setup from unit 2. From unit 1 make the following settings.

   - Set Custom mode to ON.
   - Set data sequence to PN23.
   - Select Symbol Rate to 1Msps.
   - Select Modulation Type to BPSK.
   - Select a Gaussian pulse shaping filter with Bbt=0.22.

3. Select I/Q modulation from unit 2. From unit 1 make the following settings.

   - Set I/Q to ON.
   - Set Burst Envelope to OFF.

4. Select Frequency from unit 3. Set Freq Ref to OFF. Set Frequency to 2.376 GHz.

5. Select Amplitude from unit 3. Set the appropriate amplitude value.

6. From unit 4, enable modulation by setting Mod to ON, and also enable the RF output by setting RF to ON.

7. Connect a 50 Ohm connector to unit 5.

Appendix C

Matlab functions

C.1 MATLAB Codes

C.1.1 QPSK Modulation

```
hMod = comm.QPSKModulator;
hMod.PhaseOffset = pi/4;
hMod.BitInput = true;
hMod.SymbolMapping = 'Gray';
modDataf = step(hMod, encodedData);
```

C.1.2 QPSK Demodulation

```
function [demodDataf] = demodulate_qpsk(input)

hDeMod = comm.QPSKDemodulator;
hDemod.PhaseOffset = pi/4;
hDeMod.BitOutput = true;
hDeMod.SymbolMapping = 'Gray';
demodDataf = step(hDeMod, input);
```

C.1.3 Interleaving

```
function [out,test] = interleavingFinal(input,n,m)
```

```
size_in = length(input);   % size of the input signal
size_inter = m*n;         % size of the interleaver

inter = zeros(n,m); % interleaver matrix with n rows and m columns
out_temp=zeros(size(input)); %

times = floor(size_in/size_inter); % number of interleaving operations

rr = reshape(input.',96,times).';

for k = 1:times
    for i = 1:n
        for jj = 1:m
            if ((jj ==1)&(i==1))
               inter(1,1)=rr(k,1);
            else
               inter(i,jj)=rr(k,n*(jj-1)+i);
            end
        end
    end

     if k==1
         out_temp(1,1:96) = reshape(inter.',1,96);
     else
         out_temp(1,(k-1)*96+1:k*96) = reshape(inter.',1,96) ;
     end
end
test = inter;
out = out_temp.';
```

C.1.4 Deinterleaving

```
function [out,test] = deinterleavingFinal(input,n,m)

size_in = length(input);   % size of the input signal
size_inter = m*n;         % size of the deinterleaver
times = floor(size_in/size_inter); % number of deinterleaving operations
out_temp= zeros(size(input)); %


rr = reshape(input.',96,times).';

for k = 1:K
```

```
    deinterm(1:n,1:m) = reshape(rr(k,:).',m,n).';
    deintermout(k,1:96) = reshape(deinterm,1,96);

     if k==1
         out_temp(1,1:96) = deintermout(1,:);
     else
         out_temp(1,(k-1)*96+1:k*96) = deintermout(k,:);
     end
 end
out = out_temp;
test = intermout;
```

C.1.5 Convolutional Encoder

```
function [encoded] = encoder(data)
% Code properties
codeRate = 1/2;
constlen = 7;
codegenpoly  = [171 133];
tblen    = 34;
trellis  = poly2trellis(constlen, codegenpoly);

% Create a rate 1/2, constraint length 7 ConvolutionalEncoder System object.
hConvEnc = comm.ConvolutionalEncoder(trellis);

% Convolutionally encode the data
encoded = step(hConvEnc, data);
```

C.1.6 Viterbi Decoder

```
function [decoded] = decoder(data)
% Code properties
codeRate = 1/2;
constlen = 7;
codegenpoly  = [171 133];
tblen    = 34;
trellis  = poly2trellis(constlen, codegenpoly);
% create decoder object
hVitDecHD  = comm.ViterbiDecoder(trellis, 'InputFormat', 'Hard',...
    'TracebackDepth', tblen);

% Decode the data
```

```
decoded = step(hVitDecHD, data);
```

C.1.7 Long Training Sequence Creation

```
function [trainout] = ofdmTrain(train)
% OFDM modulation
dc = zeros(1,1);
nFFT = 64;
nDSC = 52;
% Insert training sequence
tF = [train zeros(1,11)] ;
tF = [tF(:,27:64) tF(:,1:26)].';

% Taking FFT, the term (nFFT/sqrt(nDSC)) is for normalizing the power of transmit symbol to
traint = (nFFT/sqrt(nDSC))*ifft(tF).';
trainout = traint;
```

