Master’s Thesis

Analysis and Design of an OFDMA Wideband Repeater Architecture

Georg Piewald

Analysis and Design of an OFDMA Wideband Repeater Architecture

Master's Thesis

at

Graz University of Technology
Technical University of Denmark

submitted by

Georg Piewald

Signal Processing and
Speech Communication Laboratory

Graz University of Technology
A-8010 Graz, Austria

Department of
Photonics Engineering

Technical University of Denmark
DK-2800 Kgs. Lyngby, Denmark

4th June 2010

Advisor:	Assoc.-Prof. Dr. Klaus Witrisal
Co-Advisor:	Assoc.-Prof. Dr. Henrik Wessing
Co-Advisor:	MSc. Christian Lanzani

Abstract

For wireless network providers, repeaters are a cheap and attractive way of filling coverage gaps and extending range. The goal of this thesis is to evaluate the possibilities and challenges of implementing a repeater in a modern OFDMA-based communication network, such as WiMAX or LTE. The work is divided into two parts. The first part entails a case study, which gives a broad overview over various repeater and relay schemes.

In the second part one particular design was chosen and studied in detail. It was decided to focus on an on-channel repeater, which forwards an incoming signal at the same frequency, at which it was received. One of the major challenges of such a device is the unwanted feedback from the output to the input antenna. To counteract this effect, an adaptive cancellation system is considered and elaborated, both analytically and through simulations.

The work also involved a thorough study of the Mobile WiMAX physical layer and an implementation thereof in Matlab. The technical background of WiMAX and a description of the implementation is given in the appendix. Furthermore, the Matlab implementation of the repeater and the cancellation system is described there.

Pledge of Integrity

I hereby certify that the work presented in this thesis is my own, that all work performed by others is appropriately declared and cited, and that no sources other than those listed were used.

Place:	____________________
Date:	____________________
Signature:	____________________

Contents

List of Figures

List of Tables

Acronyms

3GPP	3rd Generation Partnership Project
4G	4th Generation (mobile communication network)
AES	Advanced Encryption Standard
AMC	Adaptive Modulation and Coding
ARQ	Automatic Repeat Request
BER	Bit Error Rate
BPSK	Binary Phase Shift Keying
BS	Base Station
CP	Cyclic Prefix
DAB	Digital Audio Broadcast
DL	Downlink
DVB-T	Terrestrial Digital Video Broadcast
EAP	Extensible Authentication Protocol
FCH	Frame Control Header
FEC	Forward Error Correction
FFT	Fast Fourier Transform
FDD	Frequency-Division Duplex
FIC	Feedback Interference Cancellation
FIR	Finite Impulse Response
FUSC	Full Usage of Subcarriers
GSM	Global System for Mobile Communications
H-FDD	Half Duplex Frequency-Division Duplex
HUMAN	High-Speed Unlicensed Metropolitan Area Network
IEEE	Institute of Electrical and Electronics Engineers
IF	Intermediate Frequency
IFFT	Inverse Fast Fourier Transform
IP	Internet Protocol
ISI	Inter-Symbol Interference
LFSR	Linear Feedback Shift Register
LMS	Least Mean Squares
LNA	Low Noise Amplifier
LOS	Line-Of-Sight
LTE	Long Term Evolution
MAC	Medium Access Layer
MAN	Metropolitan Area Network

MIMO	Multiple Input Multiple Output
MS	Mobile Station
MSE	Mean Squared Error
MSD	Mean Squared Deviation
NLOS	Non-Line-Of-Sight
OCR	On-Channel Repeater
OFDM	Orthogonal Frequency Division Multiplexing
OFDMA	Orthogonal Frequency Division Multiple Access
OSI	Open Systems Interconnection Model
PHY	Physical Layer
PUSC	Partial Usage of Subcarriers
QAM	Quadrature Amplitude Modulation
QPSK	Quadrature Phase Shift Keying
REP	Residual Error Power
RF	Radio Frequency
RRH	Remote Radio Head
RS	Relay Station
RTG	Receive Transition Gap
SAW	Surface Acoustic Wave
SFN	Single-Frequency Network
SNR	Signal-to-Noise Ratio
SS	Subscriber Station
TDD	Time-Division Duplex
T-DMB	Terrestrial Digital Multimedia Broadcasting
TDMA	Time-Division Multiple Access
TTG	Transmit Transition Gap
TUSC	Tile Usage of Subcarriers
UL	Uplink
WiMAX	Worldwide Interoperability for Microwave Access
WAN	Wide Area Network
WMAN	Wireless Metropolitan Area Network
WWAN	Wireless Wide Area Network

Chapter 1

Introduction

The progress in wireless technologies in recent decades has brought forth a multitude of communication systems and applications which support high data rates even in demanding environments with strongly varying channel conditions. One of the most noteworthy developments in this field is orthogonal frequency division multiplexing (OFDM). Due to its dense packing of data carriers and its inherent robustness against dispersive multipath channels, it is usually the modulation technique of choice, whenever high data rate transmission is required.

Broadcasting networks like digital audio broadcast (DAB) and digital video broadcast (DVB) have successfully employed OFDM since the mid-1990s. In multi-user mobile communication networks on the other hand, the establishment of OFDM took somewhat longer. However, recent developments show a clear shift of such networks towards OFDM, particularly with the emergence of orthogonal frequency division multiple access (OFDMA). Major candidates for fourth generation (4G) cellular networks, such as Mobile WiMAX (IEEE 802.16e-2005) and 3GPP Long Term Evolution (LTE) are based on OFDMA.

The above mentioned network types can be categorized as wireless wide (or metropolitan) area networks (WWAN or WMAN), which implies that they require region-wide infrastructures of transmission sites to provide coverage in the designated area. Typically a transmission site is thought of as a fully equipped base station (BS), consisting of one or several antennas mounted on a mast or rooftop and a nearby located enclosure which accommodates all analog and digital signal processing. More precisely, the enclosure contains all functionalities of the physical (PHY) and the medium access (MAC) layer of the system as well as RF components and backhaul connectors to the core network. This "traditional" type of transmission site suffers from several drawbacks like high installation- and maintenance costs, a large site footprint and high power loss between enclosure and antenna. For network operators the installation of a fully equipped BS is often too costly to fill minor coverage gaps or shaded areas. Attractive alternatives are *distributed base stations* and *repeaters*.

A distributed base station in the context of cellular communication networks is based on the idea of detaching the BS enclosure from the transmission site [Lanzani et al., 2010] [Leavey, 2006]. For this purpose a so-called remote radio head (RRH) is mounted close to the antenna on the mast or rooftop, which only performs the last steps of the transceiver chain, such as analog/digital conversion, frequency translation and analog filtering and amplification. The BS enclosure is connected to the RRH via optical fibres and can thus serve a multitude of transmission sites at distances up to 20 km. In digital broadcasting networks a similar concept exists for low power transmission sites, which cover smaller areas or fill coverage gaps [Channelot WP].

The other alternative to extend wireless coverage at low cost is to employ repeaters. A repeater (or relay – the differences will be explained in detail in Chapter 2) is particularly beneficial in situations where a wired backhaul link is difficult or impossible to install. The only wired connection it requires is a power supply, which makes it relatively independent from the operational environment. Possible scenarios for using a repeater are [Sydir et al., 2006]:

- **Fixed infrastructure extension**, e. g. to supply coverage holes or improve the signal strength at the edges of the coverage area.
- **Temporal coverage**, e. g. to quickly establish a communication infrastructure in emergency cases or to improve capacity during special events.
- **In-building coverage**, e. g. inside buildings or tunnels.
- **Mobile vehicle coverage**, e. g. to provide a stable signal on-board a train or a ferry.

At first glance, designing a repeater seems trivial – taking two transceiver chains and cross-connecting input and output connectors should yield an applicable device. However, in practice a multitude of realization possibilities with numerous of options exist, each with advantages and drawbacks for different use cases and network types. The implementation and deployment of repeater schemes in WWANs requires in-depth knowledge and careful consideration to ensure successful operation.

This project arose from the idea of modifying the hardware of an existing remote radio head for WiMAX base stations and turn it into a repeater. The aspired type of repeater as well as performance and usage objectives were initially entirely unspecified and should be determined in the course of the work. Therefore a major part of this thesis is bound to investigate the various alternatives and analyze their applicability. The result of this part is a high level case-study of repeater schemes, which is presented in Chapter 2. It attempts to give an exhaustive overview of possible methods of information forwarding in wireless networks, evaluating advantages and disadvantages of each. Though the investigation is tried to be kept as broad and general as possible, a certain emphasis on OFDM and in particular WiMAX systems is existent.

The conclusion of this case study left two alternatives on which the project could be continued. First, a *Layer 1 amplify-and-forward repeater* could be studied, with focus on signal enhancement through digital signal processing in the forwarding path. Second, a *Layer 2 multihop-relay* could be investigated. As will be explained in Section 2.2, Layer 2 forwarding needs to be explicitly defined by the wireless system definition. Though the original WiMAX standard does not support multihop relaying, an amendment labeled IEEE 802.16j has been under development since 2006 for this purpose. However, at the time of deciding the further direction of this project, these specifications were not available[1]. For this reason it was agreed to discontinue the work on Layer 2 relaying, even though this is a promising and innovative technology. A brief overview over the IEEE 802.16j standard is given in Section 2.4.

Instead, subsequent research focused on *on-channel repeaters* (OCR), which are a special type of Layer 1 repeaters. The concept is analyzed in detail in Chapter 3, where specific chal-

[1]In fact, IEEE 802.16j was formally approved in June 2009, which was before the start of this thesis. Nevertheless it became freely available to public only in late December of the same year.

lenges in this scheme are explained and quantified through simulations. This discussion forms the basis for ideas to improve the repeater performance. In particular an adaptive feedback cancellation system is investigated and presented in Chapter 4.

The thesis entailed a comprehensive study of the Mobile WiMAX physical layer. An overview over the specification is given in Appendix A. In order to perform accurate simulations, a detailed implementation of the WiMAX transceiver chain was created in Matlab. This implementation is explained in detail in Appendix B, together with a listing of the source codes. Finally, Appendix C explains the structure and design considerations for the implementation of the OCR. Both the WiMAX transceiver and the OCR were implemented as object oriented Matlab classes. The source code listings only cover the code belonging to the classes. The numerous simulation scripts developed during this work are not included in this print.

Chapter 2

Case Study of Repeater and Relay Architectures

The case study presented in this chapter attempts to summarize the various possibilities of data relaying schemes. The following sections present common ways to classify these schemes in order to give a broad overview of this extensive topic. First, a very general classification in Section 2.1 briefly explains the most frequently encountered terms and their origins coming from wired networks. Sections 2.2 and 2.3 classify relaying schemes by their operational level and by the hardware complexity respectively. The focus is on metropolitan area wireless networks and in particular cellular networks, such as IEEE 802.16 (WiMAX). Nevertheless, single frequency networks like digital audio and digital video broadcast (DAB and DVB) are also mentioned at times. In Section 2.4 an amendment to the WiMAX specification is presented, as an example for protocol supported multihop relaying.

2.1. General Classification of Information Forwarding Schemes

In the history of communication systems and networks a multitude of technologies has been invented to forward data from a source to a sink by means of active devices on the path. The objectives are to improve signal strength and quality for the receiver, to translate the representation of information from one communication link to another or to interconnect different network segments. Depending on factors such as the complexity of the device or its purpose, a number of terms are in use to classify these schemes. However, it is hard to find precise definitions of terms and often they are used contradictory in the literature.

One of the most common ways to classify forwarding schemes is by the *open systems interconnection model* (OSI), depending on which OSI layer the forwarding device operates. Table 2.1 summarizes typical terms for such devices. In the OSI world, people usually talk about hubs, bridges and routers and distinguish them by how data-forwarding is done. On the physical layer, a *hub* blindly forwards incoming data to all connected devices or network segments. A *bridge*, operating on data link layer is aware of site-specific addresses and can thus decide on which ports to forward incoming packets. A *router* refers to a device which operates on the network layer and is thus capable of resolving global IP addresses into site-specific addresses, to make forwarding decisions.

These terms are usually used in conjunction with computer networks. Two other commonly encountered terms, repeater and relay, are used more generally. A *repeater* is widely understood as a device that only amplifies and possibly improves (reshapes) an incoming signal without understanding the contained information. Probably the most ambiguous (or general) term is *relay*. It is often used when information has to be forwarded using the same communication technology and an alike transmission medium, but more complexity involved than with a repeater. In systems with a multiple-access medium this usually requires involvement of the media access (MAC) layer, which justifies the commonly found term *Layer 2 relay*. However, from a more traditional point of view, relay stations mainly refer to communication links via directional radio. This says nothing about the operational layer and may well be a pure signal amplification.

Layer	**OSI model**		**More general**	
	Device	forwards...	Device	recognizes...
Layer 1	Hub	to all ports	Repeater	waveforms
Layer 2	Bridge, Switch	based on MAC address	Relay	system specific information (e.g. MAC packets)
Layer 3	Router, L3-Switch	based on IP address	Router	inter-network information (e.g. IP packets)

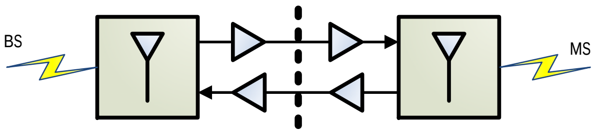
Table 2.1.: Common classification of forwarding schemes

2.2. Forwarding in Wireless Networks: OSI-oriented Classification

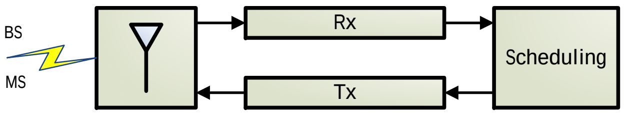
Forwarding received signals and data in a wireless network is significantly more challenging than in wired networks. The reason is that the transmission medium at the incoming port (i.e. antenna) is the same as at the outgoing port and special care must be taken that incoming and outgoing signals don't interfere and eliminate each other. A classification of forwarding schemes frequently found in the literature is directly inherited from wired networks and based on the operational layer (according to the OSI model) of the device, i.e. Layer 1 repeater, Layer 2 relay and Layer 3 router.

However, the boundaries between the layers are somewhat fuzzy in wireless networks. For example, a device depicted as Layer 1 repeater might still require some MAC layer functionality to extract timing and scheduling information contained in the signal. Therefore a better means of distinguishing the categories is by the number of antennas and transceiver chains required for the device. Implicitly, this defines how the problem of the common medium is dealt with. A Layer 1 repeater separates the medium either physically or by shifting the forwarded signal to another frequency band. In any case, this requires two antennas and some sort of filtering and amplification in between them. A Layer 2 relay has access to the MAC layer and can thus use specially reserved multiple access slots to forward data without interference. Since most modern wireless communication systems involve some sort of time-division multiple access (TDMA), receiving and retransmitting can be scheduled at different times. In this case, only one antenna and transceiver chain is necessary. A Layer 3 router acts like an individual base station (BS) operating on another channel (i.e. frequency band) than the donor BS. Hence it requires two complete transceiver chains and all functionality contained in a regular base station.

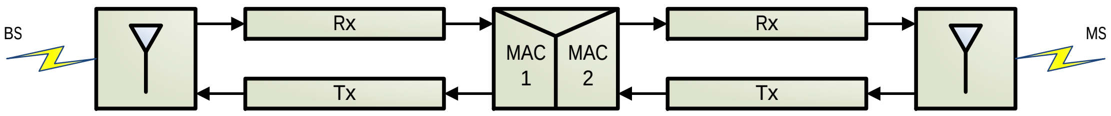
The three schemes are schematically depicted in Figure 2.1. The following subsections explain them in more detail.



(a) Layer 1 repeater



(b) Layer 2 relay



(c) Layer 3 router

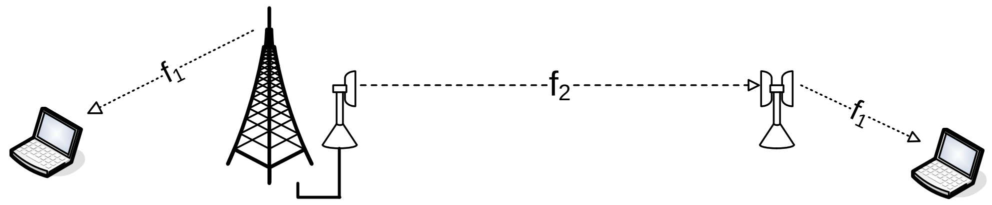
Figure 2.1.: Classification of forwarding schemes in wireless networks

2.2.1. Layer 1 Repeater

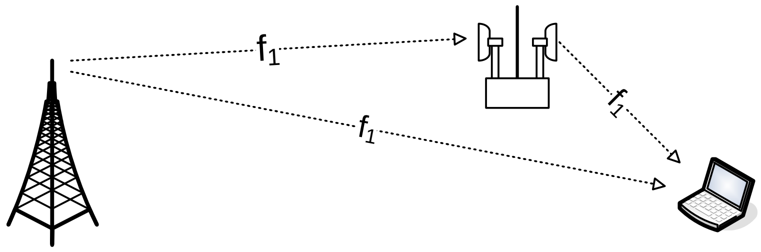
A Layer 1 repeater amplifies and retransmits incoming signals instantaneously and without any modification of the contained data. This means that reception and retransmission are happening simultaneously and two antennas (or antenna systems) are required. The complexity of the device can vary greatly. In the simplest case it would work entirely in RF domain and only consist of a low-noise amplifier, an analog bandpass filter (often a SAW filter) and a high power amplifier. Alternatively it could be equipped with down- and up-conversion units to shift the signal to an intermediate frequency (IF) which allows analog bandpass filtering with superior selectivity. The next stage of complexity would be to convert the IF signal to digital domain. An equalizer could then be used to remove some of the received noise. Furthermore, the signal can be shifted to baseband, demodulated and even decoded before being retransmitted again. In practice however, many of the latter steps cannot be realized, as they would introduce a too long processing delay. This problem will be further explained in Chapter 3.

If the repeater performs a down-conversion to an IF band, it is also possible to up-convert the signal later to a different radio frequency. Such a device is called *frequency translating repeater*. The major advantage is that the received signal and the retransmitted signal do not interfere. Hence there is no stringent requirement on the isolation between the antennas and signal paths. However, deploying a frequency translating repeater is not always possible. Obviously, in a single frequency network, frequency shifting is not an option. But also in cellular networks such repeaters cannot be used directly, by just forwarding the signal on an alternative channel. In this case, problems arise on the MAC layer. A mobile station (MS) which is in reach of both transmission sites (the base station and the repeater) would see the same information on two different channels. This causes a conflict for the MS when it tries to subscribe to a BS (both at network entry and during handover) as it receives the same BS identity as well as acknowledge messages on different channels.

Despite these limitations, frequency translating repeaters can still be used and are being deployed in many network infrastructures [Axell RCB] [RFI-CE900]. This is done by using two repeaters, as sketched in Figure 2.2(a). One repeater is located directly at the BS and shifts the signal to a frequency other than that of the cell. This is called the *link frequency* and used to transmit the signal (usually via directional antennas) to a second repeater. This second repeater then shifts the signal back to the original (coverage) frequency, from where it is broadcast using e.g. an omni-directional antenna. As an example, a GSM operator who has licences in both 900 MHz and 1800 MHz bands, could provide coverage in rural areas on 900 MHz and interconnect repeaters on 1800 MHz.



(a) Frequency translating scheme, employing two repeaters



(b) On-channel repeater

Figure 2.2.: Two types of Layer 1 repeater schemes

A repeater that does not perform frequency translation is called *on-frequency repeater* or – more common though less precise – *on-channel repeater* (OCR). Two challenges are to be solved in this architecture. First, if the output signal, which is often as much as 80 dB stronger than the input signal, is fed back to the input antenna, the repeater falls into oscillatory behaviour. This has to be avoided with a proper isolation between the antennas, as depicted by the thick line in Figure 2.2(b). The second issue is the overlapping of the direct signal from the sender to the receiver and the signal from the repeater to the receiver. Since forwarding takes a certain amount of time (usually a few µs), the receiver will see a delayed version of the original signal which acts like an additional multi-path component and may cause inter-symbol interference (ISI). However, systems which are designed for frequency selective channels (e.g. OFDM) can use both signals constructively, as long as the delay doesn't exceed a certain limit. Therefore, a short processing delay of the repeater is of vital importance. On-channel repeaters are a focus of this thesis and will be discussed in detail in Chapter 3.

2.2.2. Layer 2 Relay

A Layer 2 relay has to demodulate and decode at least part of the signal, in order to get access to MAC layer information and thus to the multiple access scheme. To avoid both problems of a Layer 1 on-channel repeater (feedback interference and interference between direct and indirect signals), a relay can schedule reception and retransmission in separate time-slots. Therefore a single transceiver chain and antenna is sufficient and no antenna isolation is necessary.

A major problem of Layer 2 forwarding is that the MAC protocol has to be aware of relay stations. In other words, relaying has to be explicitly included in the standard definition of the wireless system. One of the reasons for this is that the BS, which is usually in charge of scheduling has to know about relay stations (RS) within its coverage area and allocate special slots for communication with them. Another reason is that both BS and RS have to agree on which MSs each serves and how packet routing is done. Also network entry and initial ranging are issues that have to be addressed.

WiMAX, as most other current wireless systems, does not have included support for Layer 2 forwarding. However, an extension to the standard called IEEE 802.16j was approved in June 2009 and describes the necessary modification in the PHY and MAC layer to allow multihop relaying. An overview over the amendments is given in section 2.4.

2.2.3. Layer 3 Router: Self-Backhauling

Probably the most obvious way of increasing coverage by relaying is to add an additional base station with a wireless link back to another base station which has access to the core network. A link between BS and core network is often called *backhaul link*. Since this forwarding scheme uses the wireless system itself as a backhaul connection, wireless layer 3 relaying is frequently referred to as *self-backhauling*.

This is the most complex (and thus expensive) method of forwarding, as it requires the entire functionality of a BS. It implies that two complete transceiver chains and antenna systems are required. The advantage is that in an all-IP core network no changes to the system specification are necessary. Signals don't interfere since they are transmitted on different frequency bands and packet routing can be done with native IP routing mechanisms. In such a scheme, the router of course has to be considered as an individual cell of the cellular network. This has to be taken into account during network enrolment and frequency planning and is thus only an option for network operators, not for end users.

2.3. Hardware-oriented Classification of Wireless Forwarding

Depending on the purpose of the forwarding scheme, an alternative classification may be appropriate, which reflects the complexity of the hardware of the device (i.e. the number of stages involved in the transceiver chain). Three categories can be distinguished:

- Amplify and forward (A&F)
- Modulate and forward (M&F)
- Decode and forward (D&F)

Figure 2.3 shows the building blocks of an OFDM transceiver. The dashed lines indicate which blocks are involved in each of the forwarding schemes.

Note that the nomenclature only refers to the data-path of the device. The control path, which for example has to decide at what times to switch between reception and transmission mode, may have to decode information from the incoming signal in any of the above repeaters. Thus, even though this is a hardware level categorization, it cannot be seen as a sub-division of a Layer 1 (physical layer) repeater.

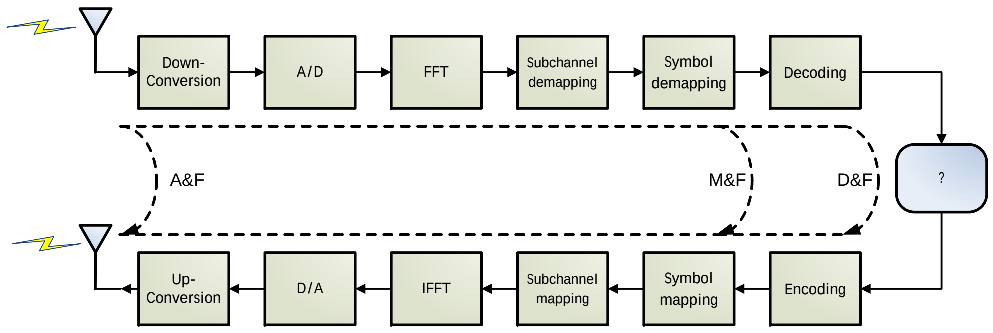


Figure 2.3.: Possible forwarding schemes of an OFDM transceiver

2.3.1. Amplify & Forward

The most simple version is an amplify and forward (A&F) repeater. As it does not involve any demodulation, it can in principle be a purely analog device, comprised of a low-noise amplifier, a power amplifier and several filter stages. Nevertheless, a conversion to digital domain may also be performed, allowing advanced features such as equalization or interference cancellation. This requires additional hardware for frequency translation (to baseband or an intermediate frequency) and A/D conversion. Furthermore, if the system is operated in time-division duplex mode (uplink and downlink are separated in time) a synchronization module is necessary to switch between transmission and reception mode of the two antennas. The disadvantage of an A&F repeater is that it does not only amplify and forward the useful information but also the received noise.

2.3.2. Modulate & Forward

In addition to analog amplification, a modulate and forward (M&F) repeater performs all steps required for symbol detection and re-mapping. This can be understood as a signal enhancement, since the received noise is removed and not forwarded. However, if M&F shall be used for an on-channel repeater, the signal processing involved with demodulation normally takes too much time to allow immediate forwarding (several OFDM symbol durations in WiMAX), and the delayed forwarded signal would interfere destructively with the possibly existent original signal. Therefore some Layer 2 understanding is required to schedule the retransmission at a special reserved multiple access slot. In WiMAX the slot allocation is done by the BS and advertised to the subscribers in so-called MAP messages in the beginning of each frame. Therefore, even though the data which has to be forwarded is not decoded in an M&F repeater, the control path still needs to decode each frame header and extract necessary data. This is also necessary in order to find out the used symbol constellation, which may be different for each user and may be changed from frame to frame.

A special feature of a M&F repeater is the possibility to change the symbol constellation before forwarding. In this way both links (BS-RS and RS-MS) could be operated with modulation schemes optimal to their respective channel. An example for an M&F scheme operating on Layer 2 is the IEEE 802.16j option for *direct relay zones*, which will be described in Subsection 2.4.3.

2.3.3. Decode & Forward

A decode and forward (D&F) repeater is comprised of the entire transceiver chain, including de- and encoder. This further enhances the M&F scheme by allowing error correction along the forwarding path. However, decoding comes at the cost of considerable longer processing time. While M&F in WiMAX takes a few symbol durations, D&F may take several frame durations and significantly increases the delay.

After decoding, the raw MAC data are available to the RS and can be modified before for forwarding. This may be necessary in some Layer 2 relaying schemes for routing purposes. A Layer 3 router needs decoding in any case, to modify frame headers and MAC information.

2.4. Layer 2 Relaying in WiMAX: IEEE 802.16j

In mid 2005 the IEEE 802.16 *Mobile Multihop Relay Study Group* was formed with the task to study prospects and impacts of relaying in WiMAX networks. A project authorization request titled "IEEE 802.16j - Multihop Relay Specification" was granted in March 2006 and after nine drafts, the standard was approved in June 2009.

802.16j is an amendment to Mobile WiMAX (IEEE 802.16e-2005) to allow Layer 2 forwarding. The main requirement for the development was that modifications of the standard should only affect BSs and RSs, while MSs which are compatible to Mobile WiMAX shall be usable in the relay enhanced network without changes.

The task group identified four usage scenarios for relaying [Sydir et al., 2006], which are summarized in Table 2.2. Each scenario aspires certain performance objectives, which can be

Usage Model	**Example**	**Main Objective**	**Relay Mobility**
Fixed Infrastructure	cell edge coverage hole outside cell area	coverage capacity range	fixed
In-building Coverage	inside buildings inside tunnels under ground	coverage capacity	fixed nomadic
Temporary Coverage	emergency recovery special events	capacity range	nomadic
Coverage on a Mobile Vehicle	inside buses inside ferries inside trains	coverage capacity	mobile

Table 2.2.: Use cases of IEEE 802.16j [Sydir et al., 2006]

coverage, capacity, range or any combination of these. The capacity enhancement can be understood twofold. Since a relay reduces the distance between sender and receiver, the received signal strength naturally is increased. This allows the use of a higher modulation order and coding rate, thus increasing the overall throughput. The additional throughput can be used either to increase the data rate of individual users or to allow a higher number of users entering the network. Both aspects are contained in the term *capacity*.

Note that these usage models do not define the complexity of the relay. For any scenario, relays ranging from very simple to highly complex are possible, including Layer 1 A&F repeater and Layer 3 self-backhauling router (which both are not in the scope of 802.16j). However, depending on whether the desired enhancement is capacity or coverage/range, 802.16j offers two significantly different ways of forwarding, which also differ in complexity. These are referred to as *transparent* and *non-transparent* relaying and will be briefly described in the following two subsections. A more complete overview over IEEE 802.16j can be found in [Peters and Heath, 2009] and [Genc et al., 2008].

2.4.1. Transparent Relaying

Figure 2.4 shows an ordinary frame of the WiMAX PHY layer in time-division duplex (TDD) mode. It consists of a downlink subframe and an uplink subframe, separated by a transition gap of one symbol duration (which allows the transceivers to switch between receiving and transmitting mode). A frame always starts with a preamble, which is used for synchronization purposes. Following the preamble, the frame control header (FCH) gives general information concerning the configuration of the frame. The allocation of time/frequency bursts to different users is specified in downlink and uplink MAP messages. For more details about the frame construction confer Appendix A.

In 802.16j both the uplink and the downlink subframe are split into access and relay zones. During the *access zone* the RS is in receiving mode (receives either from BS or MS) and during

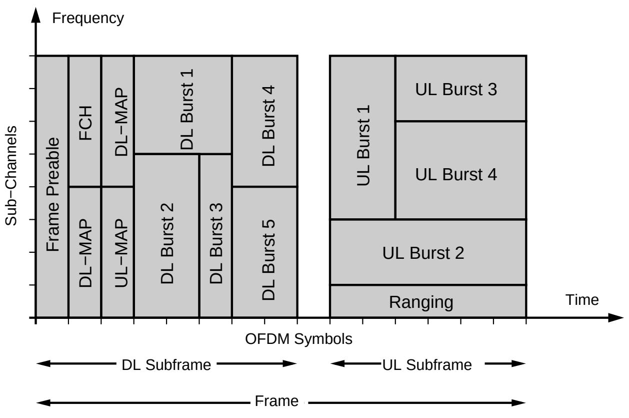


Figure 2.4.: Standard WiMAX TDD frame

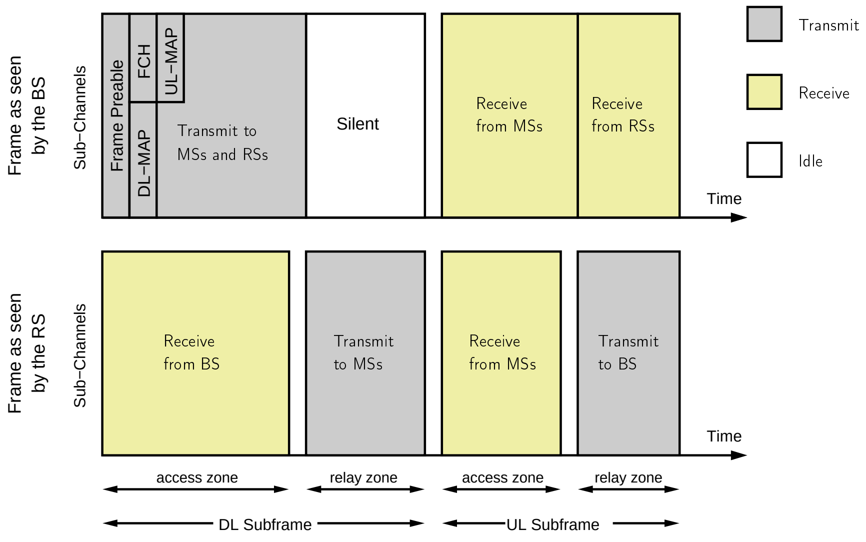


Figure 2.5.: 802.16j frame structure in transparent mode

the *relay zone* it is in transmitting mode. The modified frame structure of 802.16j in transparent mode is depicted in Figure 2.5. It shows twice the same frame, above how it is seen by the base station and below as seen by the relay station. During downlink access zone, the BS transmits preamble, headers and bursts to all MSs and RSs it directly serves. The RS only receives in this time interval. In downlink relay zone (also called *transparent zone*) the BS remains quiet while the RS transmits data to subordinate MSs. In this way BS and RS do not interfere with each other.

In the uplink access zone, both BS and RS receive from their MSs in different slots. The relay zone consists of slots which are allocated to RSs for transmitting data to the BS.

A very important implication of this frame configuration is that the RS is in receive mode in the beginning of each frame and thus forwards neither frame preamble nor any header or MAP message. This implies that MSs associated to a RS must also be in the coverage area of the BS to receive the headers. The transparent mode is thus not capable of extending the range or coverage of a BS. Instead, transparent relaying can be used to increase the capacity by providing high throughput links (high modulation order and coding rate) to areas where the BS signal is weak, such as the cell edge.

The name *transparent relaying* arises from the fact that the MS does not notice the existence of a RS. The allocation of data slots is performed only by the base station (*centralized scheduling*) which allows a low-cost implementation of the RS. A drawback is that since the MS synchronizes to the BS, it also performs the channel estimation on the BS-MS link, while the actual data is transmitted on the RS-MS link. Hence, synchronization and equalization algorithms employed by the receiver might give sub-optimal results.

2.4.2. Non-Transparent Relaying

In non-transparent mode a RS also forwards frame header information such as preamble, FCH and MAP messages. It can thus be used to extend the cell size or to to fill coverage gaps and shadowed areas. The frame structure is shown in Figure 2.6. In the downlink access zone both BS and RS simultaneously transmit to their subordinate MSs. In the following relay zone, the BS sends data bursts only to RSs. This zone starts with a midamble, containing a relay-specific frame control header (R-FCH) and a MAP message with the slot allocation for BS-RS communication. The uplink subframe is split in the same way as in transparent mode: an access zone in which BS and RSs receive from their MSs and a relay zone in which the BS forward data to the BS.

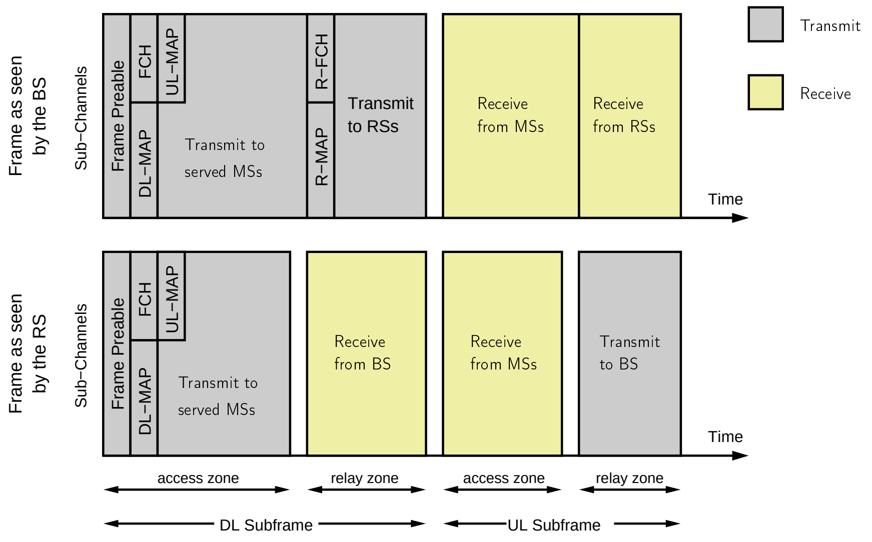


Figure 2.6.: 802.16j frame structure in non-transparent mode

Differently from transparent relaying, the non-transparent mode allows forwarding over multiple hops. Furthermore, it is also allowed for RSs to take scheduling decisions and allocate slots to users. This is referred to as *distributed scheduling* and is beneficial in larger topologies with more than two hops. In any case, a MS perceives a RS as if it was an ordinary BS, which gives rise to the term *non-transparent relaying*.

A disadvantage of non-transparent relaying is the increased interference between RS cells, since all RSs are transmitting simultaneously. Hence, the use of non-transparent relays to increase network capacity is limited.

2.4.3. M&F Relaying in 802.16j

The standard also defines an option to allow a modulate and forward scheme, i.e. data bursts are demodulated but not decoded. Since demodulation can be performed in less than one OFDM symbol duration, the burst can be forwarded within the same frame, in which it was sent from the BS. Such bursts have to be placed in a special zone, called *direct relay zone*. Figure 2.7 shows a possible frame configuration containing direct relay zones. The bursts received in an access zone, will be forwarded in the immediately following relay zone.

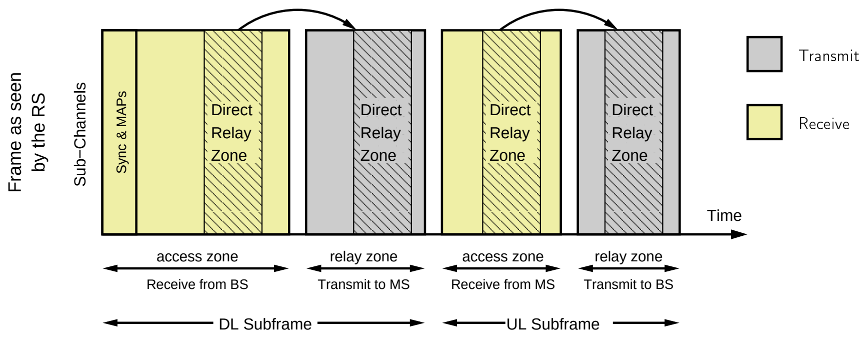


Figure 2.7.: 802.16j frame structure for M&F relaying using direct relay zones

This scheme can only be used in transparent mode and is beneficial when low RS complexity or low relaying latency is envisaged. The RS may also change the modulation order before re-modulating a burst and hence optimize both links to their respective channel. The decision to do so lies, of course, by the BS, since only centralized scheduling is possible. A draft of this option was published in [Chae et al., 2007].

2.4.4. MAC Layer Modifications

Besides the amendments of the frame structure on the PHY layer, 802.16j also requires a series of adaptations on the MAC layer. These cover the topics of MAC-addressing, data encapsulation, routing, path management, initial ranging, RS network entry, HARQ (hybrid automatic repeat request), RS grouping as well as link layer encryption and authentication. Each of them offers several options and realization possibilities depending on the relaying and scheduling mode. However, it is not yet clear which of all these will be included in a WiMAX system profile. Summaries of the MAC layer modifications can be found in [Peters and Heath, 2009] and [Genc et al., 2008].

Chapter 3

On-Channel Repeaters

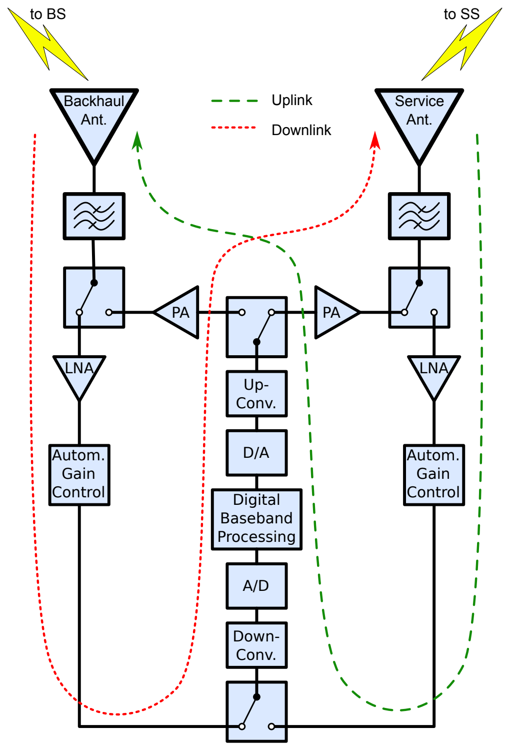
The concept of an on-channel repeater (OCR) was already introduced in Section 2.2.1. It describes a device operating on Layer 1, which forwards the received signal on the same frequency band. The complexity can range from purely analog amplifiers in RF domain to sophisticated digital systems, operating in baseband and employing equalizers, symbol decision blocks or even decoders.

Section 3.1 gives an overview over the concept and technology of OCRs. Block diagrams of possible implementations as well as references to existing repeaters are given. In Sections 3.2 and 3.3 the two major challenges of OCR-enhanced networks are treated in detail. Limitations of such schemes are discussed and conditions for efficient operation are presented.

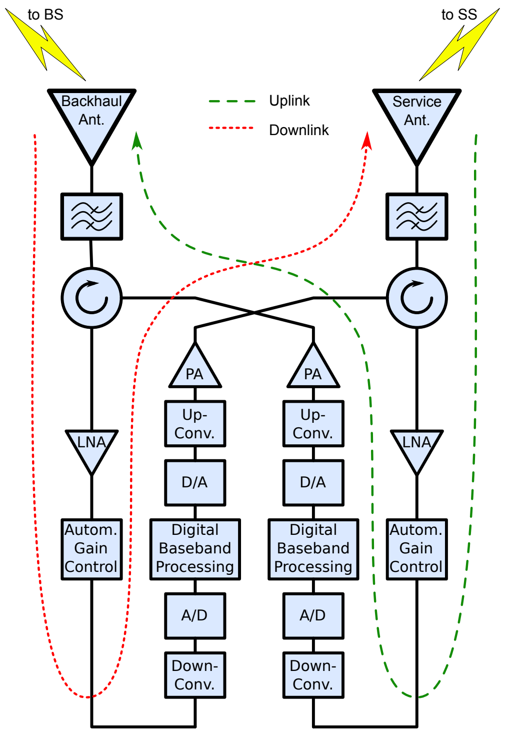
3.1. Overview

On-channel repeaters are not novel devices, they have been in use for many years in various types of networks. In OFDM systems however, new challenges but also new possibilities to improve the performance of an OCR arise. In recent years much literature on OFDM OCRs has been published, introducing baseband techniques like adaptive interference cancellation and equalization. One of the earliest and most frequently cited papers in this field was written by Wiewiorka and Moss [2005] at BBC R&D and describes an OCR for digital audio broadcast. A large part of more recent work comes from South Korea [e.g. Park et al., 2009] and focuses on OCRs for terrestrial digital multimedia broadcasting (T-DMB).

While there is a lot of research going on concerning OCRs in broadcasting networks, little can be found about repeaters in OFDM-based communication networks like IEEE 802.16 or LTE. Part of the reason is that the requirements of an OCR are more stringent in such networks. Owing to shorter OFDM guard intervals, the propagation delay of signals through the repeater has to be very short, for reasons explained in Section 3.3. Furthermore, in a communication network the OCR is a bi-directional device and therefore requires more complex hardware. If the network is run in frequency-division duplexing (FDD) mode, uplink and downlink occur simultaneously, which compels a duplication of the entire forwarding path. In time-division duplexing (TDD) systems, the OCR has to switch the signal flow between uplink and downlink direction, requiring accurate timing synchronization.



(a) Time-division duplexing



(b) Frequency-division duplexing

Figure 3.1.: OCR architectures for bi-directional wireless networks.

Figure 3.1 shows possible architectures of a frequency-division duplexing and a time-division duplexing OCR. In both cases the antenna directed towards the base station (BS) is called *backhaul antenna* and the one providing coverage for mobile stations (MS) is called *service antenna*. Incoming signals on either antenna first pass a bandpass filter which acts as a pre-selector and are then amplified by a low noise amplifier (LNA). Typically an OCR accepts a range of input power levels amplifying them to one fixed output power level. This requires an automatic gain control after the LNA, normalizing the incoming signal strength.

In FDD mode a circulator at each antenna takes care of the routing of incoming and outgoing signal. In TDD mode instead four switches are used to change the direction of the signal flow. The TDD mode of WiMAX allows time for these switches to change their state during transition gaps between uplink and downlink part of a frame (cf. Appendix A). The baseband processing block, together with frequency conversion and analog/digital conversion is used for both signal directions. However, if adaptive filtering for signal enhancement is employed, a distinct set of filter coefficients for each flow direction should be used, since the wireless channels may vary significantly between uplink and downlink.

A conversion to baseband is not always necessary. Depending on the type of signal enhancement to be employed, some algorithms might also work on an intermediate frequency. Only if the signal has to be demodulated (e. g. to extract pilot symbols to train an equalizer), it needs to be shifted to baseband.

For the analysis of signal processing algorithms in the forwarding path it is sufficient to treat the OCR as a uni-directional device which is done throughout this work. The antennas are then renamed to *input* and *output* antenna, referring to backhaul and service antenna in downlink direction, or vice versa in uplink direction.

As briefly mentioned in Section 2.2.1, an OCR scheme suffers from two types of interference which significantly limit its use in practice. In the following, these shall be denoted as *multihop interference* and *feedback interference* and are depicted in Figure 3.2. The subsequent two sections treat these types in detail and explain conditions that have to be fulfilled to circumvent interference.

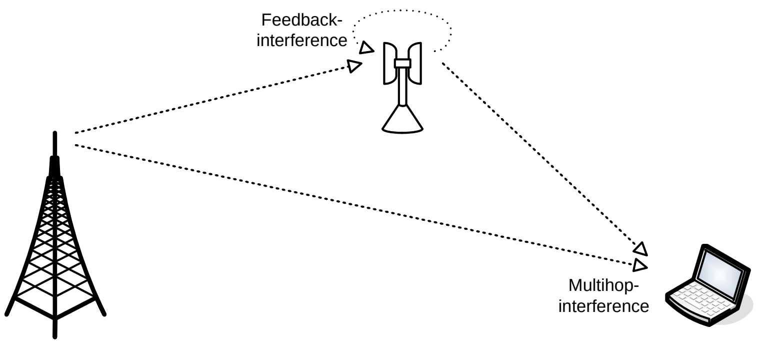


Figure 3.2.: Interference occurring in an on-channel repeater scheme.

Examples of four existing and commercially deployed WiMAX or LTE repeaters are shown in Table 3.1, together with specifications of the delay and maximum amplification. None of these employs digital signal enhancement techniques.

Repeater	Delay	Max. Gain	Reference
Renaissance, 8MTCA5L	$5\ \mu s$	$38.5\ dB$	[Renaissance Rep.]
now-tel, NWiMAX-100S	$2\ \mu s$	$60\ dB$	[NOWTEL 100S]
now-tel, NWiMAX-1000S	$3\ \mu s$	$80\ dB$	[NOWTEL 1000S]
TTT WiMAX Repeater	$< 50\ ns$	$80\ dB$	[TTT WiMAX Rep.]

Table 3.1.: Gain and delay of commercially built WiMAX/LTE repeaters

3.2. Feedback Interference

Feedback interference occurs when a part of the retransmitted signal reaches the input antenna and is amplified again. Since the output signal is many dB stronger than the arriving main transmitted signal, it can entirely eliminate the main signal, unless proper measures for isolation between the antennas are taken. Furthermore the feedback signal enters the repeater and is amplified again. This causes an oscillatory behaviour of the OCR and may eventually lead to instability. On the other hand, if the signal on the feedback path is attenuated stronger than the total forward gain of the repeater, its effect will die away.

This intuitive stability criterion will be formally derived in Subsection 4.2.2. In practice, to be sure of a stable operation even in varying channel conditions, it is commonly suggested [e.g. Slingsby and McGeehan, 1995] that the antenna isolation should be stronger than the forward amplification by a margin of at least $15\ dB$. Confer Table 3.1 for gain values of existing WiMAX repeaters. Sufficient attenuation can be achieved in several ways: physical isolation between the antennas (e. g. walls), beam forming (antenna patterns with strong attenuation in feedback direction), wave polarization and actively through interference cancellation. The latter option is the focus of Chapter 4.

For many simulations the feedback channel is modelled simply as a static single-tap FIR filter ([Park et al., 2009], [Suzuki et al., 1999]), with the tap having a delay between a few ns and several μs. An actual measurement of an OCR feedback channel was done by Slingsby and McGeehan [1995]. The experimental setup consisted of two shrouded yagi antennas, located at a distance of $5\ m$ and shielded by two brick walls. The result can be simplified to three taps as shown in Figure 3.3. The first and strongest received tap was attenuated by $87\ dB$ compared to the power level of the retransmitted signal. This means that the repeater gain must be less than $72\ dB$ to guarantee a stable operation. A very similar model for the feedback channel (in terms of tap delay and relative amplitudes) was used by Lee et al. [2009]. However, this model assumes the last tap (at $5\ \mu s$) to be non-stationary and change with a Doppler frequency of up to $10\ Hz$.