C.1.8 Short Training Sequence Creation

```
function [trainout] = ofdmTrainShort(symbols)
% symbols denotes the number of short training sequences that form a short preamble
nFFT = 64;
subcarrierIndex = [-26:-1 1:26];
ShortPreambleInput = [zeros(1,8) 1+j 0 0 0  -1-j 0 0 0 ... % [-32:-17]
 1+j 0 0 0  -1-j 0 0 0 -1-j 0 0 0   1+j 0 0 0 ...             % [-16:-1]
 0   0 0 0  -1-j 0 0 0 -1-j 0 0 0   1+j 0 0 0 ...             % [0:15]
 1+j 0 0 0   1+j 0 0 0  1+j 0 0 0   0   0 0 0 ];             % [16:31]

short_freq = sqrt(13/6)*fftshift(ShortPreambleInput);

% taking ifft
short_time = ifft(short_freq,nFFT); % generate 64 sample sequence

% create the short preamble
shortPreamble = [short_time short_time short_time(1:32)];
```

C.1.9 OFDM Frame Creation

```
function [frame,number] = ofdmFrame(input,long_train,data_symbols,short_train)

numberofFrames = data_symbols/20; % each frame has 20 OFDM symbols
tempout=[];

cp = long_train(1,33:64);

% frame process
  for i = 1:1:numberofFrames
      if i == 1
         tempout = [tempout short_train cp long_train long_train...
           input(1,1:1920-4*80)];

      else
         tempout = [tempout short_train cp long_train long_train...
            input(1,(i-1)*(20*80)+1:i*(20*80))];
      end
  end

frame = tempout;
number = numberofFrames;
```

C.1.10 OFDM Frame Extraction

```
function [rxdata,rxtraining] = ofdmdeFrame(input,numberofframes)

tempdata=[];
temptrain=[];
symperFrame = 24; % each frame has 24 OFDM symbols
% 20 for data and 4 for the preamble


  for i = 1:1:numberofframes
      if i == 1
         tempdata=[tempdata input(1,321:symperFrame*80)];
         temptrain=[temptrain input(1,1:320)];
      else

         tempdata=[tempdata input(1,(i-1)*(symperFrame*80)+321 :i*(symperFrame*80) )];
         temptrain=[temptrain ...
         input(1,(i-1)*(symperFrame*80)+1 : (i-1)*(symperFrame*80) + 320 )];

      end

  end
```

```
    rxdata = tempdata;
    rxtraining = temptrain;
```

C.1.11 OFDM Modulation

```
function [ofdm] = ofdmMod(input,symbols,bitsperSym)
% OFDM modulation
pilot = ones(symbols,4);
dc = zeros(symbols,1);
nFFT = 64;
nDSC = 52;

 %% OFDM Modulate
   ipMod = reshape(input,bitsperSym,symbols).'; % grouping into multiple symbols

   % Assigning modulated symbols to subcarriers
   xF = [ipMod(:,[1:5]) pilot(:,1) ipMod(:,[6:18]) pilot(:,2) ipMod(:,[19:24])...
    dc(:,1) ipMod(:,[25:30]) pilot(:,3) ipMod(:,[31:43]) pilot(:,4)...
     ipMod(:,[44:48]) zeros(symbols,11)] ;

   xF = [xF(:,27:64) xF(:,1:26)].';

% Taking FFT, the term (nFFT/sqrt(nDSC)) is for normalizing the power of transmit symbol to
    xt = (nFFT/sqrt(nDSC))*ifft(xF).';

% Appending cylic prefix
  xt = [xt(:,[49:64]) xt];


% Concatenating multiple symbols to form a long vector
ofdm = reshape(xt.',1,symbols*80);
```