For completeness it shall be noted that feedback interference is sometimes also called *loop-interference* [Riihonen et al., 2009] or *self-interference* [Larsson and Prytz, 2008] by some authors. However, particularly the latter term is ambiguous, as it is also used for inter-symbol interference, inter-carrier interference and other types.

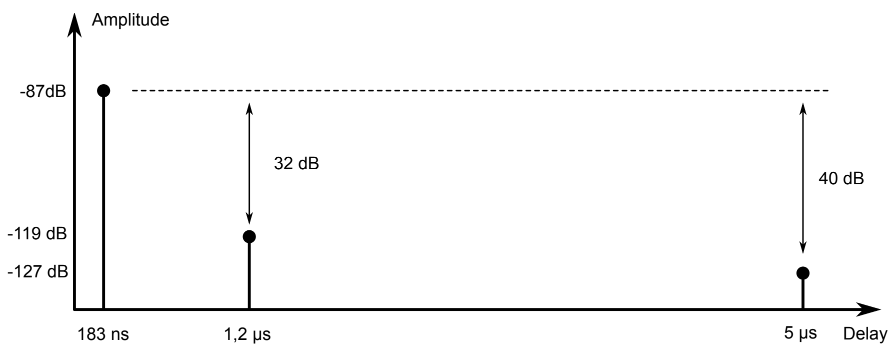


Figure 3.3.: Results of feedback channel measurements by Slingsby and McGeehan [1995] (simplified).

3.3. Multihop Interference

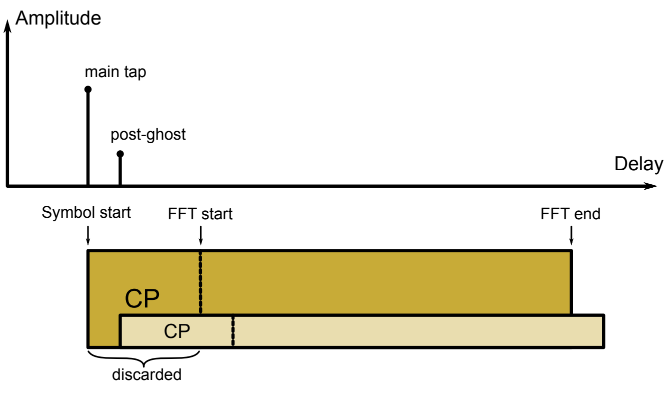
Typically an OCR would be installed in places where the power of the main transmitted signal is low, that is at the edge of the coverage zone or in shadowed areas. Even though the main signal is weak, it is often still clearly detectable by a receiver. A precise separation between the coverage area of the BS and of the repeater is usually not possible[1]. As a result the MS receives the same signal twice, first the weak main signal, then with some delay the stronger repeated signal (assuming that the MS is closer to the repeater than to the BS). Such a scenario can be treated as a special case of multipath environment.

In a conventional multipath environment, the channel impulse response typically has most power in the first received tap, which corresponds to the path with shortest length and least number of reflections. The following taps are more attenuated and are sometimes referred to as *post-ghosts*. OFDM systems can cope well with such environments, as long as the delay between the main tap and the post-ghost does not exceed the length of the OFDM symbol guard time (cyclic prefix length). Figure 3.4(a) shows a channel impulse response consisting of a main tap and one post-ghost. Below is a sketch of an OFDM symbol which is received twice, matching the two taps in delay and attenuation. If the receiver now synchronizes its timing such that the start of the symbol coincides with the main tap, it discards the first samples which belong to the CP. These samples also contain a part of the previous symbol which was received through the delayed path. The Fast Fourier Transform (FFT) however is only taken over samples belonging to the same OFDM symbol, thereby maintaining orthogonality between the subcarriers.

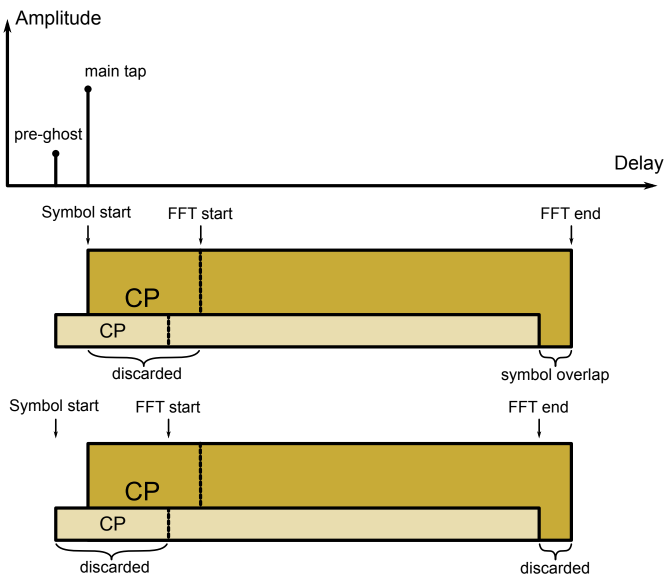
In a repeater-enhanced network, the channel impulse response (as perceived by the MS) looks different. The first tap, coming directly from the BS, is usually weaker than the main tap coming from the OCR and therefore called *pre-ghost*. This situation is depicted in Figure 3.4(b). Note that in reality the signals from both sources (BS and OCR) arrive through a multipath environment and therefore each of the taps would have a "tail" of post-ghosts. For simplicity these are omitted in the following discussion.

The figure shows two different ways how a receiver can synchronize its timing. If it sets

[1]An exception to this could be an indoor repeater, where the backhaul antenna is mounted on the outside and the service antenna on the inside of a building. If the walls are thick enough, the main transmitted signal will be attenuated below the detection level of the receiver.



(a) Channel with one post-ghost.



(b) Channel with one pre-ghost.

Figure 3.4.: Timing synchronization for the reception of an OFDM symbol after a multipath channel.

Parameter	**Specification**
Channel bandwidth	7 MHz
FFT size	1024
CP length	128 samples (22.8 μs)
Sampling period	125 ns
Subcarrier Modulation	16-QAM
Power of main tap	0 dB
Power of ghost tap	-5 dB

Table 3.2.: Simulation parameters for analyzing the influence of a ghost-tap.

the start of the OFDM symbol to the time of the strongest tap (like in Figure 3.4(a)), the FFT window will also contain part of the following OFDM symbol, which arrives ahead of time due to the pre-ghost. At the symbol boundary, non-steady shifts in amplitude and phase of each subcarrier can occur. If such a boundary falls within the FFT window, the periodicity of the window and the orthogonality between subcarriers is destroyed, leading to errors in the output of the FFT.

On the other hand, the receiver might set the symbol start time before the main tap, as shown in the bottom of Figure 3.4(b). If the start time is as early as the arrival of the pre-ghost or even earlier, the FFT window again only contains samples of the desired symbol and subcarrier orthogonality is maintained. The drawback in this case is that the ability to cope with post-ghosts decreases, since the guard time between the main tap and the start of the FFT window is reduced.

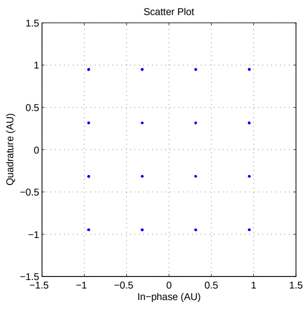
In practice finding the right symbol timing is a non-trivial task. A low complexity receiver may simply look for a correlation peak between the input signal and a known sequence such as a frame preamble. Sophisticated receivers usually employ a two-stage algorithm, first coarse correlation-based time-domain synchronization, then frequency-domain fine tuning using demodulated pilot symbols. The choice of algorithm and the decision of where to place the FFT window is entirely up to the receiver implementor and not defined in the system specifications. Hence it is difficult to state a general rule on how much time-skew between the main tap and a pre-ghost should be permitted.

The effects of pre- and post-ghosts on the signal quality can be seen in Figure 3.5, which shows scatter plots of subcarrier symbols at the receiver, after performing FFT and gain/phase synchronization[2]. Each scatter plot was calculated for noiseless reception with a ghost tap leading or lagging the main tap by a different amount of samples. Symbol timing was always synchronized to the main tap, which implies that the entire CP was used only for post-ghost cancellation. Nevertheless, the simulations are also meaningful for receivers that set the symbol start earlier than the main tap. In this case the lead time can be interpreted as the number of samples by which the pre-ghost is outside (before) the start of the CP.

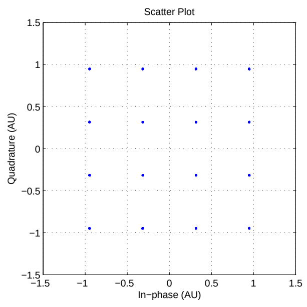
The detailed parameters for the simulation are listed in Table 3.2. The given sampling period results from the WiMAX specifications with a chosen signal bandwidth of 7 MHz.

The good performance of OFDM in the presence of a post-ghost that doesn't exceede the

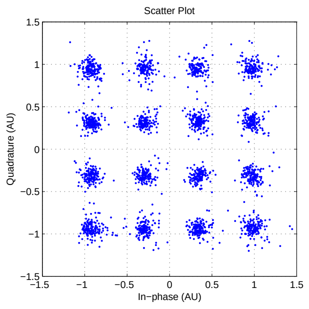
[2]The gain and phase synchronization was implemented as a one-tap equalizer as explained in Appendix B.4



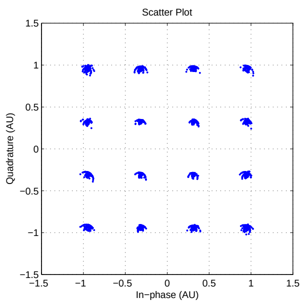
(a) Post-ghost (5 samples).



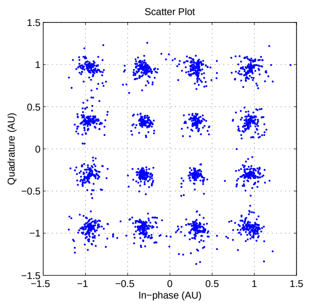
(b) Post-ghost (128 samples).



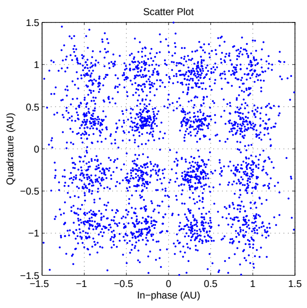
(c) Post-ghost (135 samples).



(d) Pre-ghost (-1 sample).



(e) Pre-ghost (-5 samples).



(f) Pre-ghost (-50 samples).

Figure 3.5.: Scatter plots of OFDM signals received through a channel with a main tap and a pre- or post-ghost tap which is $5\ \mathrm{dB}$ below the main tap power.

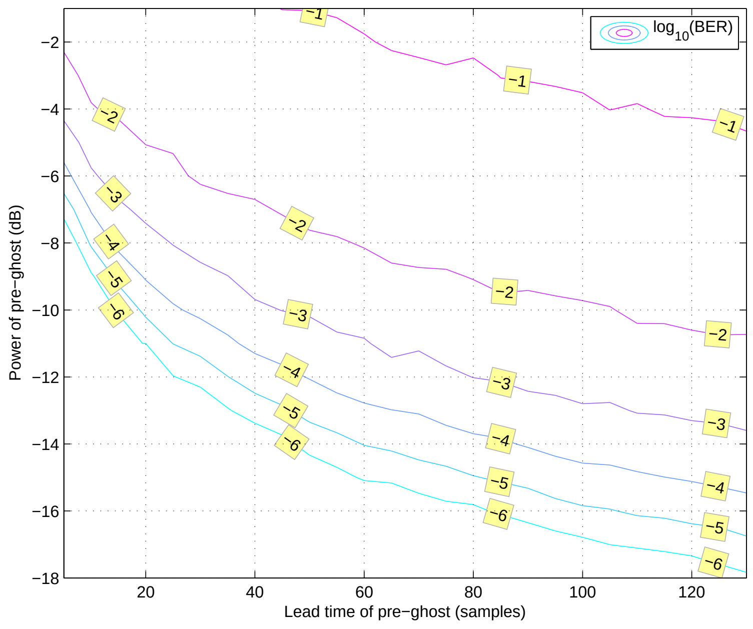


Figure 3.6.: Contour of a BER surface calculated for different values of the lead time and relative power of a pre-ghost.

CP length can be seen in plots (a) and (b) of Figure 3.5. For a delay of up to 128 samples the QAM symbols are perfectly recovered, without any distortions. A delay longer than 128 samples quickly degrades the signal, as in plot (c). Plots (d), (e) and (f) visualize the effects of a pre-ghost. Already a lead time of 1 sample causes notable distortions.

A more quantitative evaluation of pre-ghost effects is shown in Figure 3.6. It shows bit error rates calculated for different combinations of lead time and pre-ghost power. The contour lines are plotted for integer values of the common logarithm of the error rate, i. e. $\log_{10}(BER)$. The simulation parameters were the same as previously (cf. Table 3.2) and no forward error correction was used. On the x-axis, the lead time is plotted in samples from 5 to 130, which corresponds to time values ranging from 0.625 µs to 16.25 µs.

Not surprisingly, the BER increases for a larger time skew and for a more powerful pre-ghost. If the power of the main transmitted signal (which causes the pre-ghost) is less than one hundredth (-20 dB) of the power of the repeated signal (main tap), its influence can be neglected, alleviating the requirement of a short repeater delay. However, in coverage areas where the BS signal is still existent at a power of about one tenth (-10 dB) of the repeated signal, a short time skew becomes of major importance. Even lead times as low as 5 µs (40 samples) cause unacceptable high error rates.

In summary it can be said that in order to avoid multihop interference in an OCR enhanced environment, either the BS signal has to be very weak (20 dB below the main transmitted signal) or the signal delay introduced by the repeater has to be very short. More precisely, the delay has to be shorter than the amount of early timing (symbol start before the main tap

System	Symbol Time	Guard Time
DAB (Mode I)	1 ms	246 µs
DAB (Mode II)	250 µs	61.5 µs
DAB (Mode III)	125 µs	30.8 µs
DAB (Mode IV)	500 µs	123 µs
DVB-T (8k Mode)	896 µs	28/56/112/224 µs
DVB-T (2k Mode)	224 µs	7/14/28/56 µs
LTE (short CP)	66.7 µs	4.69/5.21 µs
LTE (long CP)	66.7 µs	16.7 µs
Fixed WiMAX (128)	64 µs	8 µs
Mobile WiMAX (512)	91.4 µs	11.4 µs
Mobile WiMAX (1024)	128 µs	16 µs

Table 3.3.: Comparison of useful symbol time and CP duration between different wide-area OFDM-based networks.

or correlation peak) at the receiver, such that the tap of the BS signal falls within the CP. As explained previously, it is hard to state a general rule for the maximum allowable delay, but some authors ([Nasr et al., 2007], [Park et al., 2009]) suggest that the repeater delay shall not exceed 30% of the CP duration.

Table 3.3 lists guard time values of several wide-area OFDM-based networks. It clearly points out the difference between broadcasting networks like DAB and DVB-T and communication networks such as LTE and WiMAX. The former define significantly longer symbol durations and guard times, which are typically in the order of tens to hundreds of microseconds while guard times of communication networks are below 20 µs. Hence it is more challenging to implement OCRs for the latter network type. The repeaters listed in Table 3.1 fulfill the criterion of a delay smaller than 30% of the CP duration for certain WiMAX or LTE configurations. However, these repeaters do not perform analog/digital conversion and digital signal processing in the forwarding path. A feedback interference cancellation system as described in Chapter 4 requires conversion to digital domain and is therefore expected to increase the repeater delay considerably. This limits the OCR to be installed in areas with a sufficient attenuation of the BS signal.

Chapter 4

Feedback Interference Cancellation

In the previous chapter the problem of feedback interference in an on-channel repeater (OCR) was explained. As mentioned, this problem can be alleviated either by sufficient isolation between input and output antenna (e. g. through walls or specific antenna patterns) or actively by a feedback interference cancellation (FIC) system. This chapter describes and analyzes an FIC system based on adaptive filtering in detail. What is more, the performance and stability of the system is evaluated analytically and through simulations.

4.1. Description of the FIC System

The principal idea of an FIC system is to employ an adaptive filter which attempts to replicate the feedback signal and then subtracts this replica from the input signal. Figure 4.1 illustrates this mechanism. Note that this is a simplified block diagram, which will be justified in Section 4.2.

The signal received on the input antenna $rx[n]$ is the sum of the main transmitted signal $s[n]$ (coming from the base station during downlink or from the subscriber station during uplink) and the feedback signal $fb[n]$ coming from the output antenna. The latter signal is the part of the retransmitted signal $tx[n]$, which, after being attenuated by possible isolation and multipath environment, arrives at the input antenna. Common abstractions of the feedback channel impulse response $h_{FB}[n]$ were discussed previously in Section 3.2. The received signal $rx[n]$ is subsequently amplified by a repeater gain G and retransmitted. All delays occurring on the forwarding path (due to analog and digital processing) are subsumed under the delay variable d.

Interference cancellation is done by an FIR filter, whose coefficients w_k in ideal case equal those of the feedback channel. Running the transmit signal through this filter yields an estimate of the feedback signal $\widehat{fb}[n]$, which can then be subtracted from $rx[n]$ to recover the original signal. To adjust the filter coefficients, an adaptive algorithm is used. From the algorithm's point of view, $tx[n]$ is the input and $fb[n]$ is the desired signal. However, the latter is not available for calculations inside the repeater. Instead, a "noisy" version $rx[n]$ has to be used, where the otherwise wanted signal $s[n]$ is regarded as measurement noise. The difference between the "noisy desired" signal $rx[n]$ and the estimate $\widehat{fb}[n]$ can be understood as an estimation error,

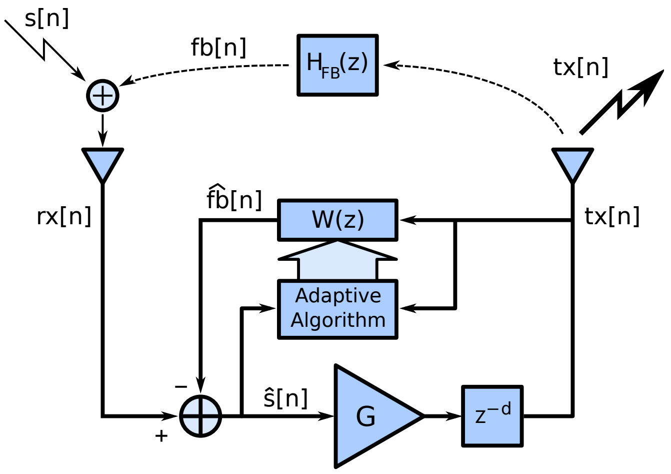


Figure 4.1.: Simplified block diagram of an OCR with feedback interference cancellation.

which forms the basis for the cost function, that has to be minimized by the adaptive algorithm.

Due to its simplicity and its good performance, the well-known *least mean squares* (LMS) algorithm [Haykin, 2002, Chap. 5] was chosen to perform the filter weight updates. It uses a cost function which is defined as the mean square value of the estimation error, that is

$$J[n] = \mathbf{E}\left\{|\hat{s}[n]|^2\right\}. \tag{4.1}$$

$\mathbf{E}\{\cdot\}$ denotes the expectation operator taken with respect to the ensemble of realizations of the stochastic process. In the following, $\mathbf{E}\{\cdot\}$ will always refer to the ensemble average.

The LMS algorithm is an iterative algorithm, calculating a new filter coefficient vector $\mathbf{w}[n+1]$ using the previous weights $\mathbf{w}[n]$, the estimation error $\hat{s}[n]$ and the last N samples of the filter input signal $\mathbf{tx}[n]$. The number of filter weights is given by N and the bold variables denote vectors defined as

$$\mathbf{w}[n] = [\,w_1[n]\;\; w_2[n]\;\; \cdots\;\; w_N[n]\,]^T, \tag{4.2}$$

$$\mathbf{tx}[n] = [\,tx[n]\;\; tx[n-1]\;\; \cdots\;\; tx[n-N+1]\,]^T. \tag{4.3}$$

Equations (4.4) to (4.6) below show the calculation steps of the algorithm, using the variable names of Figure 4.1. Since the FIC system operates on complex baseband signals, it is important to use the complex version of LMS. This means that the filtering operation in Equation (4.4) has to be carried out with the complex conjugate of the filter coefficients ($\mathbf{w}^H[n]$ denotes the hermitian transposition of the coefficient vector) and also the estimation error $\hat{s}[n]$ in Equation (4.6) has to be complex conjugated.

1. Calculate the filter output:

$$\widehat{fb}[n] = \mathbf{w}^H[n] \cdot \mathbf{tx}[n] \tag{4.4}$$

2. Calculate the estimation error:

$$\hat{s}[n] = rx[n] - \widehat{fb}[n] \tag{4.5}$$

3. Update the filter coefficients:

$$\mathbf{w}[n+1] = \mathbf{w}[n] + \mu \cdot \hat{s}^*[n] \cdot \mathbf{tx}[n] \tag{4.6}$$

The variable μ is the iteration step width which can be chosen as a performance parameter. For the adaptive filter to be stable, μ has to be smaller than a certain limit. A conservative rule for the maximum value is

$$\mu < \frac{2}{N \cdot \sigma^2}, \tag{4.7}$$

where σ^2 refers to the power of the filter input, i. e. the signal $tx[n]$. However, simulations presented in Subsection 4.6.2 show that μ has to be significantly smaller than this limit to give acceptable performance of the FIC system.

It can be shown [Moschytz and Hofbauer, 2007], that this algorithm minimizes the cost function of Equation 4.1, causing the filter coefficients to converge towards the taps of the feedback channel. A further derivation of the cost function towards the minimum achievable error is performed in Section 4.3

4.2. Z-Domain Model of the Repeater

In order to get a good understanding of the behaviour of the repeater and the impact of the FIC on the system, it is useful to have an analytic description of the overall system in z-domain. Having derived a system transfer function allows to analyze the frequency response and draw conclusions on the stability of the system. The derivation is done in two steps. In the following subsection a very simple amplify and forward repeater without FIC is analyzed. Afterwards the FIC is taken into consideration and the changes to the repeater transfer function are examined.

4.2.1. Analysis of the Repeater without FIC

Figure 4.2 shows an abstracted model of the repeater. All signals are discrete-time baseband signals, sampled at a rate dependent on the bandwidth of the incoming OFDM signal. Feedback

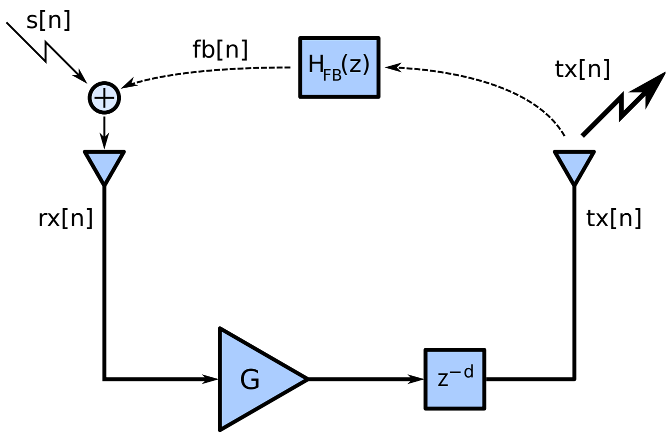


Figure 4.2.: OCR without feedback interference cancellation.

interference happens before the input antenna, where the incoming main signal $s[n]$ adds up with the feedback signal $fb[n]$, before entering the repeater.

$$rx[n] = s[n] + fb[n] \tag{4.8}$$

The received signal is then amplified by a gain G and retransmitted. Between the antennas and the discrete-time baseband processing blocks of the repeater a series of analog and mixed signal processing components (amplifiers, filters, up-/down- converters, analog/digital converters) can be found. Under ideal assumptions, these can be omitted for the z-domain model. However, they do introduce a propagation delay of the signals running through the repeater, which can't be neglected. The sum of all delays on the forward path is collected in a discrete delay of d samples.

$$tx[n] = G \cdot rx[n-d] \tag{4.9}$$

The feedback signal is the result of a convolution of the transmitted signal with the channel impulse response $h_{FB}[n]$ on the feedback path. This impulse response incorporates the propagation delay, multipath effects and attenuation (due to physical isolation measures) of the feedback signal. Single taps of the impulse response may also be subject to Doppler shifts in a non-stationary environment.

$$fb[n] = tx[n] * h_{FB}[n] \tag{4.10}$$

Performing a z-transform on the above three equations yields

$$Rx(z) = S(z) + Fb(z) \tag{4.11}$$

$$Tx(z) = z^{-d} \cdot G \cdot Rx(z) \tag{4.12}$$

$$Fb(z) = Tx(z) \cdot H_{FB}(z)\,. \tag{4.13}$$

Substituting these equations into one another and eliminating $Rx(z)$ and $Fb(z)$ gives the transfer function of the overall repeater system, including feedback interference,

$$H_{Rep}(z) = \frac{Tx(z)}{S(z)} = \frac{z^{-d}G}{1 - z^{-d}GH_{FB}(z)}. \tag{4.14}$$

The desired behaviour of the repeater can be seen from the numerator of Equation (4.14). It describes an amplification by the aspired gain G and a delay of d samples. The denominator is caused by the unwanted feedback channel and generates poles in the transfer function. For better understanding, the feedback channel shall be simplified in the following, to consist of only a single tap which delays the transmitted signal by τ samples and attenuates it by a factor A,

$$H_{FB}(z) = Az^{-\tau}. \tag{4.15}$$

The transfer function thus adopts the form

$$H_{Rep}(z) = \frac{z^{-d}G}{1 - AGz^{-(d+\tau)}}. \tag{4.16}$$

To determine the stability of the repeater it is necessary to find the poles of its transfer function, i. e. the roots of the denominator of Equation (4.16). It turns out that $H_{Rep}(z)$ has $d+\tau$ poles at locations

$$z_k = (AG)^{\frac{1}{d+\tau}}\, e^{j\frac{2\pi k}{d+\tau}}, \qquad k = 0,\ldots,d+\tau-1 \tag{4.17}$$

For the system to be stable, all poles must lie within the unit circle i. e. $|z_k| < 1$. This results in the condition

$$A \cdot G < 1, \tag{4.18}$$

which is intuitively understandable. The feedback attenuation must be stronger than the forward amplification for the loop effects to fade away. In practice the product of A and G should be below -15 dB to guarantee stability (cf. Section 3.2).

Figure 4.3(a) shows the pole-zero plot of a repeater with a forward gain of 30 dB and a delay of 5 samples. The feedback channel has an attenuation of -40 dB at a delay of 4 samples. This system has 4 zeros at the origin of the z-plane caused by the repeater delay and nine poles equally spaced on a circle with radius $(AG)^{1/(d+\tau)}$. The corresponding frequency response is shown in Figure 4.3(b). In ideal case the frequency response would be flat at 30 dB, but the feedback channel causes a ripple between about 27 and 34 dB. A smaller value for the product of A and G reduces the ripple and thus flattens the frequency response.

4.2.2. Z-Domain Model of the Repeater with FIC

In order to find an accurate representation of the repeater with feedback interference cancellation, careful consideration of the signal delays inside the system is necessary. In Figure 4.4(a) a rather implementation specific block diagram of the repeater is shown. For an accurate feedback channel estimation the transmit signal should be measured as close to the output antenna as possible. Several sources ([Lee et al., 2009], [Park et al., 2009]) suggest to pick up the signal even after the power amplifier (HPA). This in turn requires additional analog modules for gain control, frequency down-conversion and A/D-conversion. The abstracted model, shown in Figure 4.4(b) treats all analog processing chains in the repeater as independent delays. The receive chain at the input antenna is abstracted as a delay of d_1 samples, the transmit chain as d_2 samples and the receive chain for the transmit signal as d_3 samples. Furthermore, the model contains two new signals compared to Figure 4.2, $\widehat{fb}[n]$, which is an estimate of the feedback signal and $\hat{s}[n]$, which is an estimate of the main signal $s[n]$. With these definitions, the z-domain equations of the system can be written as

$$Tx(z) = \widehat{S}(z) \cdot G \cdot z^{-d_2} \tag{4.19}$$

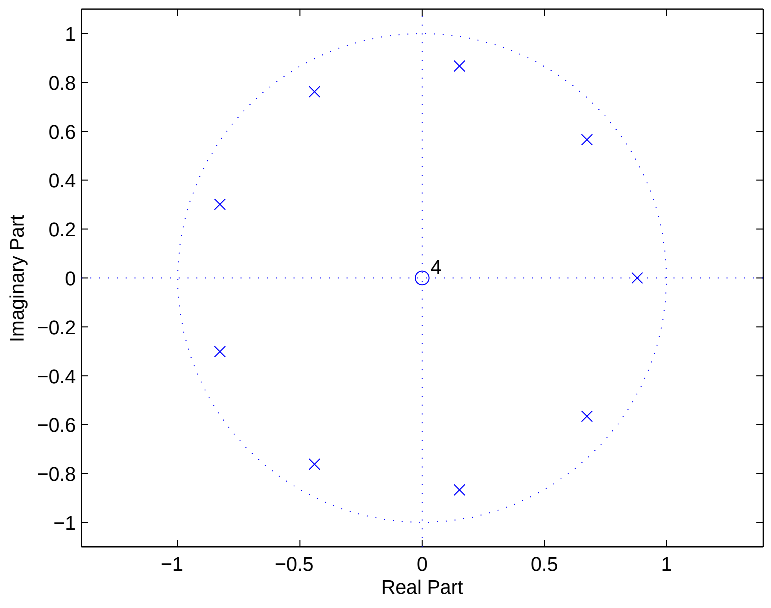
$$\widehat{S}(z) = Rx(z) \cdot z^{-d_1} - \widehat{Fb}(z) \tag{4.20}$$

$$Rx(z) = S(z) + H_{FB}(z) \cdot Tx(z) \tag{4.21}$$

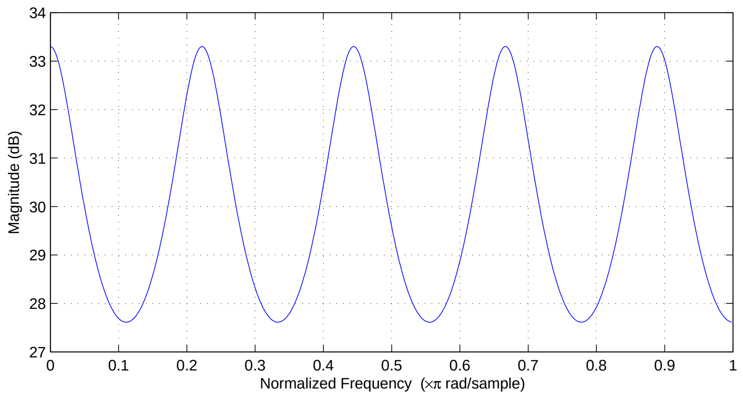
$$\widehat{Fb}(z) = W(z) \cdot Tx(z) \cdot z^{-d_3}. \tag{4.22}$$

Again the transfer function is found through substitution and rearrangement.

$$H_{Rep}(z) = \frac{Tx(z)}{S(z)} = \frac{z^{-(d_1+d_2)}G}{1 - \left(z^{-d_3}H_{FB}(z) - z^{-d_1}W(z)\right) z^{-d_2}G} \tag{4.23}$$

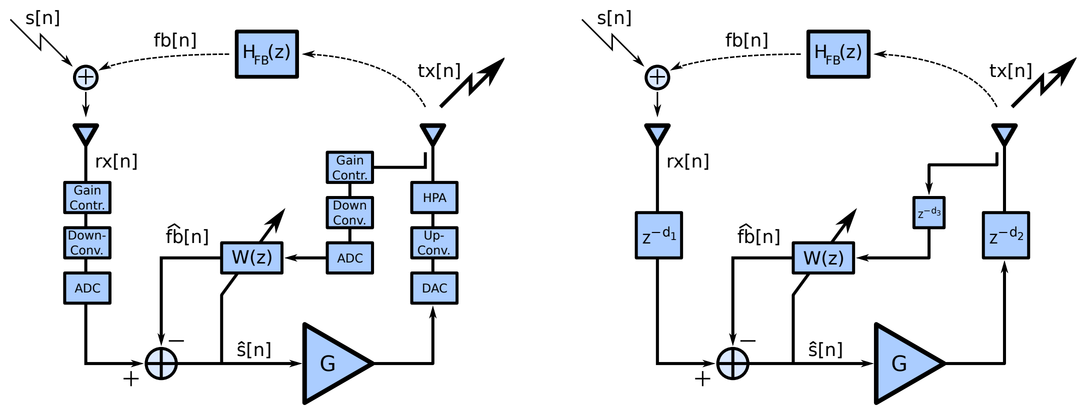


(a) Pole-zero plot.



(b) Magnitude of frequency response over $\omega = 0 \ldots \pi$.

Figure 4.3.: Z-domain analysis of a repeater with $G = 30$ dB, $d = 5$, $A = -40$ dB, $\tau = 4$.



(a) Detailed version showing analog components.

(b) Analog components abstracted through delays.

Figure 4.4.: Block diagram of an OCR with feedback interference cancellation.

Owing to the high similarity of the two down-conversion paths at input and output antenna, it can be assumed that $d_1 = d_3$. This allows the following simplification of the transfer function.

$$
\begin{aligned}
H_{Rep}(z) &= \frac{z^{-(d_1+d_2)}G}{1 - z^{-(d_1+d_2)}G\left(H_{FB}(z) - W(z)\right)} \\
&= \frac{z^{-d}G}{1 - z^{-d}G\left(H_{FB}(z) - W(z)\right)}
\end{aligned}
\tag{4.24}
$$

In the last step the sum of d_1 and d_2 has been substituted for the total repeater delay d. The final form of the transfer function shows a high similarity with the transfer function of the repeater without FIC in Equation (4.14). The transfer function of the adaptive filter $W(z)$ is subtracted from $H_{FB}(z)$ and therefore alleviates its impact. In case of perfect adaptation, i. e. $W(z) = H_{FB}(z)$, the denominator disappears and the repeater transfer function takes its desired form $H_{Rep}(z) = Gz^{-d}$.

Figure 4.1 in the beginning of this chapter shows a simplified model of the repeater with FIC. Using the same derivations as above, it can be shown that this model has the transfer function given in Equation (4.24). It is thus an appropriate description of the system under the assumption $d_1 = d_3$ and will be used throughout this work.

4.3. Analytical Assessment of the Adaptation Algorithm

In Section 4.1 the LMS algorithm was reviewed and the cost function based on the mean squared error was stated. The cost function will be further derived in this section, which gives insight on conditions to yield a minimum error of the FIC. For simplicity, all signals in this section will be treated as real valued.

Assuming a stationary environment and a time index n big enough such that all transient (learning) behaviour is over, the cost function $J[n]$ approaches its steady state value J_{ss}. This value can be written as

$$\begin{aligned} J_{ss} &= \mathbf{E}\left\{\hat{s}^2[n]\right\} = && (4.25)\\ &= \mathbf{E}\left\{(rx[n] - \mathbf{w}^T\mathbf{tx}[n])^2\right\} = && (4.26)\\ &= \mathbf{E}\left\{rx^2[n] - 2rx[n]\,\mathbf{w}^T\mathbf{tx}[n] + \mathbf{w}^T\mathbf{tx}[n]\,\mathbf{tx}^T[n]\,\mathbf{w}\right\} = && (4.27)\\ &= \mathbf{E}\left\{rx^2[n]\right\} - 2\mathbf{w}^T\mathbf{E}\left\{rx[n]\,\mathbf{tx}[n]\right\} + \mathbf{w}^T\mathbf{E}\left\{\mathbf{tx}[n]\,\mathbf{tx}^T[n]\right\}\mathbf{w}. && (4.28) \end{aligned}$$

Note that the expectation operator still denotes an ensemble average, and not a time average. The time index is only dropped due to the steady state condition. The last step of the above derivation took advantage of the linearity of the expectation operator and the fact that in steady state the coefficient vector $\mathbf{w}$ equals the Wiener solution and can be taken out of the expectation operator. With $rx[n] = s[n] + fb[n]$, the first part of Equation (4.28) can be written as

$$\begin{aligned} \mathbf{E}\left\{rx^2[n]\right\} &= \mathbf{E}\left\{\left(s[n] + \mathbf{h}_{FB}^T\mathbf{tx}[n]\right)^2\right\} = && (4.29)\\ &= \mathbf{E}\left\{s^2[n] + 2s[n]\,\mathbf{h}_{FB}^T\mathbf{tx}[n] + \mathbf{h}_{FB}^T\mathbf{tx}[n]\,\mathbf{tx}^T[n]\,\mathbf{h}_{FB}\right\} = && (4.30)\\ &= \sigma_s^2 + 2\mathbf{h}_{FB}^T\mathbf{p} + \mathbf{h}_{FB}^T\mathbf{R}_{tx}\mathbf{h}_{FB}. && (4.31) \end{aligned}$$

Three variables were introduced here, σ_s^2, which is the variance of the input signal, $\mathbf{R}_{tx}$ being the auto-correlation matrix of the transmit signal and $\mathbf{p}$, which is the cross-correlation vector between $s[n]$ and $tx[n]$. They are defined under the steady state condition as

$$\sigma_s^2 = \mathbf{E}\left\{s^2[n]\right\} \qquad \mathbf{R}_{tx} = \mathbf{E}\left\{\mathbf{tx}[n]\,\mathbf{tx}^T[n]\right\} \qquad \mathbf{p} = \mathbf{E}\left\{s[n]\,\mathbf{tx}[n]\right\}. \tag{4.32}$$

The second term of Equation (4.28) can be be derived in a similar manner.

$$\begin{aligned} \mathbf{E}\left\{rx[n]\,\mathbf{tx}[n]\right\} &= \mathbf{E}\left\{\left(s[n] + \mathbf{h}_{FB}^T\mathbf{tx}[n]\right)\mathbf{tx}[n]\right\} = && (4.33)\\ &= \mathbf{p} + \mathbf{R}_{tx}\mathbf{h}_{FB} && (4.34) \end{aligned}$$

Plugging Equations (4.31) and (4.34) into Equation (4.28) gives

$$\begin{aligned} J_{ss} = \sigma_s^2 &+ 2\mathbf{h}_{FB}^T\mathbf{p} - 2\mathbf{w}^T\mathbf{p}\\ &+ \mathbf{h}_{FB}^T\mathbf{R}_{tx}\mathbf{h}_{FB} + \mathbf{w}^T\mathbf{R}_{tx}\mathbf{w} - 2\mathbf{w}^T\mathbf{R}_{tx}\mathbf{h}_{FB}. \end{aligned} \tag{4.35}$$

The last three terms of this equation can be simplified to a very compact form, as proven in [Moschytz and Hofbauer, 2007, p. 55]. This gives the following final form

$$J_{ss} = \sigma_s^2 + 2\left(\mathbf{h}_{FB} - \mathbf{w}\right)^T\mathbf{p} + \left(\mathbf{h}_{FB} - \mathbf{w}\right)^T\mathbf{R}_{tx}\left(\mathbf{h}_{FB} - \mathbf{w}\right). \tag{4.36}$$

One obvious result from Equation (4.36) is that the minimum error decreases if the coefficient vector $\mathbf{w}$ approaches the channel impulse response $\mathbf{h}_{FB}$. If the two are equal, the remaining error is σ_s^2, which is the variance of the desired signal.

If they are not exactly equal, the cross-correlation vector $\mathbf{p}$ and the autocorrelation matrix $\mathbf{R}_{tx}$ influence the smallest achievable error. As is known from adaptive filter theory, white noise input has ideal properties. An OFDM signal very well resembles white noise, except for two correlation peaks coming from the cyclic prefix. This is depicted in Figure 4.5 which

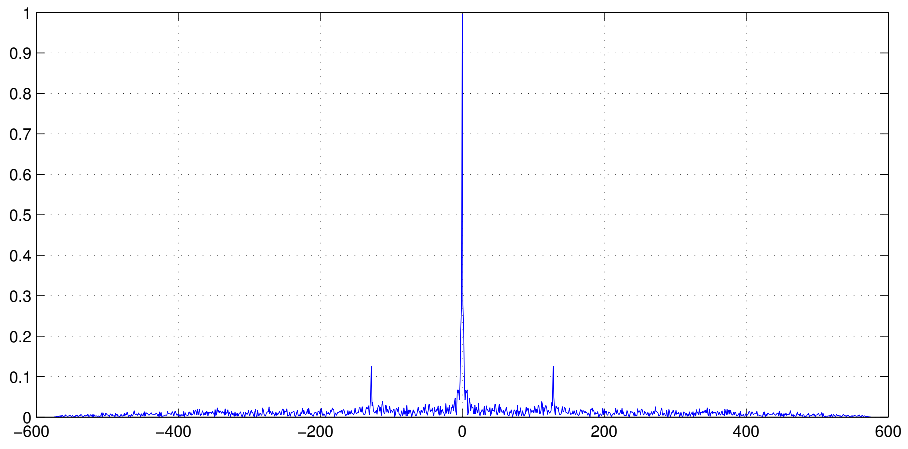


Figure 4.5.: Autocorrelation function of an OFDM signal (FFT-size 128).

shows the autocorrelation function of an OFDM signal with an FFT size of 128 and a cyclic prefix length of 16. The correlation peaks at a distance of 128 samples off the center are clearly visible (averaging over 10 realizations was performed in this plot).

Figure 4.6 visualizes the autocorrelation matrix of a WiMAX signal as a gray scale image. Lighter colors in this image refer to higher correlation values. The WiMAX signal consists of a preamble and three data symbols. Four squares were added to the image, which enframe the autocorrelation matrix of each individual OFDM symbol. The bright line along the main diagonal of the matrix corresponds to the variance of each sample. In the upper right and lower left corner of each data symbol, two shorter lines are visible, which come from the cyclic prefix. The only symbol which does not have these distinct properties is the preamble. The reason is that it contains the exact same data in each realization and hence is not effected by averaging.

Other than the autocorrelation matrix, the cross-correlation vector is more problematic. The signal $s[n]$, which acts as a measurement noise for the adaptive system would in best case be uncorrelated with the filter input $tx[n]$. In the repeater however, the two signals are perfectly correlated. In fact, ideally they are just scaled and time-shifted versions of one another.

Figure 4.7 shows plots of the cross-correlation function, with FIC switched off (a) and on (b). They were calculated for a repeater delay of 180 samples and a single tap feedback channel with a delay of 20 samples. The input was again a WiMAX signal with an FFT size of 128. Both plots have a main correlation peak at a time skew of 180 samples. Without FIC, this peak repeats itself every loop delay number of samples ($180 + 20 = 200$) again, causing peaks at 380, 580, 780, etc. The plot with FIC clearly shows that these repeated signal components are removed. However, two minor peaks remain, located 128 samples left and right of the main peak, which originate from the cyclic prefix. These can also be seen in the first plot, around each of the replicated main peaks.

The plot in Figure 4.7(c) shows a close-up image of the main peak, which is essentially equal for both versions (with or without FIC). It highlights the fact, that this peak extends over about 5 samples on both sides, which notably differ from zero. The implication is that the N samples contained in the correlation vector $\mathbf{p}$ should be sufficiently far away from the main correlation

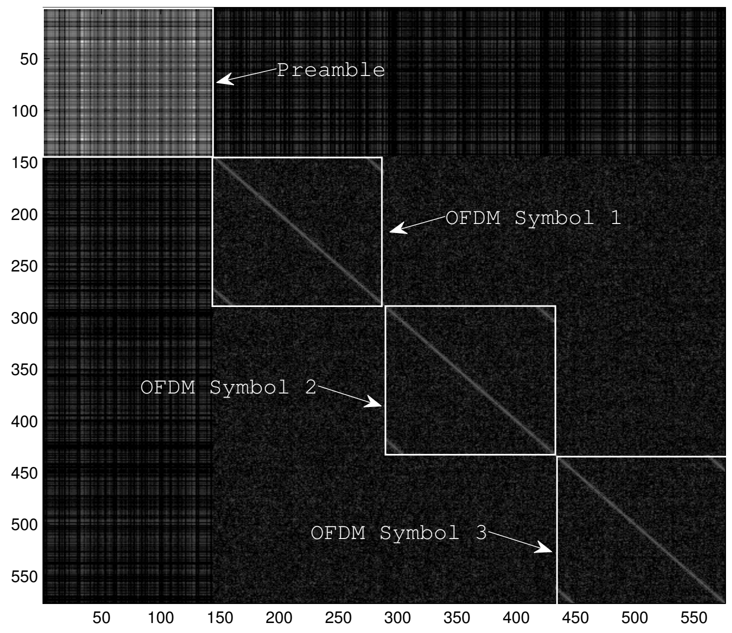


Figure 4.6.: Visualization of the autocorrelation matrix of a WiMAX signal consisting of a preamble and 3 data symbols.

peak. In other words, a certain minimum repeater delay on the forwarding path is required to decorrelate $s[n]$ and $\mathbf{tx}[n]$. With a typical baseband sample period of 178 ns (as defined by the WiMAX standard for a 5 MHz channel), the decorrelation delay should be at least 1 $\mu\mathrm{s}$. On the other hand, the overall forward delay should not be big enough as to generate correlations with the cyclic prefix. The effects of correlation depending on different delay values are highlighted in bit error rate simulations, which will be presented in Section 4.6.3.

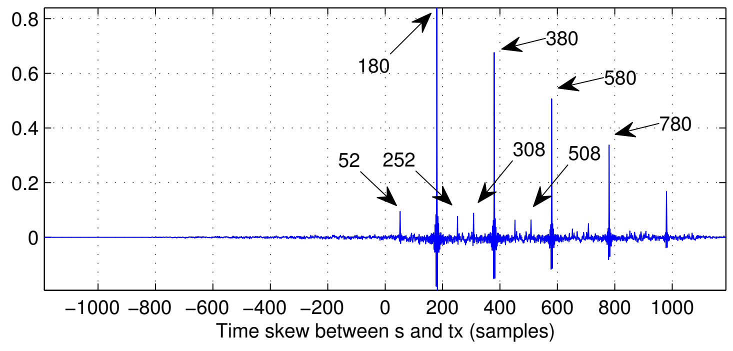
4.4. Learning Curves

A common way of evaluating the performance of an adaptive system is to examine its learning curve and derive parameters like adaptation rate and steady state error. Often, the cost function of the adaptation algorithm is used as learning curve. In the LMS algorithm this is defined as the mean-square value of a given error signal. Mathematically that means

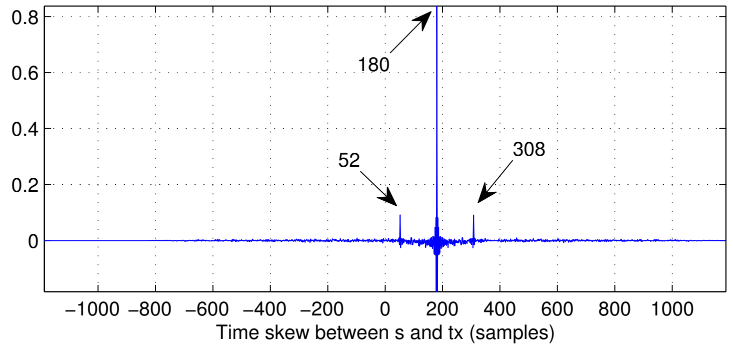
$$J[n] = \mathbf{E}\left\{|e[n]|^2\right\}, \tag{4.37}$$

where $J[n]$ is the learning curve and $e[n]$ is the error signal (i. e. the difference between a desired signal and the actual filter output).

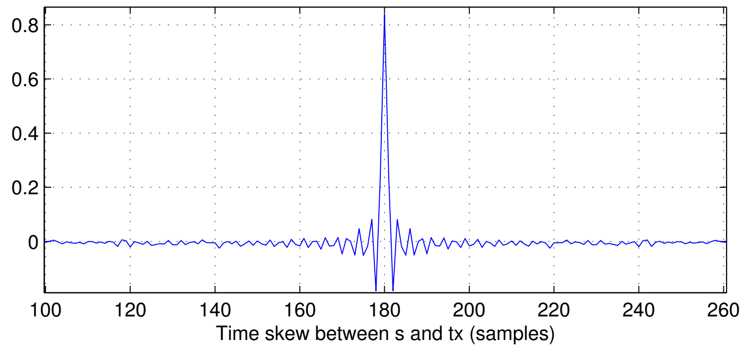
In the given repeater architecture other definitions for the learning curve are possible and it is worth to investigate, which is preferable for evaluating the performance of the feedback cancellation system. Below, three different definitions of error signals and their according mean-



(a) Without FIC



(b) With FIC



(c) Zoom on the main peak at 180 samples

Figure 4.7.: Cross-correlation function between $s[n]$ and $tx[n]$.

square values are given, which can be used to plot learning curves. These will be discussed and compared subsequently.