C.1.12 OFDM Demodulation

```
function [ofdmDemod,test1] = ofdmDeMod(input_demod,train_demod,train_init,symbols,...
bitsperSymm,taps,hF1,numberofframes,noise_variance)

test1 = 0;

%%  OFDM Demodulate Parameters
nFFT = 64;
```

```
nDSC = 52;

% formatting the received vector into symbols
yt = reshape(input_demod.',80,symbols).';

% format the received preamble into OFDM symbols
traint = reshape(train_demod.',80,size(train_demod,2)/80).';
% extract long preamble
traina = reshape(train_demod.',160,size(train_demod,2)/160);
% obtain long training sequence
trainb = traina(33:160,:);
traint = reshape(trainb,64,numel(trainb)/64).';


yt = yt(:,[17:80]); % removing cyclic prefix
% converting to frequency domain
yF = (sqrt(nDSC)/nFFT)*fftshift(fft(yt.')).';

% converting to frequency domain
trainF = (sqrt(nDSC)/nFFT)*fftshift(fft(traint.')).';

yF =fftshift(yF);
trainF =fftshift(trainF);


% remove 11 zeros
yMod_53 = [yF(:,39:64) yF(:,1:27)];
train_53 = [trainF(:,39:64) trainF(:,1:27)];

yMod_53b = [yMod_53(:,1:26) yMod_53(:,28:end)];
train_53b = [train_53(:,1:26) train_53(:,28:end)];


%% ZF Equalize
train_theory  = [train_init(:,1:26) train_init(:,28:end)];


[equalized,testxx] = equalize_new(yMod_53b, train_53b,train_theory,numberofframes);

times = size(equalized,1);
% compute phase offset of every OFDM symbol
nr_samples = 1;
for i = 1:1:(times/nr_samples)
     if i==1
        input2 = equalized(1:nr_samples,:);
     else
        input2 = equalized((i-1)*nr_samples+1:i*nr_samples,:);
     end
```

```
     [outf tt1] =phaseComp(input2);
     % reshape the output again
     cfo(i) = outf;
% perform phase offset compensation
equalized((i-1)*nr_samples+1:i*nr_samples,:) = equalized((i-1)*nr_samples+1:i*nr_samples,:)
.*exp(j*outf);
end

pilot_out = cfo;

% extract data subcarriers
yMod_48 = [equalized(:,1:5)   equalized(:,7:19) equalized(:,21:32)...
          equalized(:,34:46)   equalized(:,48:end)];

temp = reshape(yMod_48.',1,(symbols)*48).';

ofdmDemod = temp;
```

C.1.13 Coarse and Fine Frequency Estimation and Correction

```
function [output,outputtrain, short1,long1] = FreqCorrection1 (input,inputtrain,offset)
 short1 = 0; % coarse frequency estimation
 short2 = 0; % for testing
 long1 = 0; % fine frequency estimation
 long2 = 0;  % for testing
 L32 =32;
 L64 = 64;
 L128 = 128;
 N = 64;
 outputtrain = 0; % frequency compensated preamble
 output = 0;      % frequency compensated data sequence

%% FIRST SP COARSE CORRECTION
% Obtain the parts of the short preamble which are going to be correlated
for ll = 1:1:10
    if ll == 1
       yys(1,1:16) = inputtrain(1,offset+1:offset+16*ll);
    else
       yys(ll,1:16) = inputtrain(1,offset+1+(ll-1+0)*16:offset+16*(ll+0));
    end
end
% perform the correlation
for i =1:1:size(yys,1)-1
   for k  = 1:1:16
       mul(i,k) = yys(i,k).*conj(yys(i+1,k));
```

```
   end
   summation(i)=sum(mul(i,:));
end

argum = angle(summation);
fdelta_short = -argum*N/(pi*L32);
fdelta_av_short = sum(fdelta_short)/numel(fdelta_short);
short1 = fdelta_av_short;

% perform coarse frequency correction to the input sequence and to the training sequence
input = input.*exp(1i*2*pi*(-fdelta_av_short)*(0:length(input)-1)./64);
inputtrain = inputtrain.*exp(1i*2*pi*(-fdelta_av_short)*(0:length(inputtrain)-1)./64);


%% LP FINE CORRECTION
% Obtain the parts of the long preamble that are going to be correlated
for jj = 1:1:2
    if jj == 1
       yyl(1,1:64) = inputtrain(1,160+offset+32+1:160+offset+32+64*jj);
    else
       yyl(jj,1:64) = inputtrain(1,160+offset+32+1+(jj-1)*64:32+160+offset+64*jj);
    end
end
% perform the correlation
for i =1:1:size(yyl,1)-1
   for k  = 1:1:64
       mull(i,k) = yyl(i,k).*conj(yyl(i+1,k));
   end
   summationl(i)=sum(mull(i,:));
end
arguml = angle(summationl);
fdeltaLong = -arguml*N/(pi*L128);
fdelta_av_long = sum(fdeltaLong,1)/size(fdeltaLong,1);
long1 = fdelta_av_long;
% perform fine frequency compensation according to the fine estimation
input = input.*exp(1i*2*pi*(-fdelta_av_long)*(0:length(input)-1)./64);
inputtrain = inputtrain.*exp(1i*2*pi*(-fdelta_av_long)*(0:length(inputtrain)-1)./64);
outputtrain = inputtrain;
total = short1+short2+long1+long2;
long2 = total;
output = input;
```