- Adaptation error, mean-squared error (MSE)

$$e[n] = rx[n] - \widehat{fb}[n] = \hat{s}[n] \tag{4.38}$$

$$MSE[n] = \mathbf{E}\left\{|e[n]|^2\right\} \tag{4.39}$$

- Coefficient error vector, mean-squared deviation (MSD)

$$\mathbf{v}[n] = \mathbf{w}[n] - \mathbf{h}_{FB} \tag{4.40}$$

$$MSD[n] = \mathbf{E}\left\{\|\mathbf{v}[n]\|^2\right\} \tag{4.41}$$

- Repeater-introduced distortion, residual error power (REP)

$$\varepsilon[n] = tx[n] - G \cdot s[n-d] \tag{4.42}$$

$$REP[n] = \mathbf{E}\left\{|\varepsilon[n]|^2\right\} \tag{4.43}$$

The *adaptation error* $e[n]$ is the previously mentioned difference between the desired signal and the adaptive filter output. In terms of the adaptation algorithm, the *desired signal* is the otherwise unwanted feedback signal. It is measured as a “noisy” value $rx[n]$, corrupted by the main transmitted signal $s[n]$. Recalling Figure 4.1 it can be seen that the adaptation error equals the estimated main signal $\hat{s}[n]$. The ensemble average of the mean-squared value of the adaptation error is frequently referred to as the *mean-square error* (MSE) and is identical to $J[k]$ in Equation (4.37).

The second definition, the *coefficient error vector* or *weight error vector* $\mathbf{v}[n]$, is given by the difference between the coefficient vector of the adaptive filter and the tap vector of the feedback channel. Note that this requires that the two vectors have to be equal length, and, if this is not the case, the shorter one has to be padded with zeros. If the filter vector is shorter, it means that the adaptive filter has too few taps to ideally replicate the feedback channel. This causes an error which is also reflected in the error vector (due to padding of $\mathbf{w}[n]$ rather than truncation of $\mathbf{h}_{FB}$). Taking the expected value of the length of this vector, which is defined as the squared Euclidean norm, gives the *mean-squared deviation* MSD.

While the first two definitions are typical performance metrics of adaptive systems, the third is specific to the OCR. Assuming that an ideal, noise-free signal is received at the input antenna, it describes the distortions introduced in the forwarded signal caused by the feedback loop. To this end, the transmitted signal $tx[n]$ is compared to the ideal transmit signal, which is the main received signal $s[n]$ times the repeater gain G. A possible delay d of the repeater has to be taken into account for this calculation. The mean-squared value of this error signal is referred to as *residual error power* REP, since it is a measure for the remaining noise, which the FIC cannot remove.

Figures 4.8, 4.9 and 4.10 show learning curves based on the three discussed metrics. In each figure, the left plot shows the curve in a dB scale and the right plot in a linear scale, both over the iteration number n. The curves were calculated using unit variance white Gaussian noise

Parameters	Specifications
Input signal power	0 dB
Sampling period	178.5 ns
Repeater gain	30 dB
Repeater delay	3 µs (17 samples)
Feedback channel type	Single tap
Feedback delay	1 µs (6 samples)
Feedback attenuation	−30 dB
Adaptive filter size	8 baseband taps
LMS step size μ	1.25e-6

Table 4.1.: Repeater setup parameters for calculating learning curves

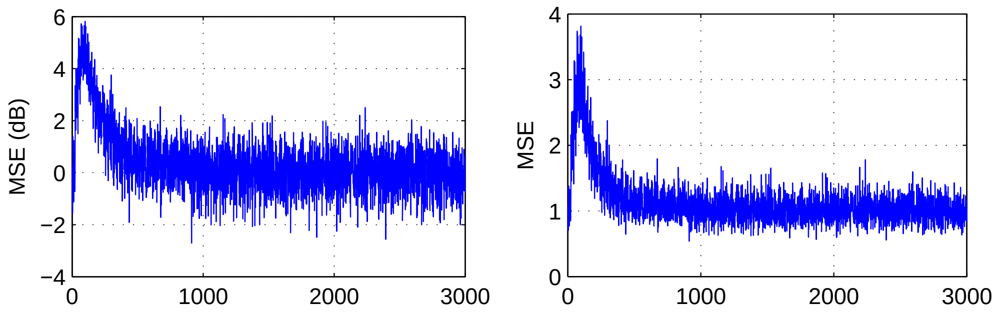


Figure 4.8.: Learning curve based on mean-squared error

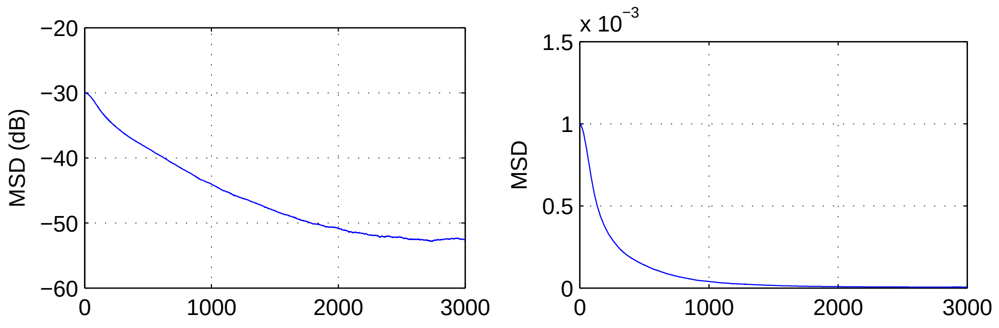


Figure 4.9.: Learning curve based on mean-squared deviation

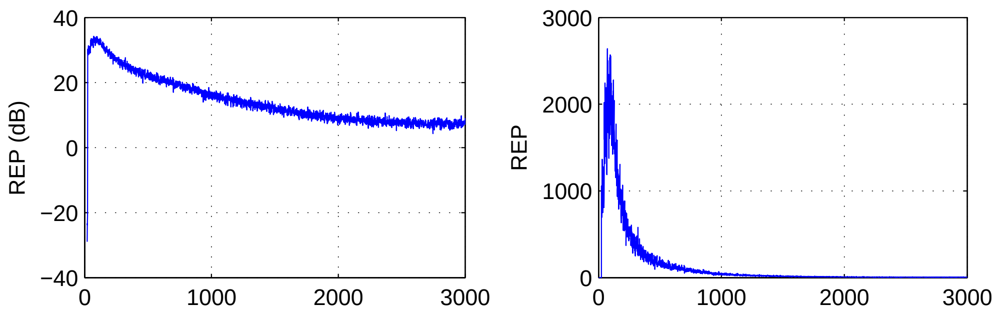


Figure 4.10.: Learning curve based on residual error power

as input and a repeater setup as summarized in Table 4.1. The expectation operator has been approximated by averaging over 70 independently calculated curves.

Each of the three definitions has a different advantage over the others, which determines its usage. The MSE is the only metric which can be measured in the actual implemented system, whereas the other two require signals which are not available inside the repeater ($s[n]$ and $\mathbf{h}_{FB}$). Therefore it is the only possible choice as a cost function for the adaptation algorithm. However, as can be seen in Figure 4.8, the learning curve is very noisy (even after strong averaging) and what is more, it is not monotonously decreasing. During approximately the first 70 samples, the MSE increases, before it reaches a maximum and starts descending again. This phenomenon can be explained by the fact, that initially the received signal $rx[n]$ perfectly equals the input signal $s[n]$. The first non-zero tap of the feedback signal arrives at the input antenna only after the accumulated delay through the repeater (d samples) and the feedback channel (τ samples until the first tap of $h_{FB}[n]$). In the terms of the adaptation algorithm, the desired signal ($fb[n]$) is zero and only noise ($s[n]$) is available for learning. In total it takes $2d + \tau$ samples (40 in the above example) until the feedback loop for the first time influences $tx[n]$. During this time the error is very low, because it is only influenced by minor variations in the filter coefficients. Afterwards the feedback signal becomes apparent and the error increases rapidly. After about one more loop duration most of the transient behaviour is over and adaptive algorithm starts counteracting, thereby decreasing the error signal again.

The MSD metric does not suffer from this phenomenon, as it compares the instantaneous state of the adaptive filter to the actual taps of the physical feedback channel. This not only makes the curve much smother (even with a small number of averages), but also lets it descend monotonously during the entire training phase. These properties make the MSD perfectly suited for measuring performance characteristics of the adaptation algorithm such as learning rate and steady state error.

The learning curve based on the REP has a very similar shape to the MSE curve. It also suffers from the same deficiencies, as it is not smooth and not monotonous. However, it is the one metric which best describes the quality of the overall system, depicting the noise introduced between input and output of the repeater. Hence the REP can be used to compare different architectures and FIC systems.

4.5. The Simulation Model

Testing and evaluating the performance of the FIC is done through simulations in Matlab. This requires functions for an OFDM transmitter (Tx), a receiver (Rx) and an on-channel repeater. Due to the original focus on WiMAX systems, a detailed implementation of the Mobile WiMAX physical layer was coded and is explained in Appendix B. Details and code listings of the OCR implementation are given in Appendix C.

In order to cover a wide range of effects occurring in an OCR-enhanced environment, a simulation environment as close to reality as possible was designed. Figure 4.11 shows this model. It contains four transmission channels emulating the propagation effects on the radio signal. The channels on the segments Tx – OCR and OCR – Rx shall be called *Channel A* and *Channel B*, and the feedback interference is caused by the *FB Channel*.

Channel C represents the direct path between Tx and Rx, which causes multihop interfer-

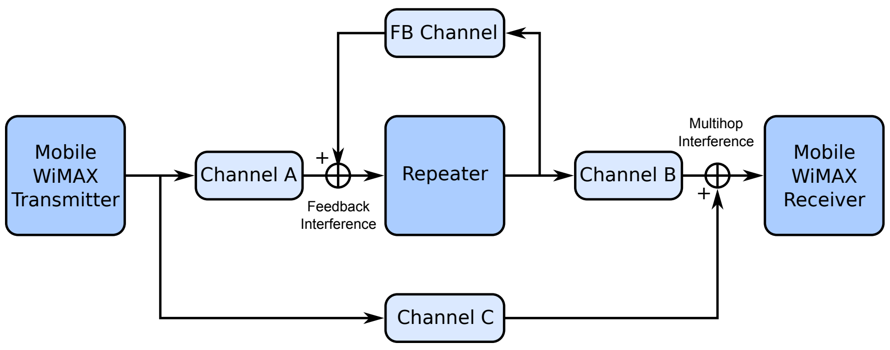


Figure 4.11.: Comprehensive version of the simulation model.

ence. This channel has several profound implications, making the entire model significantly more complex. In particular, two questions are raised:

- What is the ratio of the power of the retransmitted signal (output of channel B) to the power of the direct signal (output of channel C)?
- What is the timing difference between these two signals and how should the receiver cope with this?

To find a realistic answer for the first question, it was decided to equip the channels with a path loss model (while initially it was planned to simulate only small-scale fading and additive white Gaussian noise). The reduction of average signal power by a channel shall be calculated from parameters specifying the distance and terrain type of the channel. A model for rural and suburban environments suitable for WiMAX systems was taken from [Erceg et al., 2001]. It defines the path loss PL in dB by

$$PL = A + 10\gamma \log_{10}\left(\frac{d}{d_0}\right) + s \qquad \text{for } d > d_0, \tag{4.44}$$

where A and γ are parameters depending on wavelength, base station height, and some constants for the terrain type (hilly, flat, forest, suburban). The propagation distance is defined in d (in meters) and d_0 is 100 m. The variable s adds log-normal distributed shadowing effects, which are of minor importance for this simulation and were therefore ignored.

Hence, the average powers of the repeated (P_{rep}) and the direct signal (P_{dir}) are

$$P_{rep} = P_{Tx} - PL_A + G_{OCR} - PL_B \tag{4.45}$$
$$P_{dir} = P_{Tx} - PL_C, \tag{4.46}$$

with all variables given in dB. P_{Tx} is the output power of the transmitter (normalized to 0 dB in all simulations), G_{OCR} is the repeater gain and PL_x is the path loss on channel A, B or C. Naturally, PL_C is greater than PL_A or PL_B, due to greater distance.

As an example, imagine a setup with 1000 m distance on both channel A and B, and 2000 m at channel C. In a flat, suburban terrain this would give path losses of about 68 dB and 82 dB respectively. If the repeater adjusts its gain such that the output power level is 0 dB again, the repeated signal would be 14 dB stronger than the direct signal in this example. If the distance of channel B was only 400 m, resulting in a path loss of 50 dB, the power ratio would be 32 dB instead.

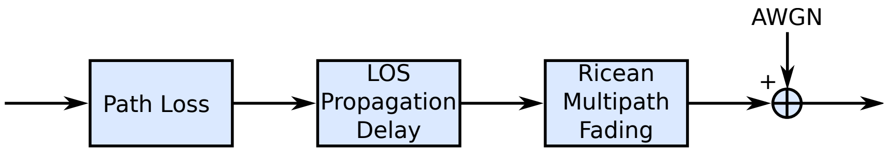


Figure 4.12.: Fully featured model of a transmission channel.

The second question concerns the unequal propagation delay on the two paths. Having specified the distance d of a channel, the introduced delay δ (line-of-sight condition) is easily calculated by

$$\delta = \frac{d}{c} \tag{4.47}$$

where c is the speed of light. Any additional delays caused by non-line-of-sight conditions can be specified through multipath fading parameters. In the implementation the delay is rounded to an integer multiple of the sampling period. The signals, which are handled as vectors containing all baseband samples of an entire WiMAX frame, are then shifted according to their delay, by zero-padding the vectors. At the multihop interference point, the vectors from the two paths are padded to equal length, summed up and handed over to the receiver. The receiver in turn has to decide on which sample to start recording the OFDM symbols of the frame. Thus it requires a timing synchronization mechanism, which was implemented by correlating the received signal with the known frame preamble. The symbol start is scheduled at the peak of the correlation function.

Figure 4.12 shows the components of the fully featured channel model. The first two blocks are the path loss and line-of-sight propagation delay discussed so far. The third block realizes small-scale fading and uses the *Ricean Channel* class of the Matlab Communication Toolbox (Version 4.3). An object of this class requires several parameters, among them delay, gain, Doppler shift and Ricean k-factor of each single multipath component. To chose realistic values for these parameters, the *Stanford University Interim* (SUI) model was used. It defines 6 sets of parameter values (labeled SUI-1 through SUI-6), associated with different terrain types and subscriber mobility. The precise definition of the models can be found in [Erceg et al., 2001].

The last stage in the channel is the addition of white Gaussian noise (AWGN) at arbitrary power. This allows to specify a signal-to-noise ratio (SNR), which is used in many simulations to calculate BER curves.

In summary, a great deal of work was put into an accurate description of the channels. However, simulations with this environment gave very bad results. It was first at this point, that dramatic effects of a pre-ghost (which is caused by channel C) were discovered. Subsequently it was decided to investigate the pre-ghost issue separately (which was done in Section 3.3) and remove channel C from the simulation environment. Accordingly, the blocks for path loss and propagation delay were obsolete and removed as well. What is more, the timing synchronization block in the receiver was also no longer necessary and therefore discarded.

The simplified model, which was finally used for the simulations is shown in Figure 4.13. The AWGN just before the receiver was used to vary the receiver SNR. In most simulations also the multipath fading channel was disabled. Note that all discussions about channel models so far only apply to channels A, B and C. The feedback channel instead is implemented inside the repeater and for practical reasons simply as a stationary FIR filter. This is explained in detail in

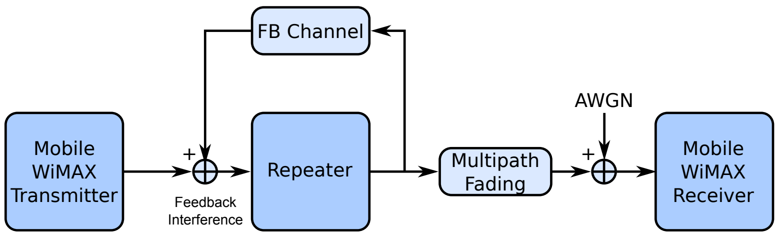


Figure 4.13.: Simplified version of the simulation model.

Appendix C.

4.6. Simulation Results

This section presents the results of several simulations which confirm previously found properties of the repeater. In each simulation, the adaptive filter was given 30 coefficients, which is enough to identify a feedback channel impulse response of up to 5 µs (28 samples).

4.6.1. Stability Regions

A theoretical criterion for the stability of the system was derived in Section 4.2 and states that without FIC the attenuation on the feedback path must be stronger than the forward gain of the repeater. For a feedback channel impulse response with a single tap of amplitude A this means

$$A < \frac{1}{G}. \tag{4.48}$$

If FIC is enabled, this criterion is relaxed since the adaptive filter counteracts the feedback channel as was shown in Equation (4.24). The stability was then experimentally tested by simulations, in which random signals were sent through the system and checked if the output exceeds unnaturally high values. The limit was chosen to be 20 times the variance of the output signal which is a compromise between a too low level that could trigger even in stable operation, and a too high value which would require unnecessary long simulation times.

The forward gain was kept constant at 30 dB and a single-tap feedback channel with varying delay and attenuation was used. For each combination of delay and attenuation, the simulation was run multiple times and finally stored a binary value which states if an instability condition was detected. The outcome can be seen in Figure 4.14. It shows the feedback attenuation on the vertical axis, with weak attenuation on the upper, and strong attenuation on the lower end. Logically, the top area of the image contains instable conditions. Areas in which no instability was detected are visualized with another color. Four such areas exist in the figure, which were calculated for different values of the forward delay.

Somewhat surprisingly, the system tends to be less stable for shorter forward delay. With $d = 1$ µs, instability occurs at feedback powers above -24 dB, which is an improvement of 6 dB compared to the theoretical limit without FIC at -30 dB (or an improvement of 21 dB compared to the practical limit, when considering a margin of 15 dB as described in Section 3.2). At a forward delay of 12 µs on the other hand, the system shows stable behaviour below

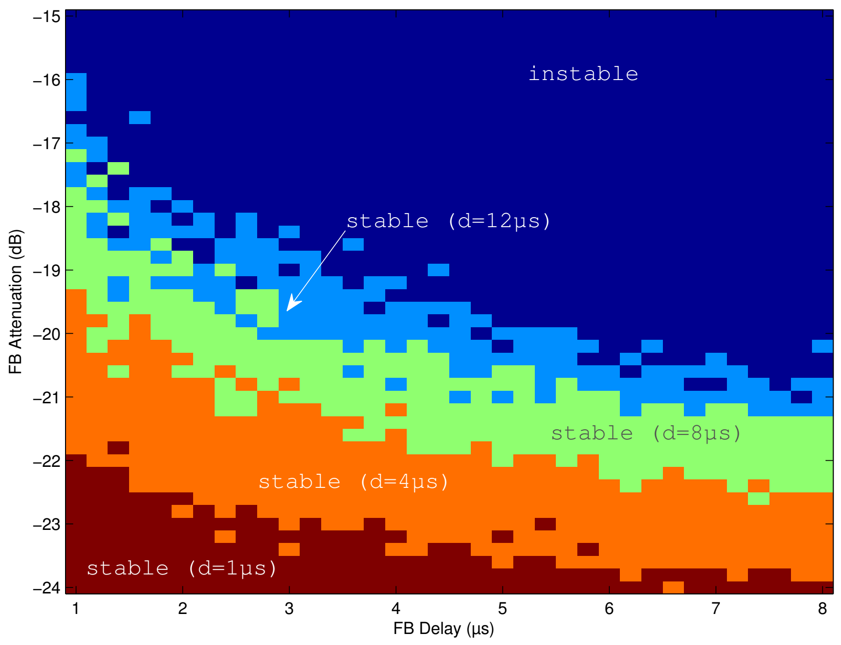


Figure 4.14.: Stability regions of the repeater with FIC.

-21 dB, giving an improvement of 9 dB. Though not entirely clarified, this difference might be caused by correlation effects, occurring at short delays (cf. Section 4.3).

Note that the instabilities in this simulation are only caused by the outer feedback channel and not by the adaptation algorithm. Care was taken to use LMS parameters (step width and filter size) which guarantee a stable operation.

4.6.2. LMS Step Size Dependency

The step size of the adaptation algorithm has a profound influence on the overall system performance. In Section 4.1, an upper limit for the step size μ was given by

$$\mu < \frac{2}{N \cdot (avg.\ input\ power)}. \tag{4.49}$$

The average input power is the variance of the transmit signal, which is 30 dB in this simulation setup. Using $N = 30$ filter coefficients, the above equation gives a maximum step width of $6.7 \cdot 10^{-5}$. However, with this step size the LMS algorithm performs highly inaccurate, which basically renders the device useless. Reducing the step size increases the accuracy.

Figure 4.15 shows bit error rate calculations for different step size values. Additionally it contains two curves showing the performance of the repeater without FIC and the ideal case without a feedback channel. These curves represent boundaries, within which the FIC curves should lie for successful operation. The figure clearly shows the bad performance when μ is 10^{-6} or greater. In these cases the algorithm inaccuracy introduces noise into the output signal,

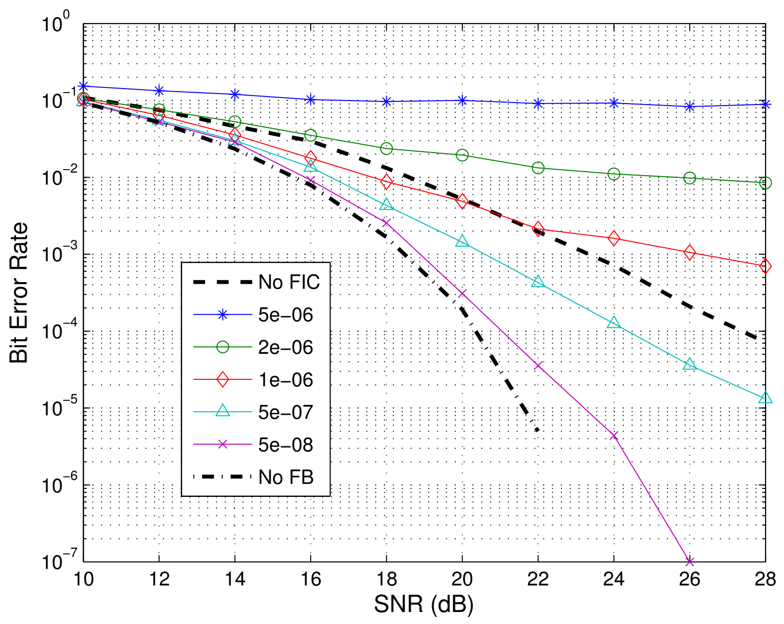


Figure 4.15.: Bit error rate performance evaluation for different values of the step size of the LMS algorithm.

which exceeds the noise on the channel and causes the system to perform even worse than if no FIC was employed.

On the other hand, a smaller step size reduces the bit error rate. In fact, the ideal curve, which corresponds to a theoretical repeater without feedback interference, can be approached arbitrarily close, by choosing a sufficiently small step size. The drawback of a small step size is that it takes longer to train the filter and that its ability to track channel variations is reduced. However, non of these is a major issue for the repeater. Training the filter is done when the system is powered and takes at most a few milliseconds, even if several thousands of samples are used. Tracking is also not a problem, considering that the feedback channel is highly static for a non-mobile repeater. Occasionally a feedback channel description with Doppler shifts of up to 10 Hz can be found in the literature [Lee et al., 2009]. This corresponds to a coherence time of the channel of about $100\ \mathrm{ms}$, which can be easily tracked at any practically used step size.

4.6.3. Influence of the Repeater Delay

In Section 4.3 the correlation properties between the received signal $s[n]$ and the retransmitted signal $tx[n]$ were discussed. It was shown that, on the one hand a certain delay within the repeater is required to decorrelate the signals, and on the other hand the delay should not be as large as to cause correlation with the cyclic prefix. These effects are very well visible in Figure 4.16, which shows a BER plot over a range of delay values.

At very low delays, the repeater performs terribly bad, with bit error rates as high as 10%. This comes from the strong correlation between input ($tx[n]$) and noise ($s[n]$) of the adaptive

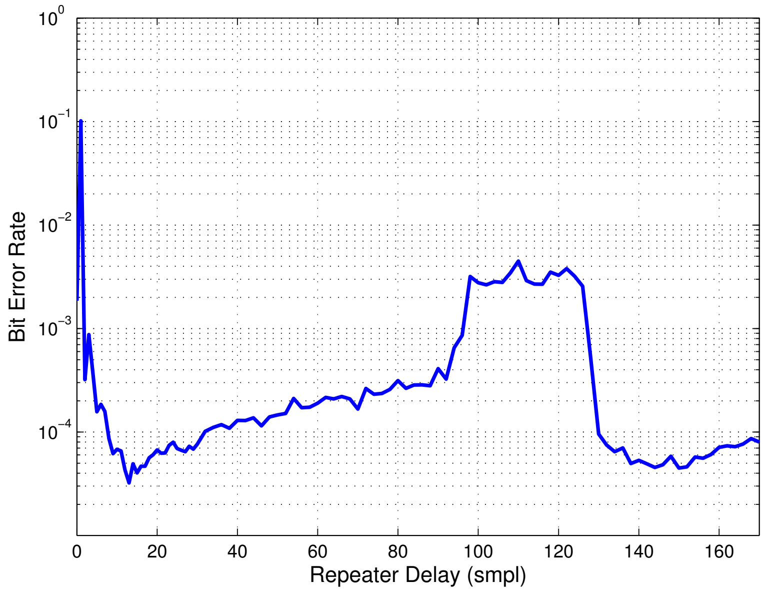


Figure 4.16.: Bit error rate depending on the forward delay of the repeater.

system. Only at delays larger than about 6 samples the BER drops to about 10^{-4}, which is an expected value for a chosen channel SNR of 24 dB and an LMS step size of $5 \cdot 10^{-7}$ (cf. Figure 4.15).

The second effect can be seen between 98 and 128 samples. These values come from the chosen FFT size of 128 (and the according time shift of the cyclic prefix) and the adaptive filter length of 30 coefficients. The impact is not as strong in this case, because only 16 (length of the CP) out of 144 (length of the entire OFDM symbol) samples are identical. Nevertheless, the BER is increased by a factor of more than 10, compared to surrounding values. A delay in this area should therefore be avoided.

4.6.4. Multipath Feedback channel

This simulation shows the performance of the FIC system if the feedback channel contains three multipath components. Table 4.2 summarizes delay and gain values of the three taps. The simulation result is shown in Figure 4.17. While the curve without FIC is strongly degraded in this scenario and does not reach BER values below 10^{-3}, the FIC curve is essentially unaffected (the step size in this simulation was 10^{-7}).

	Delay μs	**Delay** samples	**Amplitude** dB
Path 1:	0.17	1	-40 dB
Path 2:	1.25	7	-40 dB
Path 3:	5.00	28	-45 dB

Table 4.2.: Multipath components of the feedback channel.

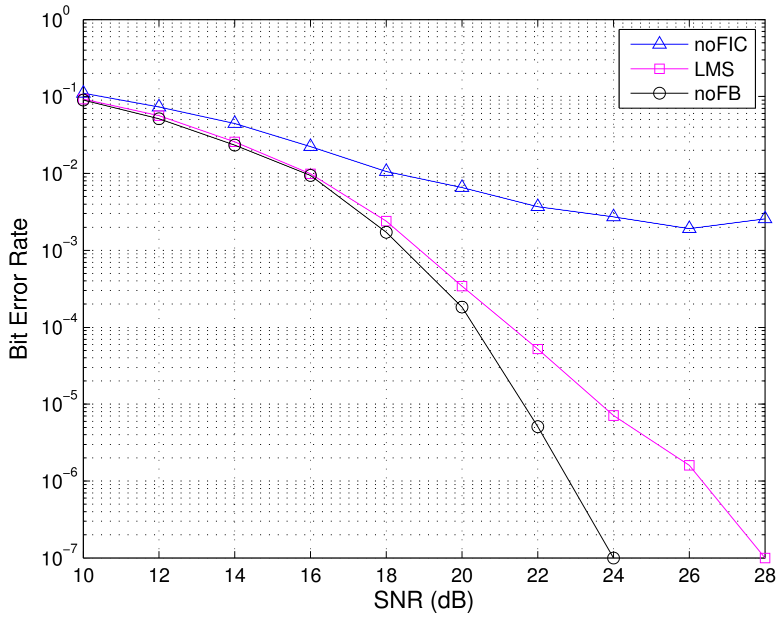


Figure 4.17.: Bit error rate in case of a challenging multipath feedback channel.

Chapter 5

Summary and Conclusion

The work presented in this thesis consists of two major parts: first a general case study over wireless repeater schemes and second a detailed investigation of one chosen architecture. While the first part was mostly a literature research, the second part rests upon theory of adaptive filtering and numerical analysis through simulations. The general focus throughout this work were wireless OFDMA-based communication networks, and in particular Mobile WiMAX (IEEE 802.16-2009).

In Chapter 2 the case study was presented. It summarized the various information forwarding schemes and classified them from different points of view. The most important type of classification is based on the ISO OSI model, which groups the schemes into Layer 1 repeaters, Layer 2 relays and layer 3 routers. It was shown that in Layer 1 repeaters sophisticated signal processing such as demodulation and decoding is usually not possible, due to the high delay introduced. Layer 2 relays on the other hand allow time-intensive processing, by re-scheduling received data into later time slots. The problem with this kind of relays is that considerable changes in the standard are required. In the case of WiMAX, these changes are defined in the amendment 802.16j, which had not been available during the first part of this work. Nevertheless, a preliminary overview over this amendment was given in Section 2.4.

For the second part it was decided to focus on a Layer 1 on-channel repeater. The concept of this architecture was explained in Chapter 3 and the two major challenges, feedback interference and multihop interference were treated in detail. Feedback interference arises from insufficient isolation between receive and transmit antenna and can cause instability of the whole system. Multihop interference arises when the receiver picks up the signal of both transmitters, the base station and the repeater. In such a scenario, the channel impulse response as seen by the receiver includes a so-called pre-ghost. This can mislead the timing synchronization algorithm at the receiver and thereby strongly degrade the signal quality. The impact of the pre-ghost phenomenon was addressed empirically through simulations. To alleviate this problem, either the direct signal (from the base station) has to be strongly attenuated, or the repeater delay has to be shorter than a fraction of the OFDM guard interval, such that the time skew between pre-ghost and main tap is very short.

Chapter 4 addressed the problem of feedback interference and an active cancellation system was presented. The idea is to imitate the feedback channel with an adaptive filter, run the transmit signal through this filter and subtract the result from the input signal. To adapt the filter coefficients, the least mean squares (LMS) algorithm was used. The performance of this system was examined in several ways analytically and through simulations. First, a z-domain model was derived in Section 4.2, which allowed to locate the poles of the system and draw conclusions on its stability.

In Section 4.3 the cost function of the adaptive system was evaluated. It was shown that the minimum achievable error depends on the correlation between input and output signal of the repeater. Undesired correlation effects arise, if the time skew between these signals is either very short, or large enough to be in the vicinity of the cyclic prefix of the OFDM signal. Hence, a short delay is required to decorrelate the two signals. Section 4.4 introduced three performance metrics of the system, which were used to plot learning curves with different properties.

A lot of work was spent on developing a realistic simulation model. Section 4.5 explains the considerations, which led to a highly complex model and why this model was abandoned later on. The simulations presented in Section 4.6 are based on a much simpler model. They highlight stability areas of the system and show the bit error rate performance under various conditions.

Given the focus on WiMAX networks, a comprehensive implementation of the WiMAX physical layer was created in the course of this work. An overview over the WiMAX specifications is given in Appendix A. The Matlab implementations of the WiMAX transceiver and the on-channel repeater are explained in the Appendices B and C.

Concluding this work it can be said that a feedback interference cancellation system not only enhances the antenna isolation (and thereby increases stability), but also substantially improves the quality and bit error rate of the overall system. However, the performance strongly depends on a reasonable choice of parameters such as LMS step width, decorrelation delay and maximum gain. These have to be kept in mind for a successful operation.

A major issue left for investigation is multihop interference. A repeater with an FIC system might introduce a propagation delay which is too long to be covered by the short cyclic prefix of the WiMAX signal. This could be the reason why – to the author's knowledge – no WiMAX repeater with FIC system has been implemented so far (in contrast to broadcasting networks like DAB and DVB). To clarify this issue, a hardware implementation and field tests would be required. However, it must be conceded that the more promising technology for WiMAX is Layer 2 relaying, which is inherently unaffected by feedback or multihop interference.

Appendix A

Overview over the Mobile WiMAX PHY Layer

This appendix gives a brief overview over the IEEE 802.16 standard family and then summarizes the specification of the Mobile WiMAX physical layer, as defined in [IEEE 802.16-2009]. For an in-depth treatment of the standard, the books [Andrews et al., 2007] and [Nuaymi, 2007] are recommendable.

A.1. History and Features of WiMAX

The IEEE 802.16 working group was founded in 1998 with the goal to define an air-interface specification for a broadband wireless metropolitan area network (WMAN). The original idea was to place single transmission sites e. g. in sparsely populated rural areas, to which receivers could establish a link via directional antennas. Hence, the initial focus was on high-frequency line-of-sight (LOS) connections. The first version of the standard was completed in 2001 and specified a single-carrier time-division multiplexed (TDM) system in the frequency range from 10 GHz to 66 GHz.

Since then, the focus has gradually shifted towards a cellular network with non-line-of-sight (NLOS) connections and mobility support. The first amendment, 802.16a, specified an NLOS OFDM-based air-interface in the 2 GHz to 11 GHz frequency band. This later resulted in a new standard called IEEE 802.16-2004, which was still aimed at fixed installations.

Mobility was first introduced with amendment 802.16e-2005. Through an OFDMA physical layer, it supports dynamic bandwidth allocation based on the current number of subscribers and individual data rate requirements. Furthermore it allows different values for the FFT size (i. e. the numbers of OFDM subcarriers), thus giving network operators more flexibility in selecting an ideally suited configuration. This amendment and all previous revisions are merged into the so far newest standard IEEE 802.16-2009. Table A.1 gives an overview over the different versions of 802.16.

The standards are highly comprehensive in their nature and specify a multitude of options and configuration parameters, which can be combined in innumerable ways. To ease device implementation and compatibility, a reduced set of options is desirable. This is accomplished by

	802.16	**802.16-2004**	**802.16e-2005**
Completed	December 2001	June 2004	December 2005
Frequency Band	$10 - 66$ GHz	$2 - 11$ GHz	$2 - 11$ GHz (fixed) $2 - 6$ GHz (mobile)
Application	Fixed LOS	Fixed NLOS	Fixed and mobile NLOS
PHY Layer	Single Carrier	OFDM	OFDMA
Channel Bandwidth	$20 \ldots 28$ MHz	$1.25 \ldots 15$ GHz	$1.25 \ldots 15$ GHz
FFT-Size	n/a	256	128, 512, 1024, 2048
WiMAX System Profile	none	Fixed WiMAX	Mobile WiMAX

Table A.1.: Overview over IEEE 802.16 standards.

the *WiMAX Forum*, which is a non-profit consortium of about 370 member organizations (as of April 2010), including equipment manufacturers and network operators. The WiMAX Forum creates implementations with chosen values for parameters such as frequency band, channel bandwidth, duplexing scheme, FFT size etc. Such reference implementations are referred to as *certification profiles*, since manufacturers can get their products certified for compatibility with such a profile. Certification profiles belonging to a specific standard are grouped together as *system profiles*. Currently two system profiles exist, *Fixed WiMAX*, referring to 802.16-2004 and *Mobile WiMAX*, referring to 802.16e-2005. The certification profiles defined thus far are listed in Table A.2.

In the following, an incomplete list of the more outstanding features of WiMAX is given [Andrews et al., 2007].

- **High data rates:** A single channel in Mobile WiMAX can supply a data rate of about 40 Mbit/s. Individual user can expect data rates between 1 Mbit/s and 5 Mbit/s [WMF-FAQ], depending on the service provided by the operator, and various other factors such as channel conditions and number of users in the network.
- **Adaptive modulation and coding (AMC):** WiMAX supports different baseband modulation types (QPSK, 16-QAM and 64-QAM) and coding rates (1/2, 2/3 and 3/4). Depending on the received signal quality, the modulation and coding scheme is automatically adapted, such that the data rate is maximized while the bit error rate is kept low. It may be different for each user and change on a per-frame basis.
- **Link-layer retransmissions:** To guarantee reliable transmission, WiMAX has an option to require each data packet to be acknowledged by the receiver. Unacknowledged packets are automatically retransmitted (automatic repeat request, ARQ).
- **Different duplexing schemes:** A network operator may chose to provide time-division duplex (TDD) or frequency-division duplex (FDD). The former is usually preferred as it allows less complex (and less expensive) equipment, flexibility in the choice of UL-to-DL data ratio as well as reduced bandwidth requirements. If the network is run in FDD-mode, a subscriber station may optionally use half-duplex (H-FDD) which allows a simplified implementation.
- **Advanced antenna techniques:** The physical layer also includes support for sophisticated multiple-antenna techniques such as transmit diversity (MIMO), beamforming,

Frequency Band	**Channel Bandwidth**	**Duplexing Mode**	**FFT Size**
Fixed WiMAX			
3.5 GHz	7 MHz	TDD	256
	3.5 MHz	TDD	256
	3.5 MHz	FDD	256
	7 MHz	FDD	256
5.8 GHz	10 MHz	TDD	256
Mobile WiMAX			
2.3 – 2.4 GHz	5 MHz	TDD	512
	10 MHz	TDD	1024
	8.75 MHz	TDD	1024
2.305 – 2.360 GHz	3.5 MHz	TDD	512
	5 MHz	TDD	512
	10 MHz	TDD	1024
2.496 – 2.690 GHz	5 MHz	TDD	512
	10 MHz	TDD	1024
3.3 – 3.8 GHz	5 MHz	TDD	512
	7 MHz	TDD	1024
	10 MHz	TDD	1024

Table A.2.: WiMAX certification profiles, as defined by the WiMAX Forum.

Nominal channel bandwidth (MHz)	**3.5**	**5.0**	**7.0**	**8.75**	**10.0**
Subcarrier spacing (kHz)	7.81	10.94	7.81	9.77	10.94
Sample rate (MHz)	4.0	5.6	8.0	10.0	11.2
FFT size	512		1024		
Useful Subcarriers	408		840		
Guard Carriers	103		183		

Table A.3.: Frequency domain parameters of OFDM symbols defined in Mobile WiMAX.

space-time coding and spatial multiplexing.

- **Quality-of-service:** The MAC layer defines service classes for constant bit rate, variable bit rate, real-time, non-real-time and best effort transmission, to suit all sorts of internet traffic.
- **Security:** WiMAX includes link-layer encryption based on the Advanced Encryption Standard (AES) and user authentication through the Extensible Authentication Protocol (EAP).
- **Mobility:** With IEEE 802.16e-2005 mobility was introduced, allowing seamless handovers to SSs moving from the coverage area of one BS to another.

A.2. The Mobile WiMAX Physical Layer

The most recent version of the standard, IEEE 802.16-2009, includes the specification of three different physical layers, *Single Carrier*, *OFDM* and *OFDMA*[1]. Mobile WiMAX, as defined by the WiMAX Forum, is based on the latter type, which is often referred to as *scalable OFDMA*, as it allows a network provider to choose between different FFT sizes.

The four defined FFT sizes correspond to the chosen channel bandwidth, i. e. a wider channel utilizes a larger FFT size. To keep a subcarrier spacing of approximately 10 kHz, the signal is oversampled by a bandwidth dependent factor, such that the actual symbol occupies a larger bandwidth than the nominal channel bandwidth. Table A.3 lists the different values for subcarrier spacing and sample rate (i. e. actual bandwidth of the OFDM symbol) derived from the nominal channel bandwidths specified by the Mobile WiMAX system profile [WMF-T23]. However, for each FFT size a number of guard carriers on both sides of the spectrum are defined, which are not modulated[2]. In this way the non-zero part of the symbol does not exceed the channel bandwidth. Figure A.1 depicts an OFDM symbol in frequency domain for a 5 MHz channel.

An OFDM symbol is constructed in frequency domain and subsequently converted to time

[1]Additionally a fourth PHY layer, WirelessHUMAN (wireless high-speed unlicensed MAN), is given. This is a special configuration of the OFDM-PHY for usage in license-exempt bands

[2]The number of guard carriers also depends on the subcarrier permutation scheme. The values listed in this appendix refer to the UL PUSC scheme, which is explained in Subsection A.2.2

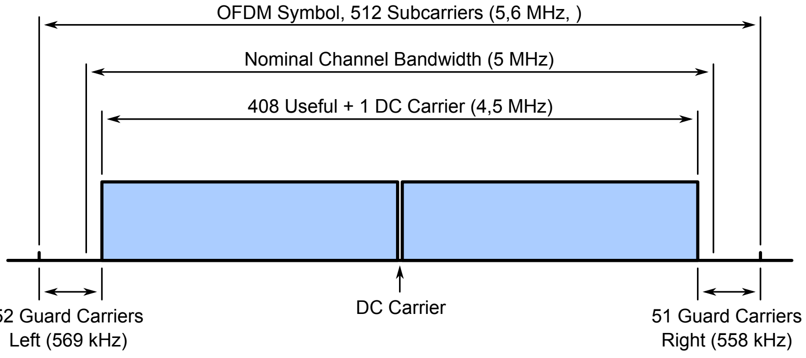

OFDM Symbol, 512 Subcarriers (5,6 MHz,)
Nominal Channel Bandwidth (5 MHz)
408 Useful + 1 DC Carrier (4,5 MHz)
52 Guard Carriers Left (569 kHz)
DC Carrier
51 Guard Carriers Right (558 kHz)

Figure A.1.: Example of an OFDM symbol in frequency domain for a channel bandwidth of 5 MHz.

domain through an IFFT operation. The symbol duration is determined by the inverse of the subcarrier spacing. A subcarrier spacing of 10.94 kHz thus gives a useful symbol time of $T_b = 91.4$ µs. The cyclic prefix (CP) is then generated by taking a number of samples from the end of the symbol and attach them to the beginning. Four different lengths for the CP are defined, given as a fraction of the useful symbol length: $1/4$, $1/8$, $1/16$ and $1/32$. However, only the value $1/8$ is mandatory for Mobile WiMAX. This results in a guard time of $T_g = 11.4$ µs and a total symbol length of $T_s = 102.8$ µs (again for a subcarrier spacing of 10.94 kHz).

A.2.1. The Baseband Transceiver Chain

The fundamental building blocks for baseband signal processing of a WiMAX transmitter are depicted in Figure A.2. Data is handed over from the MAC to the PHY layer in units of *slots*. The amount of data in a frame reserved for an individual user or connection is always an integer multiple of slots. A slot is comprised of 48 modulated data symbols, which is the same for all possible configurations and transmission modes. However, the number of data bits contained in a slot varies with the used modulation scheme and the coding rate. The variable M in Figure A.2 denotes the modulation order (i. e. the number of bits per QAM symbol), which can be 2, 4 or 6, for QPSK, 16-QAM or 64-QAM respectively. The coding rate is denoted by R and can be $1/2$, $2/3$ or $3/4$. Thus a slot, as handed to the PHY layer contains $48 \cdot M \cdot R$ bits.

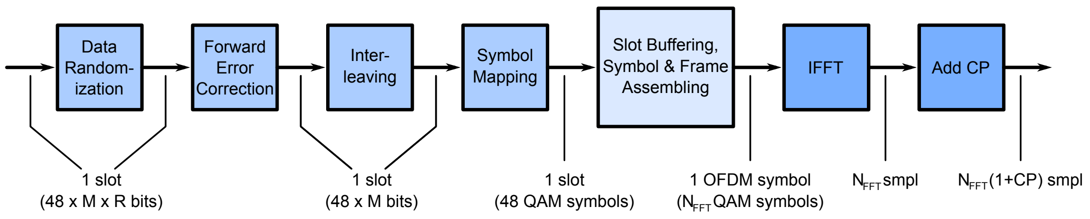


Figure A.2.: Chain of the baseband signal processing blocks of a WiMAX transmitter.

The first step in the transmitter chain is a randomization block. It's purpose it to increase the information-theoretic uncertainty by eliminating long sequences of consecutive zeros or ones. This is done by applying a bit-wise XOR operation on the incoming data stream with a pseudo-noise sequence generated by a linear feedback shift register.

Following the randomization comes a forward error correction (FEC) block. The standard defines five different encoding schemes, whereas only one, *tail-biting convolutional coding*, is mandatory. Tail-biting means that the encoder registers are initialized with the last bits of the slot to be encoded. The output of the encoder is twice as long as the input sequence and subsequently shortened by a puncture pattern to achieve the desired coding rate.

After FEC, the slot is run through an interleaver block which changes the bit order. The permutation is done in two steps: the first causes consecutive bits to be mapped to non-consecutive subcarriers (thereby providing frequency-diversity) and the second ensures that bits are placed in an alternating manner on more or less reliable constellation points.

The subcarrier modulation is performed by the next block, which maps groups of 2, 4 or 6 bits onto QPSK, 16-QAM or 64-QAM constellation points respectively. Note that pilot symbols are not modulated by this block. They are inserted later, when the frame is assembled and modulated through BPSK. After modulation, an additional block for *repetition coding* can be found which is not shown in the figure. Repetition coding means that a slot is simply duplicated a number of times, such that the receiver can employ a majority decision on the symbols of a slot, thereby increasing the chance of correct interpretation in case of poor reception. This is useful if the slot contains critical data that is important to be received without errors. Currently, repetition coding is only used for the frame control header.

A readily processed slot has to be placed inside the WiMAX frame, which is done through a series of non-trivial algorithms. Depending on the transmission scheme the symbols of a slot may be interleaved with pilot symbols, permuted over different subcarrier ranges and distributed over several OFDM symbols. This process is also called *subchannelization* and is explained in more detail in the following subsection. When the frame is prepared and guard carriers are added, the OFDM symbols are converted to time domain by an IFFT block. The final step in digital baseband is the addition of the cyclic prefix.

A.2.2. OFDMA Subchannelization

In Mobile WiMAX groups of subcarriers are arranged as subchannels, which gives rise to the term *subchannelization*. Transmission capacity is assigned to individual users in multiples of subchannels rather than single subcarriers. Furthermore, single OFDM symbols may contain multiple subchannels serving different users. This is the defining difference between OFDM and OFDMA. Pure OFDM allows switching between different users only on a per-symbol basis[3].

Figure A.3 shows the time-frequency-plane of a WiMAX frame in TDD mode. It consists of a downlink subframe and an uplink subframe, separated by transition gaps of one symbol duration, which allow transceivers to switch between transmitting and receiving mode. Accordingly these gaps are called transmit and receive transition gap (TTG and RTG). The first OFDM

[3] To be precise, Fixed WiMAX also employs a weak form of OFDMA in the uplink, even though the PHY layer is referred to as OFDM.