C.1.14 Phase Offset Compensation

```
function [tuned,test] = phaseComp(input,pknown)
```

```
p1r = input(1:size(input,1),6);
p2r = input(1:size(input,1),20);
p3r = input(1:size(input,1),33);
p4r = input(1:size(input,1),47);


% known pilot sequence/ normally the size of p1, p2, p3, p4 is 1 but
%the possibility for a bigger size was also investigated for testing purposes
p1 = ones(numel(p1r),1)*pknown(1);
p2 = ones(numel(p1r),1)*pknown(2);
p3 = ones(numel(p1r),1)*pknown(3);
p4 = ones(numel(p1r),1)*pknown(4);

% compute phases of the known pilots
phi1 = angle(p1);
phi2 = angle(p2);
phi3 = angle(p3);
phi4 = angle(p4);


% compute phase difference of the received pilots
phi1r = angle(p1.*conj(p1r))/1;
phi2r = angle(p2.*conj(p2r))/1;
phi3r = angle(p3.*conj(p3r))/1;
phi4r = angle(p4.*conj(p4r))/1;


% compute average phase
for jj=1:1:size(phi1,1)
    av1(jj,1)= (phi1r(jj,1)+phi2r(jj,1)+phi3r(jj,1)+phi4r(jj,1))/4;
end
av_tot = 1*sum(av1,1)/size(av1,1);


tuned = av_tot; % estimated phase offset
test = av1;
```

C.1.15 Zero Forcing Equalization

```
function [equalized,test1] = equalize_new(input,train_rec,train_init,nrframes)
% train_rec : received training sequence
% train_init : transmitted training sequence

k = 0;
```

```
size_train1 = size(train_rec,1);
size_train2  = size(train_rec,2);
for ll = 3:4:size(train_rec,1)
    k = k+1;
    train_av(k,1:52) = train_rec(ll,:);
    k=k+1;
    train_av(k,1:52) = train_rec(ll+1,:);
end


for i = 1:1:size(train_av,1)/1
    ratio(i,1:52) = train_av(i,:)./train_init(1,:);
end

ratio = ratio.';       %transpose the ratio
rconj = conj(ratio);
ramp = (abs(ratio).^2);

kk = 0;
for i = 1:2:size(train_av,1)
    kk = kk+1;
    mean_conj(1:52,kk) =rconj(:,i) + rconj(:,i+1)/2;
    mean_abs(1:52,kk) = ramp(:,i)+ ramp(:,i+1)/2;
    ratiob(1:52,kk) = mean_conj(1:52,kk)./mean_abs(1:52,kk);
    gaint(kk,1:52) = ratiob(1:52,kk).';
end

for i = 1:1:size(train_av,1)/2
    for j = 1:1:20
        m = 20*(i-1)+j;
%each row of gaint is used for the equalization of 20 OFDM symbols
         equalize_out(m,1:52) = gaint(i,:).*input(m,:);
     end
end
equalized = equalize_out;
test1 =   gaint;
```

C.1.16 Format Data for the Server Software

```
clc
clear all

load('outputConv7035.mat')
%create I inputs
samples = size(framed,2);
```

```
i = zeros(1,samples);
q = zeros(1,samples);

imax = zeros(1,samples);
qmax = zeros(1,samples);

binary_i = char(samples,13);
binary_q = char(samples,13);

binary_i_temp = char(samples,16);
binary_q_temp = char(samples,16);

iout = zeros(1,samples);
qout = zeros(1,samples);


i_bin = zeros(samples,16);
q_bin = zeros(samples,16);

% assign to i,q channels
i = real(framed(1,1:samples));
q = imag(framed(1,1:samples));

it = i;
qt = q;

 for ii = 1:1:numel(i)
     if i(ii)>=2
         i(ii)=2;
     end
     if i(ii)<=-2
         i(ii)=-2;
     end
     if q(ii)>=2
         q(ii)=2;
     end
     if q(ii)<=-2
         q(ii)=-2;
     end
 end

%nr of bits
bits = 10;
bits16 = 16;
%mult with maximum range for a 16 bit representation

i = i* (2^(bits-2)  );
```

```
q = q* (2^(bits-2) );

imax=  i;
qmax = q;

imax = round(imax);
qmax = round(qmax);

for ii = 1:1:numel(i)
     if imax(ii)== (2^10) /2
         imax(ii)=imax(ii)-1;
     end

     if qmax(ii)== (2^10) /2
         qmax(ii)=qmax(ii)-1;
     end

end

i = round(imax);
q = round(qmax);

extend_i(1,1:3) = dec2bin(0,3);
extend_q(1,1:3) = dec2bin(1,3);

for j=1:1:samples
    binary_i(j,1:13) = dec2twos(i(j),13);
    binary_q(j,1:13) = dec2twos(q(j),13);

    binary_i_temp(j,1:16) = [ binary_i(j,:) extend_i(1,:)];
    binary_q_temp(j,1:16) = [ binary_q(j,:) extend_q(1,:)];
end

for  j =1:1:samples
    for i=1:1:bits16
        i_bin(j,i) = str2num(binary_i_temp(j,i));
        q_bin(j,i) = str2num(binary_q_temp(j,i));
    end
end


for k = 1:1:samples
    iout(k) = bi2de(i_bin(k,1:16),'left-msb');
    qout(k) = bi2de(q_bin(k,1:16),'left-msb');
end
```

C.1.17 Recover Data from the Server Software

```
fid  = fopen('ofdmcoded.rstrm', 'r');
s = dir('C:\Project_Ilias\RF_Meas_TEST\UNIFIED_OFDM_QPSK_GEN\ofdmcoded.rstrm');

%% Allocate memory
data_size = 200e3; % extract 200e3 data samples

bit_size = 12;
idat =  zeros(1, data_size);
qdat =  zeros(1, data_size);
irec =  zeros(1, data_size);
qrec =  zeros(1, data_size);

datain = zeros(data_size,2);

iin = zeros(data_size,16);
qin = zeros(data_size,16);

i_in2 = zeros(data_size,bit_size);
q_in2 = zeros(data_size,bit_size);

i_in = char(data_size,bit_size);
q_in = char(data_size,bit_size);
% specify data size
dataint = fread(fid,[2 s.bytes/10],  'uint16');
dataint = dataint.';
fclose(fid);

datain = dataint;
% convert to binary
for k =1:1: size(datain,1)
     iin(k,:) = d2b(datain(k,1));
     qin(k,:) = d2b(datain(k,2));
end


bits = 12;
bits16 =16;
samples = size(datain,1);
DataSamples = size(iin,1);
% from the 16 bit representation only the 12 MSBs are useful
i_in2 =  iin(:,1:bits);
q_in2 =  qin(:,1:bits);
% change format
```

```
for  j =1:1:DataSamples
    for i=1:1:bits
      i_in(j,i) = sprintf('%i',i_in2(j,i));
        q_in(j,i) = sprintf('%i',q_in2(j,i));

    end
end

bits_data = 12;
% convert from complement 2 format to binary
for k = 1:1:DataSamples

     i_rec(k) = twos2dec(i_in(k,:));
     idat(k) = i_rec(k)/(2^(bits_data-2) );

     q_rec(k) = twos2dec(q_in(k,:));
     qdat(k) = q_rec(k)/(2^(bits_data-2) );

end

end
```