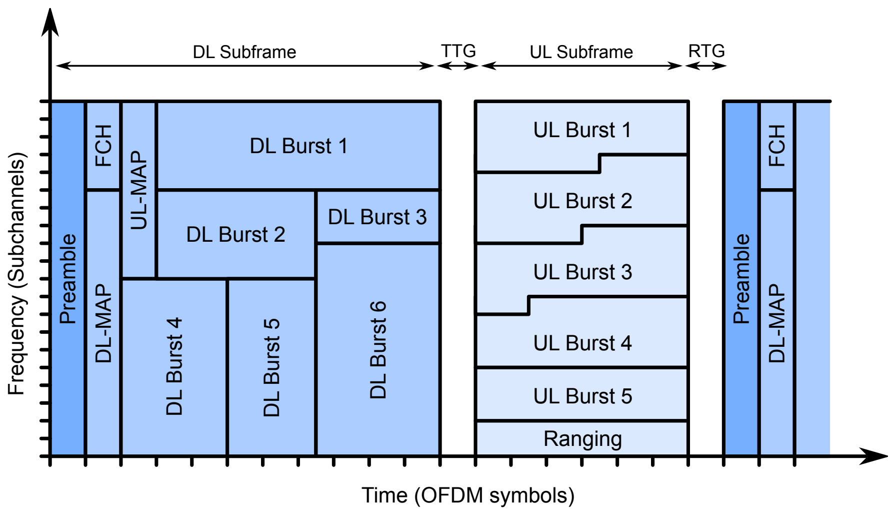


Figure A.3.: Structure of a WiMAX frame in TDD mode.

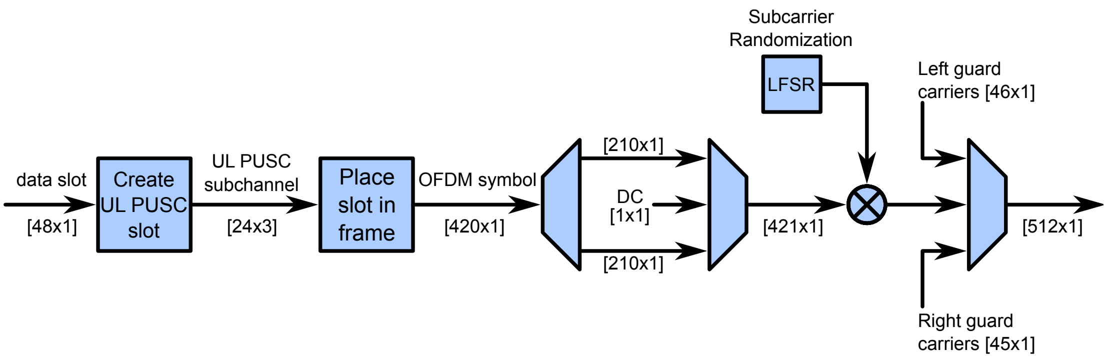


Figure A.4.: Creation of a WiMAX OFDM symbol in case of UL PUSC subcarrier permutation and an FFT size of 512.

symbol of the frame contains the preamble, which is an a priori known sequence, required to perform time and frequency synchronization and initial channel estimation. Immediately following the preamble come several fields with control information for the frame. First, a frame control header (FCH) describes the used modulation, coding rate and length of the following MAP messages. MAP (mapping) messages define how the time-frequency plane is shared among different users. An area in the time-frequency plane reserved to one user is referred to as *data region* or *burst*. Each burst may employ a different modulation and coding scheme that suits the user's channel conditions. The combination of modulation and coding scheme is called *burst profile* and also specified by the MAP messages for each user.

A special region in the UL subframe is reserved for contention-based access. It is mainly used as a ranging channel which subscribers can use to perform frequency and timing adjustments. Furthermore it can be used to make bandwidth requests and send small amounts of best-effort traffic.

The process of mapping the 48 data symbols of a slot onto their designated location in the time-frequency plane of a WiMAX frame is sketched in Figure A.4. In a first step the symbols are rearranged and augmented with pilot symbols to create a subchannel. This can be done by one out of several *subcarrier permutation schemes*, where each of them has certain advantages in terms of data rate, channel tracking capability, frequency diversity, multi-antenna support or frequency reuse in sectorized cells. The existing schemes are *Partial Usage of Subcarriers* (PUSC), *Full Usage of Subcarriers* (FUSC), *Tile Usage of Subcarriers* (TUSC) and *Band Adaptive Modulation and Coding* (Band AMC). Some schemes vary in uplink or downlink mode or may define optional sub-schemes. Within a WiMAX frame several zones (i. e. contiguous OFDM symbols) with different permutation schemes may exist. However, so far only PUSC is specified as mandatory for both BS and MS in the Mobile WiMAX definition.

As an example, the PUSC mode as used in the uplink subframe (in short: UL PUSC) shall be explained briefly. For this mode a numerical example with detailed test vectors exists in [IEEE 802.16-2009, Sec. 8.4.9.4.4]. In this case a data slot is arranged in 24 subcarriers over 3 consecutive OFDM symbols, giving a total of 72 symbols (48 data symbols and 24 pilot symbols). The arrangement is shown in Table A.4, where (for once) the horizontal axis refers to frequency (subcarriers) and the vertical axis refers to time (OFDM symbols). The cells with white background refer to the data symbols of the input slot, which comes from the modulation block. The pilot symbols are located at the cells with gray background. However, at this stage they are not modulated and only contain the real value "1". Modulation of the pilot symbols happens later at subcarrier randomization.

Subcarriers $\longrightarrow$

0	0	1	1	2	2	3	3	4	4	5	5	6	6	7	7	8	8	9	9	10	10	11	11
12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35
12	36	37	13	14	38	39	15	16	40	41	17	18	42	43	19	20	44	45	21	22	46	47	23
Tile 0				Tile 1				Tile 2				Tile 3				Tile 4				Tile 5			

Table A.4.: Structure of an UL PUSC subchannel occupying 24 subcarriers over 3 time slots. Gray fields mark the location of pilot symbols, while all others are data symbols.

A subchannel as in Table A.4 consists of 6 tiles with 4 consecutive subcarriers by 3 time slots each. The tiles are taken from the entire set of useful subcarriers and permuted from logical to physical tile number by an algorithm specified in [IEEE 802.16-2009, Sec. 8.4.6.2.2]. This permutation scheme allows only a subset of the available tiles to be used, giving rise to the name *partial usage of subcarriers*. If a BS splits its coverage zone into several sectors, a different subcarrier group may be used on each sector, which increases the frequency reuse factor of the network.

Once all data slots which are to be included in the frame are buffered and the contained symbols are placed on their designated subcarrier and OFDM symbol index, the frame is read from the buffer in a per-OFDM-symbol fashion. At this stage an OFDM symbol consists only of useful subcarriers, that is 420 for an FFT size of 512, as in the example in Figure A.4. After the DC carrier is inserted in the center of the spectrum, subcarrier randomization is performed. As defined in [IEEE 802.16-2009, Sec. 8.4.9.4.1], this is a multiplication of the symbols of each subcarrier with a pseudo-random sequence of the values "-1" and "+1", generated by a linear feedback shift register (LFSR). This multiplication also causes a BPSK modulation of the pilot symbols, which were initially only reserved with the value “1”. The LFSR is clocked once for each OFDM symbol, thus shifting the pseudo-random sequence one subcarrier per time slot. In the final step, the required number of guard carriers are added on both sides of the spectrum, which concludes the assembly of the symbol in frequency domain.

Appendix B

Implementation of a WiMAX Transceiver in Matlab

This appendix describes the implementation of a transceiver, which was created in the course of this thesis and realizes the Mobile WiMAX physical layer specification. The aim was to provide an accurate simulation environment that allows signal processing and bit error calculations on signals which closely resemble real-world WiMAX data.

The entire transceiver is implemented as a single class in Matlab named `WiMAX`. It consists of several files located in a directory called `@WiMAX`. The following section gives a general overview over the implementation and lists all aspects in which the code differs from the standard. Section B.2 describes the definition of the class and summarizes all defined methods and properties. The Matlab code of the class definition and a short example of its usage is also listed in this section. It contains declarations of properties and private methods, the class constructor and access methods `set` and `get` for each dependent property. Sections B.3 and B.4 explain the transmit and receive methods respectively, together with all private methods called from within them.

B.1. Features and Shortcomings

Primarily the implementation offers a transmit function that converts input data into a complex baseband WiMAX signal in time domain, and a receive function that inverts this process and returns the original data slots. The implementation closely follows the IEEE 802.16-2009 specification, as described in Section A.2. Nevertheless, several simplifications were made, mainly concerning details which are not required or meaningful for simulation purpose.

In particular, the simulation only features unidirectional communication links and a single subscriber station. Hence the construction of the WiMAX frame is greatly simplified, as there are no distinct DL and UL subframes and no data regions reserved for different users. What is more, the frame control header and the DL and UL MAP messages are unnecessary and omitted. Figure B.1 shows the frame structure as generated by the implementation. It consists of a frame preamble and a number of data bursts which can be configured by the user. A burst consists of all subcarriers by 3 OFDM symbols and can be thought of as a single user occupying

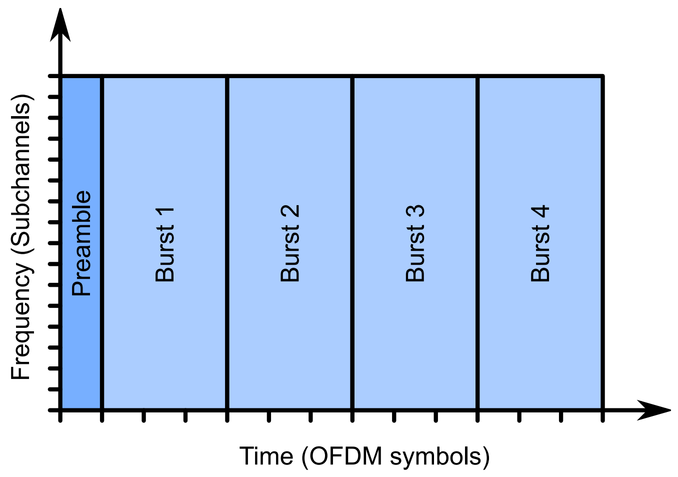


Figure B.1.: Simplified WiMAX frame structure as used for simulations.

all subchannels. The only implemented subcarrier permutation scheme is UL PUSC, which is the reason for grouping 3 consecutive OFDM symbols into one burst (cf. Section A.2.2).

Besides the frame structure, a few other simplifications and shortcomings exist, as listed below:

- As FEC scheme only the mandatory convolutional coding was implemented.
- Data randomization was not implemented.
- Tile permutation of UL PUSC was not implemented.
- The frame preamble does not contain the (rather complicated) data specified by the standard, but only a random sequence of BPSK modulated symbols.

Furthermore, for the simulations perfect frequency and timing synchronization was assumed and the according blocks are not contained in the receiver. Nonetheless, the receiver performs gain- and phase-synchronization, which is indispensable if the signal is sent through a non-trivial (i. e. pure AWGN) channels.

In certain kinds of simulations, especially when calculating bit error rates, it turned out that the fully featured implementation was too slow to yield proper results in reasonable time. Therefore an alternative set of functions for the transmitter and receiver were implemented which operate at higher performance. In these functions all code for subchannelization is removed and all subcarriers are simply filled up with the modulated data symbols. This also means that no pilot symbols are included in the transmitted signal, which increases the data rate by one third. The remaining code was rewritten in a more compact and efficient way (at the cost of readability) and features like interleaving and subcarrier randomization were removed. This significantly reduces processing time and memory consumption.

Moreover, an option to disable FEC was implemented. This can be done by setting the coding rate to 1. Besides reduced calculation time and increased data rate, disabling the FEC block is also beneficial in another way. Since the overall bit error rate is much higher without error correction, less data has to be processed to reach a certain number of errors, required for reliable BER curves. In total, comparing the fully featured implementation with the fast version, a speedup factor of about 20 is achieved. To switch the transceiver between normal and fast mode, a public property `mode` has been added to the class.

B.2. Class Definition and Usage Example

The file `WiMAX.m` contains the Matlab class definition. It consists of declarations of public and private properties, the constructor, property access methods (`set` and `get`) as well as the header lines of private methods. All methods of this class (public and private – except the constructor) are defined in separate files in the directory `@WiMAX`, but only those intended to be private need to be declared as such within the class definition.

Table B.1 lists all public properties of the class. Except `nrBurstsPerFrame` all of these are marked as *dependent*, which means that the access methods `set` and `get` are available and some processing and checking is possible upon reading and writing. Read-only properties are such for which only the `get` method was defined.

The methods of the class are listed in Table B.2. Besides the constructor, the only public methods are `Transmit` and `Receive`. These simply call the respective *normal* or *fast* implementation, depending on the current value of the `mode` property. The code for the transmitting and the receiving functions will be listed in the next two sections, together with the private methods which are called from within them.

The complete code contained in the file `WiMAX.m` is shown in Listing B.2. A simple example on how to use this class is given in Listing B.1.

Listing B.1: Example on how to use the class `WiMAX`

```
% ~~~~~~ WiMAX Transceiver Settings ~~~~~~
TRx = WiMAX();
TRx.mode = 'normal';       % 'normal' or 'fast'
TRx.BW = 5e6;              % Bandwidth (Hz)
TRx.fftSize = 128;         % 128, 512, 1024 or 2048
TRx.codeRate = 1/2;        % 1/2, 2/3 or 3/4; use 1 to disable FEC
TRx.QAMsize = 16;          % 4, 16 or 64
TRx.cpRatio = 1/8;         % 1/4, 1/8, 1/16 or 1/32
TRx.nrBurstsPerFrame = 5;  % 1 Burst = all subchan. x 3 time slots
scatterPlt = 'post-sync';  % 'none', 'pre-sync', 'post-sync' or '
    both'

% ~~~~~~~~ TRANSMITTER ~~~~~~~~
data_in = randint(TRx.inputDataSize(1), TRx.inputDataSize(2));
tx_signal = TRx.Transmit(data_in);

% ~~~~~~~~ CHANNEL ~~~~~~~~
rx_signal = awgn(tx_signal, 20); % Add AWGN with 20 dB SNR

% ~~~~~~~~ RECEIVER ~~~~~~~~
data_out = TRx.Receive(rx_signal, scatterPlt);
nr_errors = biterr(data_in, data_out);
```

Public Properties	
read/write:	
`nrBurstsPerFrame`	1 burst = 3 OFDM symbols
`mode`	`normal` or `fast`
`BW`	Channel bandwidth (Hz)
`fftSize`	128, 512, 1024 or 2048
`codeRate`	1/2, 2/3 or 3/4; 1 to switch FEC off
`QAMsize`	4, 16 or 64
`cpRatio`	1/4, 1/8, 1/16 or 1/32
read-only:	
`inputDataSize`	Size & nr. of input data slots
`Tb`	Useful symbol time (sec)
`Tg`	Symbol guard time (sec)
`Ts`	Total symbol time (sec)
`channelSamplePeriod`	Duration of a baseband sample (sec)
`timeSamplesPerFrame`	Nr. of samples returned from transmit method
`preamble`	Training seq. for equalizer (experimental)
`preambleSymbolTD`	Training seq. for equalizer (experimental)

Table B.1.: Public properties of the class `WiMAX`

Public Methods	
`WiMAX`	Constructor
`Transmit`	
`Receive`	
Private Methods	
Used by transmitter:	
`transmitNormal`	Fully featured transmitter
`transmitFast`	High performance transmitter
`convEncode`	FEC encoding
`create_UL_PUSC`	Subcarrier permutation
Used by receiver:	
`receiveNormal`	Fully featured receiver
`receiveFast`	High performance receiver
`convDecode`	FEC decoding
`extractSlots`	Subcarrier de-permutation
`synchronizeGainPhase`	Synchronization/Equalization
Used by both:	
`subcarrierRandomization`	

Table B.2.: Public and private methods of the class `WiMAX`

Listing B.2: WiMAX class definition (file `@WiMAX/WiMAX.m`)

```
classdef WiMAX < handle

%% Public Properties %%%%%%%%%%%%%%%
properties
  nrBurstsPerFrame;
end

properties (Dependent)
  mode                    % 'normal' or 'fast'
  BW
  fftSize
  codeRate
  QAMsize
  cpRatio
  % Read-only:
  inputDataSize
  Tb                      % useful symbol time
  Tg                      % symbol guard time (CP)
  Ts                      % total symbol time
  channelSamplePeriod
  timeSamplesPerFrame
  preamble
  preambleSymbolTD
end

%% Private Properties %%%%%%%%%%%%%%%
properties (Access = private)
  % General:
  mode_;
  % FEC:
  codeRate_;
  puncturePattern_;
  trellis_ = poly2trellis(7, [171 133]);
  % Interleaving:
  interleaverTable_;
  % Modulation:
  QAMsize_;
  hQAMmod_;
  hQAMdemod_;
  constellationMapping_;
  modScale_;
  % Time-/Frequency-Plane:
  fftSize_;
  nrUsedSubCarriers_;
  nrSubchannels_;
  guardCarriersLeft_;
  guardCarriersRight_;
  DCcarrierIndex_;
  preamble_;
  % Timing:
    cpRatio_;
    BW_
    FS_;
    % for subcarrierRandomization:
    IDcell = 5;
    FrameNumber = 5;
    % scatter plot:
    hScatterPre_ = 0;   % before synchronization
    hScatterPost_ = 0;  % after synchronization
  end

  properties (Dependent, Access = private)
    totNrOFDMsymbolsPerFrame_;
    nrDataOFDMsymbolsPerFrame_;
  end

  %% Public Methods %%%%%%%%%%%%%%%
  methods

    %% Constructor %%%%%%%%%%%%%%%
    function obj = WiMAX()
      obj = obj@handle();         % call superclass constructor

      obj.mode = 'normal';
      obj.BW = 5e6;
      obj.fftSize = 128;        % allowed: 128, 512, 1024, 2048
      obj.QAMsize = 16;         % allowed: 4, 16 or 64
      obj.codeRate = 1/2;       % allowed: 1/2, 2/3 or 3/4
      obj.cpRatio = 1/8;        % allowed: 1/4, 1/8, 1/16, 1/32
      obj.nrBurstsPerFrame = 3; % 1 Burst = all subch. x 3 time slots

      % FIXME Preample is not compliant with standard:
      obj.preamble_ = 1-2*randint(obj.nrUsedSubCarriers_+1, 1);
      obj.preamble_(obj.nrUsedSubCarriers_/2+1) = 0; % DC carrier
    end


    %% Property Access %%%%%%%%%%%%%%%
    % General
    function md = get.mode(obj)
      md = obj.mode_;
    end
    function set.mode(obj, md)
      if ~(isequal(md, 'normal') || isequal(md, 'fast'))
        error('Illegal mode. Must be ''normal'' or ''fast''.');
      end
      obj.mode_ = md;
    end

    % Bandwidth
    function bw = get.BW(obj)
      bw = obj.BW_;
    end
    function set.BW(obj, bw)
      obj.BW_ = bw;
      if mod(bw, 1.75e6) == 0
          samplingFactor = 8/7;
      elseif (mod(bw, 1.25e6) == 0) || ...
             (mod(bw, 1.5e6 ) == 0) || ...
             (mod(bw, 2.75e6) == 0)
          samplingFactor = 28/25;
      else
          samplingFactor = 8/7;
      end
      obj.FS_ = floor (samplingFactor * bw / 8000) * 8000;
    end

    % FFT Size
    function fft_size = get.fftSize(obj)
      fft_size = obj.fftSize_;
    end
    function set.fftSize(obj, fft_size)
      obj.fftSize_ = fft_size;
      switch fft_size
        % Values refer to UL PUSC scheme, cf. Tables 455 - 458
        case 2048
          obj.nrUsedSubCarriers_ = 1680;  %=fftSize-guardLeft-
              guardRight-DC
          obj.nrSubchannels_ = 70;
          obj.guardCarriersLeft_ = 184;
          obj.guardCarriersRight_ = 183;
          obj.DCcarrierIndex_ = 1024;
        case 1024
          obj.nrUsedSubCarriers_ = 840;
          obj.nrSubchannels_ = 35;
          obj.guardCarriersLeft_ = 92;
          obj.guardCarriersRight_ = 91;
          obj.DCcarrierIndex_ = 512;
        case 512
          obj.nrUsedSubCarriers_ = 408;
          obj.nrSubchannels_ = 17;
          obj.guardCarriersLeft_ = 52;
          obj.guardCarriersRight_ = 51;
          obj.DCcarrierIndex_ = 256;
        case 128
          obj.nrUsedSubCarriers_ = 96;
          obj.nrSubchannels_ = 4;
          obj.guardCarriersLeft_ = 16;
          obj.guardCarriersRight_ = 15;
          obj.DCcarrierIndex_ = 64;
        otherwise
          error('Illegal FFT size. Only 2048, 1024, 512 or 128 are
              allowed.');
      end

      % FIXME Preample is not compliant with standard:
      obj.preamble_ = 1-2*randint(obj.nrUsedSubCarriers_+1, 1);
      obj.preamble_(obj.nrUsedSubCarriers_/2+1) = 0; % DC carrier
    end

    % codeRate
    function cr = get.codeRate(obj)
      cr = obj.codeRate_;
    end
    function set.codeRate(obj, cr)
      obj.codeRate_ = cr;
      switch cr
        case 1
          % Switch FEC off (not WiMAX compliant)
          obj.puncturePattern_ = [];
        case 1/2
          obj.puncturePattern_ = [1; 1];
        case 2/3
          obj.puncturePattern_ = [1; 0; 1; 1];
        case 3/4
          obj.puncturePattern_ = [1; 0; 1; 1; 1; 0];
        otherwise
          error('Illegal code rate. Only 1/2, 2/3, 3/4 or 1 are
              allowed.');
      end
    end

    % QAMsize
    function qam_size = get.QAMsize(obj)
      qam_size = obj.QAMsize_;
    end
    function set.QAMsize(obj, qam_size)
      % Initialize Modulator & Demodulator
      obj.QAMsize_ = qam_size;
      switch qam_size
        case 4
          obj.constellationMapping_ = [2 3 0 1];
          obj.modScale_ = 1 / sqrt(2);
        case 16
          obj.constellationMapping_ = [13 12 14 15 9 8 10 11 1 0 2 3
              5 4 6 7];
          obj.modScale_ = 1 / sqrt(10);
        case 64
          temp = [3 2 0 1 5 4 6 7];
          obj.constellationMapping_ = [7*8+temp 6*8+temp 4*8+temp ...
                     5*8+temp 1*8+temp 0*8+temp 2*8+temp 3*8+temp];
          obj.modScale_ = 1 / sqrt(42);
        otherwise
          error('Illegal value for constellation size. Only 4, 16 or
              64 are allowed values.');
      end
      h = modem.qammod('M', qam_size, 'SymbolOrder', ...
        'user-defined', 'SymbolMapping', ...
        obj.constellationMapping_, 'InputType', 'Bit');
      obj.hQAMmod_ = h.copy;
      obj.hQAMdemod_ = modem.qamdemod(h);

      % Initialize Interleaver (cf. section 8.4.9.3, pdf-page 1139)
      Ncpc = log2(qam_size);
      Ncbps = 48 * Ncpc;
      d = 16;
      s = Ncpc / 2;
      k = 0:Ncbps-1;
      mk = (Ncbps/d)*mod(k,d)+floor(k/d);                        % Eq. 121
      jk = s*floor(mk/s)+mod(mk+Ncbps-floor(d*mk/Ncbps),s); % Eq. 122
      [x, obj.interleaverTable_] = sort(jk);

    end

    % Cyclic Prefix Ratio
    function cpr = get.cpRatio(obj)
      cpr = obj.cpRatio_;
    end
    function set.cpRatio(obj, cpr)
      obj.cpRatio_ = cpr;
    end

    % Miscellaneous dependent properties
    function tb = get.Tb(obj)
      tb = obj.fftSize / obj.FS_;
    end
    function tg = get.Tg(obj)
      tg = obj.Tb * obj.cpRatio;
    end
    function ts = get.Ts(obj)
      ts = obj.Tb + obj.Tg;
    end
    function csp = get.channelSamplePeriod(obj)
      csp = 1 / obj.FS_;
      %   = obj.Ts / (obj.fftSize * (1+obj.cpRatio));
    end
    function spf = get.timeSamplesPerFrame(obj)
      spf = obj.totNrOFDMsymbolsPerFrame_*obj.fftSize*(1+obj.cpRatio)
          ;
    end
    function sz = get.inputDataSize(obj)
      if isequal(obj.mode_, 'normal')
        sz = [48 * obj.codeRate * log2(obj.QAMsize) ...
              obj.nrBurstsPerFrame * obj.nrSubchannels_];
      else  % fast mode
        sz=[obj.nrUsedSubCarriers_*obj.codeRate*log2(obj.QAMsize) ...
            obj.nrBurstsPerFrame * 3];
      end
    end

    function nr = get.totNrOFDMsymbolsPerFrame_(obj)
      nr = obj.nrBurstsPerFrame * 3 + 1;
    end
    function nr = get.nrDataOFDMsymbolsPerFrame_(obj)
      nr = obj.nrBurstsPerFrame * 3;
    end
    function pr = get.preamble(obj)
      pr = [zeros(obj.guardCarriersLeft_, 1);
            obj.preamble_;
            zeros(obj.guardCarriersRight_, 1)];
    end
    function pr = get.preambleSymbolTD(obj)
      prFD = obj.preamble;
      pr = ifft(ifftshift(prFD));
      cp_ind = obj.fftSize_*(1-obj.cpRatio_)+1 : obj.fftSize_;
      scFact = obj.fftSize_ / sqrt(obj.nrUsedSubCarriers_);
      pr = [pr(cp_ind); pr] * scFact;
    end

  end

  %% Private Methods %%%%%%%%%%%%%%%%
  methods (Access = private)
    % Transmitter:
    tx_signal = transmitNormal(o, data_in)
    tx_signal = transmitFast(o, data_in)
    slots_out = convEncode(o, slots_in)
    frame_out = create_UL_PUSC(o, slots_in)
    % Receiver:
    out_data  = receiveNormal(o, rx_signal)
    out_data  = receiveFast(o, rx_signal)
    slots_out = convDecode(o, slots_in)
    syncd_frame = synchronizeGainPhase(o, data_frame, rx_preamble)
    slots_out = extractSlots(o, frame_in)
    % Both:
    rndMatrix = subcarrierRandomization(o)
  end

  end   % end classdef
```

B.3. Transmitter Related Methods

The public method `Transmit` reads the value of the property `mode` and calls one of the private methods `transmitNormal` or `transmitFast` accordingly. As was explained in Section B.1, the normal mode method contains the fully featured transmitter, closely following the IEEE802.16j-2009 standard, while the fast mode method can be used for long and computationally intensive simulations.