C.1.18 Compute BER of a burst after excluding the outliers

```
clc
clear all

load('ber_snr6.mat')
burst_nr = size(ber_conv,2);
check_matrix = zeros(13,burst_nr);
X = zeros(13,burst_nr);
outliers_idx = zeros(13,burst_nr);

for i = 1:1:13
%% each line of the ber_conv corresponds to a different SINR scenario
% for measurement x. each measurement consists of approximately 7 bursts.
% The average BER of these bursts for each SINR value is used.
% Before obtaining the average value, the outliers of each scenario have to
% be found. The  outliers are calculated with the Thompson Tau method.
%The remaining BER values are used for the average calculation of each
%BER for each SINR scenario.

% The execution of this scenario 18 times, since 18 measurements
% were performed, gives 18 sets of measurements. .
```

```
[X_temp, outliers_idx_temp] = outliers([ber_conv(i,:)], 1);

if isempty(outliers_idx_temp)
   outliers_idx(i,:) =  outliers_idx(i,:);
   X(i,:) = X_temp;
else
    outliers_idx(i,1:numel(outliers_idx_temp)) =  outliers_idx_temp;
    X(i,1:numel(X_temp))  =  X_temp;
end

removed_ind= 0;
average_t = [];

for k = 1:1:burst_nr
   if isnan( X(i,k))
      X(i,k) = 10;
      removed_ind= removed_ind+1;
       average_t = average_t;
   else
       average_t = [average_t X(i,k)];
   end
end
ber_average(i) = mean(average_t);
end
```

C.1.19 Create the narrowband jammer

```
function [output_jammer] = jammer(samples_nr,ratio, jam_freq)

% the ratio of the jammer bandwidth over the total bandwidth
ratioint = ratio;

%use a big number of samples if we want convergenve ot the variation
samples = samples_nr;

% create white gaussian noise
bl_base = [randn(samples,1)+1i*randn(samples,1)];

%Single stage single rate
lowrate = jam_freq/2;
Hlp = fdesign.lowpass('N,Fc',200,lowrate*1e6,20e6);    % Lowpass prototype
N = length(Hlp.Numerator)-1;

%specify frequency shift
```

```
Fc = jam_freq;                                  % Desired frequency shift
j = complex(0,1);
Hbp = copy(Hlp);

% perform frequency shift
Hbp.Numerator = Hbp.Numerator.*exp(j*Fc*pi*(0:N));

% filter the noise and produce the jammer signal
output_jammer = filter(Hbp,bl_base);
```

C.1.20 Creation of the Fountain Generator Matrix

```
function G = GenerateG(K, N, ColTH, TryNum)
%function G = GenerateG(K, N, ColTH, TryNum)
%
%Decription:
% This function generates the binary matrix for Fountain codes
%
%Input:
% K - number of rows
% N - number of columns
% ColTH - lower bound for number of 1's per col (multiple of K/S)
% TryNum - number of tests to choose from
%
%Output:
% G - a K X N binary matrix
%
%Formed by: Roee Diamant, UBC, July 2012

if nargin < 3
    ColTH = 1.3; %bound for number of 1's in each column
end
if nargin < 4
    TryNum = 1;
end

TryG = {};
MeanSummerRow = zeros(1, TryNum);
MinSummerRow = zeros(1, TryNum);
for TryInd = 1: TryNum
    while(1)
        %-----------------
        %create pdf
        c = 0.03;
        delta = 0.3;
```

```
    S = c*log(K/delta)*sqrt(K);

    rho = zeros(1, K);
    thu = zeros(1, K);
    d = 1: K;

    rho(1) = 1/K;
    rho(2:K) = 1 ./ (d(2:end) .* (d(2:end)-1));
    size(rho,1)
    size(rho,2)
    loc = floor(K/S-1);
    thu(1: loc) = S/K ./ d(1: loc);
    thu(loc+1) = S/K*log(S/delta);
    size(thu,1)
    size(thu,2)
    Z = sum(rho + thu);
    mu = (rho + thu) / Z;
    %-----------------
    %threshold for maximum number of ones per column
    MaxColTH = K/S * ColTH;

    G = zeros(K, N);
    for n = 1: N
        while(1)
            %-----------------
            %draw a number
            Cmu = cumsum(mu)/sum(mu);
            u = rand(1,1);
            diff = abs(u - Cmu);
            [M, loc] = min(diff);
            Num = d(loc);
            %-----------------
            if Num <= MaxColTH
                break;
            end
        end
        row = randperm(K);
        G(row(1: Num), n) = 1;
    end

    SummerRow = sum(G, 2);
    if min(SummerRow) > 0
        MeanSummerRow(TryInd) = mean(SummerRow);
        MinSummerRow(TryInd) = min(SummerRow);
        TryG{TryInd} = sparse(G); %#ok<AGROW>
        break;
    end
end
```

```
end

%maximize weight per row and minimize weight per col
[V, loc] = max(MeanSummerRow);
G = TryG{loc};
G = full(G);
```

C.1.21 Fountain Encoder

```
function Code = EncodeFountain(G, msg, Base)
%function Code = EncodeFountain(G, msg, Base)
%
%Description:
%This function encodes fountain code
%
%Input:
% G - a K X N binary matrix
% msg - vector of K symbols to decode
% Base - base of codeword
%
%Output:
% Code - vector of N coded symbols
%
%Formed by: Roee Diamant, UBC, July 2012

Code = mod(G' * msg, Base);
```

```
function Decoded = DecodeFountainLT(G, Code, Base)
%function Decoded = DecodeFountainLT(G, Code, Base)
%
%Decription:
% This function decodes Fountain LT code
%
%Input:
% G - binary matrix (K X N)
% Code - code word to decode (vector size of N)
% Base - base of codeword
%
%Output:
% Decoded - vector of K decoded symbols (-1 indicates unseccesfull decoding)
%
%Formed by: Roee Diamant, UBC, July 2012
```

```
K = size(G, 1);
N = size(G, 2);

GTag = G;
Decoded = -ones(K, 1);
Summer = zeros(1, K);
while(1)
    loc = find(sum(GTag,1) == 1);
    if isempty(loc)
        %code fails
        break;
    end
    for ind = 1: length(loc)
        pos = find(GTag(:, loc(ind)) == 1);
        loc2 = find(GTag(pos, :) == 1);
        GTag(pos, :) = 0;
        Summer(pos) = 1;
        Decoded(pos) = Code(loc(ind));
        Code(loc2) = mod(Code(loc2)-Decoded(pos), 2^(Base));
    end
    if sum(Summer) == K
        break;
    end
end
```

C.1.22 OEC Encoding Scheme

```
function [channelOut,freqResp] = oec_coding(G)

load ('parity_matrix_175_255.mat');
LTBase = 2;
Hs = sparse(H2);

% create objects
hEnc = comm.LDPCEncoder('ParityCheckMatrix',Hs) ;
hMod = comm.PSKModulator(4, 'BitInput',true);
hModQPSK = comm.QPSKModulator('BitInput',true,'PhaseOffset',pi/4);
hDec = comm.LDPCDecoder('ParityCheckMatrix',Hs) ;
hError = comm.ErrorRate;
crcPoly =[1 0 0 0 1 0 0 1];
hGen = comm.CRCGenerator(crcPoly);
hDetect = comm.CRCDetector(crcPoly);


K = 500;
```

```
N = 732;
nrbits = 168;
% Create input packets
operation = [zeros(1,0.5*K*nrbits) ones(1,0.5*K*nrbits)];  %# Fill the vector with 0 and 1
bits = operation(randperm(numel( operation)));  %# Randomly reorder it
packetstemp = bits.';
packets = reshape(packetstemp,K,numel(packetstemp)/K);

%%  Encode
encoded = mod(G'*packets,2);

% Fountain encoding
for i =1:1:size(encoded,1)
% take every line of the 732x168 matrix
  data=  encoded(i,:)';
% LDPC + CRC encoding of each Fountain coded packet
  [encodedData(:,i),ldpcin(:,i)] = ldpc_crc_enc(data,hEnc,hGen);
 % Modulation
  encodedData256(:,i) = [encodedData(:,i) ; 1];
  modSignal(i,:)      = step(hModQPSK, encodedData256(:,i));
end

%  Reshape
ratio =  floor(size(encoded2,1)/48);
for j = 1:1:ratio
   if j == 1
      vect(1:48*128,j) = reshape(modSignal(1:48,:),48*128,1);
   else
      vect(1:48*128,j) = reshape(modSignal((j-1)*48 +1:j*48,:),48*128,1);
   end
end

remaining = size(encoded2,1)-48*ratio;
rr = modSignal(ratio*48 +1:end,:);
vectremain = reshape(modSignal(ratio*48 +1:end,:),remaining*128,1);

vectorIntemp = reshape(vect,numel(vect),1);
vectorIn = [vectorIntemp ;vectremain];

% add some extra bits in order to achieve the specified frame size
ofdm_in_s = [vectorIn; 0.7*(ones(576,1)+1i*ones(576,1))];%% avoid clipping
```

C.2 Copyright Notice Roee Diamant