In normal mode, the input to the transmit function should be formatted in the same way like the MAC layer would pass data on to the PHY layer, that is in the form of slots containing $48 \cdot M \cdot R$ bits, where M is the modulation order and R is the coding rate (cf. Subsection A.2.1). Hence, the input parameter to `Transmit` must be a matrix of binary values, having a dimension of

$$Number\ of\ uncoded\ bits\ per\ slot \times Number\ of\ slots\ that\ fit\ in\ a\ frame.$$

This dimension is available to the user through the `inputDataSize` property. In fast mode, the dimensions of the input data matrix are

$$Number\ of\ uncoded\ bits\ per\ OFDM\ symbol \times Number\ of\ OFDM\ symbols\ in\ the\ frame.$$

The `inputDataSize` property is adjusted automatically. Using this input, the transmit function calculates and returns a WiMAX frame as a complex baseband signal in time domain. The remainder of this section contains the code listings of all methods used by the transmit functions. An overview over the listings is given in Table B.3.

Method Name	**Reference**
`Transmit`	Listing B.3
`transmitNormal`	Listing B.4
`transmitFast`	Listing B.5
`convEncode`	Listing B.6
`create_UL_PUSC`	Listing B.7
`subcarrierRandomization`	Listing B.8

Table B.3.: References to code listings of methods associated with transmitting.

Listing B.3: Public method `Transmit` (file `@WiMAX/Transmit.m`)

```
function tx_signal = Transmit(o, data_in)

if ~isequal(size(data_in), o.inputDataSize)
  error(['Wrong size of input data to method ''Transmit''.' ...
         ' Check property ''inputDataSize''.']);
end

switch o.mode
  case 'normal'
    tx_signal = o.transmitNormal(data_in);
  case 'fast'
```

```
    tx_signal = o.transmitFast(data_in);
end

end
```

Listing B.4: Private Method `transmitNormal` (f. `@WiMAX/transmitNormal.m`)

```
function tx_signal = transmitNormal(o, data_in)

% FIXME: Randomization is missing here

% FEC encoce:
if o.codeRate_ == 1
  encoded_slots = data_in;  % Disable FEC
else
  encoded_slots = o.convEncode(data_in);
end

% Interleave:
intlved_slots = intrlv(encoded_slots, o.interleaverTable_);

% Modulate:
modulated_slots = ...
  o.modScale_ * modulate(o.hQAMmod_, intlved_slots);

% Arrange slots in UL PUSC format and assamble frame:
frame = o.create_UL_PUSC(modulated_slots);

% Add DC carrier:
frame = [frame(1 : o.nrUsedSubCarriers_/2, :);
         zeros(1, o.nrDataOFDMsymbolsPerFrame_);
         frame(o.nrUsedSubCarriers_/2+1 : end, :)];

% Subcarrier randomization:
frame = frame .* o.subcarrierRandomization();

% Add preamble:
frame = [o.preamble_, frame];

% Add guard carriers
frame_fd = ...
  [zeros(o.guardCarriersLeft_, o.totNrOFDMsymbolsPerFrame_); ...
   frame; ...
   zeros(o.guardCarriersRight_, o.totNrOFDMsymbolsPerFrame_)];

% IFFT
frame_td = ifft(ifftshift(frame_fd, 1));

% Add cyclic prefix
cyclicPrefix = frame_td(o.fftSize*(1-o.cpRatio)+1 : o.fftSize, :);
frame_td = [cyclicPrefix; frame_td];
```

```

% Normalize power and send signal
tx_signal = frame_td(:).' * o.fftSize_ /sqrt(o.nrUsedSubCarriers_);

end
```

Listing B.5: Private Method `transmitFast` (file `@WiMAX/transmitFast.m`)

```
function tx_signal = transmitFast(o, data_in)

% FEC encoce:
if o.codeRate_ == 1
  encoded_data = data_in;
else
  encoded_data = o.convEncode(data_in);
end

% Modulate:
modulated_data = ...
  o.modScale_ * modulate(o.hQAMmod_, encoded_data);

% Input packing
frame = zeros(o.fftSize_, o.totNrOFDMsymbolsPerFrame_);
frame(o.guardCarriersLeft_+1 : o.DCcarrierIndex_, 2:end) = ...
  modulated_data(1:o.nrUsedSubCarriers_/2, :);
frame(o.DCcarrierIndex_+2 : end-o.guardCarriersRight_, 2:end) = ...
  modulated_data(o.nrUsedSubCarriers_/2+1 : end, :);
frame(o.guardCarriersLeft_+1 : end-o.guardCarriersRight_, 1) = ...
  o.preamble_;

% IFFT
frame_td = ifft(ifftshift(frame, 1));

% Add cyclic prefix
cyclicPrefix = frame_td(o.fftSize*(1-o.cpRatio)+1 : o.fftSize, :);
frame_td = [cyclicPrefix; frame_td];

% Normalize power and send signal
tx_signal = frame_td(:).' * o.fftSize_ /sqrt(o.nrUsedSubCarriers_);

end
```

Listing B.6: Private Method `convEncode` (file `@WiMAX/convEncode.m`)

```
function slots_out = convEncode(o, slots_in)
  % Punctured tail-biting convolutional encoding as defined
  % in section 8.4.9.2.1 of the IEEE 802.16-2009 standard

  nr_in_bits = size(slots_in, 1);
  nr_slots = size(slots_in, 2);

  slots_out = zeros(nr_in_bits/o.codeRate, nr_slots);
  for i = 1:nr_slots
    slot = slots_in(:,i);
    initState = bi2de(slot(end:-1:end-5)', 'left-msb');
    slots_out(:,i) = ...
      convenc(slot, o.trellis_, o.puncturePattern_, initState);
  end

end
```

Listing B.7: Private Method `create_UL_PUSC` (f. `@WiMAX/create_UL_PUSC.m`)

```
function frame_out = create_UL_PUSC(o, slots_in)
  % UL PUSC subcarrier permutation as defined in
  % section 8.4.6.2 of the IEEE 802.16-2009 standard

  tiles = ones(4,3,6);    % 6 tiles
  frame_out = zeros(o.nrUsedSubCarriers_, o.
      nrDataOFDMsymbolsPerFrame_);
  for burst_nr=0:o.nrBurstsPerFrame-1
    for slot_nr=0:o.nrSubchannels_-1
      % Construct 24x3 subchannel from 48x1 slot
      tot_slot_nr = burst_nr*o.nrSubchannels_+slot_nr+1;
      slot = slots_in(:, tot_slot_nr);
      for tile_nr = 0:5
        tiles(2:3,1,tile_nr+1)=slot([tile_nr*2    tile_nr*2+1]+1);
        tiles(1:4,2,tile_nr+1)=slot((tile_nr*4+12:tile_nr*4+15)+1);
        tiles(2:3,3,tile_nr+1)=slot([tile_nr*2+36 tile_nr*2+37]+1);

        % FIXME: here should be a permutation as defined in
        % section 8.4.6.2.2 of the IEEE 802.16-2009 standard.
      end
      PUSC_slot = [tiles(:,:,1); tiles(:,:,2); tiles(:,:,3);
                   tiles(:,:,4); tiles(:,:,5); tiles(:,:,6)];

      % Put the 24x3 slot into frame
      frame_out(24*slot_nr+(1:24), 3*burst_nr+(1:3)) = PUSC_slot;
    end
  end
end
```

Listing B.8: Private Method `subcarrierRandomization`

```
function rndMatrix = subcarrierRandomization(o)
  % Confer section 8.4.9.4.1 of the IEEE 802.16-2009 standard
  % (pdf page 1062)

  initVector = ...
```

```
    ([ sscanf(dec2bin(o.IDcell,5), '%1i%1i%1i%1i%1i');
      1; 1;
      sscanf(dec2bin(o.FrameNumber,4), '%1i%1i%1i%1i')])';
  lfsr = commsrc.pn( ...
      'GenPoly', [1 0 0 0 0 0 0 0 0 1 0 1], ...
      'InitialStates', initVector, ...
      'NumBitsOut', 1);
  rndMatrix = zeros(o.nrUsedSubCarriers_+1, ...
    o.nrDataOFDMsymbolsPerFrame_);
  lfsr.NumBitsOut = o.nrUsedSubCarriers_+1;
  rndMatrix(:,1) = 1-2*lfsr.generate;

  lfsr.NumBitsOut = 1;
  for symbolNr = 2 : o.nrDataOFDMsymbolsPerFrame_
    rndMatrix(1:end-1, symbolNr) = ...
      rndMatrix(2:end, symbolNr-1);
    rndMatrix(end, symbolNr) = 1-2*lfsr.generate;
  end
end
```

B.4. Receiver Related Methods

The receiver inverts the processing of the transmitter, recovering the contained data bits from a WiMAX baseband signal. The methods are implemented in a similar way to the transmitter methods, i. e. a public method `Receive` which, depending on the mode, calls either `receiveNormal` or `receiveFast`. However, the public method accepts an additional parameter `scatter_plot`, via which the receiver can be instructed to generate a scatter plot of the received symbols before and/or after synchronization.

The synchronization block is implemented in the method `synchronizeGainPhase`. It removes rotations of the symbol constellation and normalizes the signal power, using the known frame preamble as a training sequence. The calculation is done in frequency domain, i. e. after FFT. In a first step the transfer function of the channel $C(k)$ is estimated,

$$C(k) = \frac{S_{rx}(k)}{S_{train}(k)}, \tag{B.1}$$

where $S_{rx}(k)$ denotes the received preamble and $S_{train}(k)$ is the known transmitted preamble. The index k denotes to the subcarrier number. In the second step a correction vector $V(k)$ is calculated, which is subsequently multiplied with each OFDM symbol of the frame. The * operator refers to complex conjugation.

$$V(k) = \frac{C^*(k)}{C^2(k)} \tag{B.2}$$

Table B.4 lists all methods and references to their code listings given in this section.

Method Name	**Reference**
`Receive`	Listing B.9
`receiveNormal`	Listing B.10
`receiveFast`	Listing B.11
`convDecode`	Listing B.12
`extractSlots`	Listing B.13
`synchronizeGainPhase`	Listing B.14

Table B.4.: References to code listings of methods associated with receiving.

Listing B.9: Public method `Receive` (file `@WiMAX/Receive.m`)

```
function out_data = Receive(o, rx_signal, scatter_plot)

  % Prepare scatter plots
  if nargin < 3, scatter_plot = 'none'; end
  switch lower(scatter_plot)
    case 'none'
        o.hScatterPre_ = 0;
        o.hScatterPost_ = 0;
    case 'pre-sync'
```

```
      if o.hScatterPre_ == 0
        o.hScatterPre_ = commscope.ScatterPlot;
        o.hScatterPre_.SamplesPerSymbol = 1;
      end
      o.hScatterPost_ = 0;
    case 'post-sync'
      if o.hScatterPost_ == 0
        o.hScatterPost_ = commscope.ScatterPlot;
        o.hScatterPost_.SamplesPerSymbol = 1;
      end
      o.hScatterPre_ = 0;
    case 'both'
      if o.hScatterPre_ == 0
        o.hScatterPre_ = commscope.ScatterPlot;
        o.hScatterPre_.SamplesPerSymbol = 1;
      end
      if o.hScatterPost_ == 0
        o.hScatterPost_ = commscope.ScatterPlot;
        o.hScatterPost_.SamplesPerSymbol = 1;
      end
    otherwise
      error(['Illegal value for scatter_plot: only ''none'', ' ...
        '''pre-sync'', ''post-sync'' or ''both'' allowed!']);
  end

  % Call receive method
  switch o.mode
    case 'normal'
      out_data = o.receiveNormal(rx_signal);
    case 'fast'
      out_data = o.receiveFast(rx_signal);
  end

end
```

Listing B.10: Private Method `receiveNormal` (file `@WiMAX/receiveNormal.m`)

```
function out_data = receiveNormal(o, rx_signal)

% Put received samples into a matrix and remove cyclic prefix:
samplesPerSymbol = o.fftSize * (1+o.cpRatio);
samplesPerCP = o.fftSize * o.cpRatio;
frame_td = zeros(o.fftSize, o.totNrOFDMsymbolsPerFrame_);
for i=0:o.nrDataOFDMsymbolsPerFrame_
  frame_td(:,i+1) = rx_signal(i * samplesPerSymbol + ...
    samplesPerCP + (1:o.fftSize));
end

% FFT
frame_fd = fftshift(fft(frame_td), 1);

```

```
% Remove guard carriers
frame_fd = frame_fd(o.guardCarriersLeft_+1 : ...
             o.fftSize-o.guardCarriersRight_, :);

% Remove preamble:
preamble = frame_fd(:, 1);
frame = frame_fd(:, 2:end);

% Gain & Phase Synchronization
if o.hScatterPre_ ~= 0
  update(o.hScatterPre_, frame);
  autoscale(o.hScatterPre_);
end
sync_frame = o.synchronizeGainPhase(frame, preamble);
if o.hScatterPost_ ~= 0
  update(o.hScatterPost_, sync_frame);
end

% Subcarrier de-randomization:
frame = sync_frame .* o.subcarrierRandomization();

% Remove DC carrier:
frame = [frame(1 : o.nrUsedSubCarriers_/2     , :);
         frame(o.nrUsedSubCarriers_/2+2 : end, :)];

% Extract UL PUSC slots from received frame:
rx_slots = o.extractSlots(frame);

% Demodulation:
demodulated_slots = demodulate(o.hQAMdemod_, rx_slots/o.modScale_);

% De-Interleaving:
deintlved_slots = deintrlv(demodulated_slots, o.interleaverTable_);

% Decoding:
if o.codeRate_ == 1
  out_data = deintlved_slots;
else
  out_data = o.convDecode(deintlved_slots);
end

% FIXME: Randomization is missing here

end
```

Listing B.11: Private Method `receiveFast` (file `@WiMAX/receiveFast.m`)

```
function out_data = receiveFast(o, rx_signal)

% Put received samples into a matrix and remove cyclic prefix:
samplesPerSymbol = o.fftSize * (1+o.cpRatio);
```

```
samplesPerCP = o.fftSize * o.cpRatio;
frame_td = zeros(o.fftSize, o.totNrOFDMsymbolsPerFrame_);
for i=0:o.nrDataOFDMsymbolsPerFrame_
  frame_td(:,i+1) = rx_signal(i * samplesPerSymbol + ...
    samplesPerCP + (1:o.fftSize));
end

% FFT
frame_fd = fftshift(fft(frame_td), 1);

% Remove guard carriers and preamble
preamble = frame_fd(o.guardCarriersLeft_+1 : ...
  end-o.guardCarriersRight_, 1);
frame    = frame_fd(o.guardCarriersLeft_+1 : ...
  end-o.guardCarriersRight_, 2:end);

% Gain & Phase Synchronization
if o.hScatterPre_ ~= 0
  update(o.hScatterPre_, frame);
  autoscale(o.hScatterPre_);
end
sync_frame = o.synchronizeGainPhase(frame, preamble);
if o.hScatterPost_ ~= 0
  update(o.hScatterPost_, sync_frame);
end

% Remove DC carrier:
frame = [sync_frame(1 : o.nrUsedSubCarriers_/2     , :);
         sync_frame(o.nrUsedSubCarriers_/2+2 : end, :)];

% Demodulation:
demodulated_data = demodulate(o.hQAMdemod_, frame/o.modScale_);

% Decoding:
if o.codeRate_ == 1
  out_data = demodulated_data;
else
  out_data = o.convDecode(demodulated_data);
end

end
```

Listing B.12: Private Method `convDecode` (file `@WiMAX/convDecode.m`)

```
function slots_out = convDecode(o, slots_in)
  % Punctured tail-biting convolutional decoding as defined
  % in section 8.4.9.2.1 of the IEEE 802.16-2009 standard

  nr_in_bits = size(slots_in,1);
  nr_slots = size(slots_in,2);

  nr_uncoded_bits = nr_in_bits*o.codeRate;
  slots_out = zeros(nr_uncoded_bits, nr_slots);
  slots_in = [slots_in; slots_in];
  sel_bits = [nr_uncoded_bits+1:(3*nr_uncoded_bits/2) ...
               (nr_uncoded_bits/2+1):nr_uncoded_bits];
  for j = 1:nr_slots
    slot = vitdec(slots_in(:,j), o.trellis_, 34, ...
      'trunc', 'hard', o.puncturePattern_);
    slots_out(:,j) = slot(sel_bits);
  end

end
```

Listing B.13: Private Method `extractSlots` (file `@WiMAX/extractSlots.m`)

```
function slots_out = extractSlots(o, frame_in)
  % UL PUSC subcarrier de-permutation as defined in
  % section 8.4.6.2 of the IEEE 802.16-2009 standard

  slots_out = zeros(48, o.nrBurstsPerFrame * o.nrSubchannels_);
  for burst_nr=0:o.nrBurstsPerFrame-1
    for slot_nr=0:o.nrSubchannels_-1
      PUSC_slot = frame_in(24*slot_nr+(1:24), 3*burst_nr+(1:3));
      abs_slot_nr = burst_nr*o.nrSubchannels_+slot_nr+1;
      slot = [PUSC_slot([2 3 6 7 10 11 14 15 18 19 22 23], 1);
              PUSC_slot(1:24, 2);
              PUSC_slot([2 3 6 7 10 11 14 15 18 19 22 23], 3)];
      slots_out(:, abs_slot_nr) = slot;

      % Pilots currently unused
      % pilots = [PUSC_slot([1 4 5 8 9 12 13 16 17 20 21 24],1);
      %           PUSC_slot([1 4 5 8 9 12 13 16 17 20 21 24],3)];

      % FIXME: here should be a permutation as defined in
      % section 8.4.6.2.2 of the IEEE 802.16-2009 standard.

    end
  end
end
```

Listing B.14: Private Method `synchronizeGainPhase`

```
function syncd_frame = synchronizeGainPhase(o, data_frame,
   rx_preamble)

  corr_vector = rx_preamble ./ o.preamble_;
  corr_vector = conj(corr_vector) ./ (abs(corr_vector).^2);
  corr_matrix = repmat(corr_vector, ...
    1, o.nrDataOFDMsymbolsPerFrame_);
    syncd_frame = data_frame .* corr_matrix;

end
```

Appendix C

Matlab Implementation of the OCR

Implementing an on-channel repeater with a feedback interference cancellation system for simulation purpose turns out to be a significant amount of code with numerous options and parameters. In order to provide a flexible tool that is easy to use and modify, it was decided to implement the OCR as an object-oriented class in Matlab, called `OnChannelRepeater`. Some of the features of this implementation are:

- Simple configuration through versatile properties (e. g. the delay time can be set or read in seconds as well as in samples).
- Different levels of operation (the repeater can be run without FIC and even without feedback channel).
- Instability detection.
- A multipath feedback channel.
- Methods to record the learning behaviour of the FIC.
- An overloaded `disp` method, giving a clearly arranged overview over the OCR setup.
- Experimental functionalities for equalization.

Experimentally, an equalizer has been implemented which attempts to estimate and cancel the channel influence on the receive side, by using the known frame preamble as a training sequence. However, simulations did not show a significant performance gain, and some more research is required. All code concerning the equalizer is listed in this appendix, but not further explained.

Section C.1 explains the details and design choices of the implementation. The source code listings are provided in Section C.2.

C.1. Implementation Details

All files belonging to the class, are located in a directory `@OnChannelRepeater`. The class definition, containing all property declarations, access methods (`set` and `get`), the constructor and several short public methods, can be found in the file `OnChannelRepeater.m`.

Public Properties	
General:	
`SamplePeriod`	Period of one sample (read-only)
Forward path:	
`Gain`	Repeater gain (linear scale)
`Gain_dB`	Repeater gain (in dB)
`RepDelay`	Delay through OCR (in seconds)
`RepDelay_smpl`	Delay through OCR (in samples)
Feedback channel multipath components (MPCs):	
`FB_DelayGain`	Gain and delay values of the MPCs (write-only)
`FB_Att_dB`	Gain of the defined MPCs in dB (read-only)
`FB_Delay`	Delay of the defined MPCs in seconds (read-only)
`FB_Delay_smpl`	Delay of the defined MPCs in samples (read-only)
`FB_ImpulseResp`	FB channel impulse response (read-only)
`FB_Noise_dB`	Power of AWGN on the FB channel in dB
FIC:	
`AdaptFiltSize`	Number of weights of the FIC filter
`StepWidth`	LMS step size μ
`Filt_coeff`	Current weight values of the FIC filter
`Stability_limit`	Max. output value before OCR is regarded instable
Equalizer (experimental):	
`EQU_mu`	
`EQU_nrTaps`	
`EQU_weights`	

Table C.1.: Public properties of the class `OnChannelRepeater`

Table C.1 lists all public properties of the class, functionally grouped. The sample period is the only parameter which has to be specified during instantiation of an object (as an obligatory argument to the constructor) and cannot be changed later on. On the forward path, the repeater delay can be specified either in seconds or in samples, using the sample rate as conversion factor. Similarly, the gain of the OCR can be set or read either in linear or logarithmic (dB) scale.

The feedback channel allows emulating a simple form of multipath propagation. An initial idea was to use a sophisticated fading channel object from the Matlab Communication Toolbox and filter the transmit signal with this object to generate the feedback signal. However, the closed-loop nature of the OCR implicates that the input to the filter depends on its earlier output. Thus the filtering operation can only be done one sample at a time, which dramatically slows the calculation down. Tests have shown that using the fading channel object is about 300 times slower than direct convolution of the channel impulse response with the buffered transmit signal.

Therefore the feedback channel is implemented as a simple time-invariant FIR filter, with each tap representing one multipath component (MPC). Its coefficients can be specified through the property `FB_DelayGain`, which is a matrix of two rows containing the delay in seconds

Public Methods	
`OnChannelRepeater`	Constructor
`run`	Send a signal through the repeater
`run_rec`	Records error values during execution
`reset`	Reset filter weights to zero
`train_filter`	Train the FIC to the FB channel
`disp`	Print nicely formatted OCR parameters
`disp_filt_coeff`	Print current weights of FIC filter
Experimental methods for equalizer:	
`train_equalizer`	
`disp_equalizer_weights`	
`run_Equ`	
Private Methods	
`run_LMS`	Run repeater with FIC switched on
`run_noFIC`	Run repeater with FIC switched off
`run_noFB`	Run repeater without FB channel

Table C.2.: Public and private methods of the class `OnChannelRepeater`

and the gain (or rather attenuation) in dB of each MPC. The delay is rounded to the next integer multiple of the sample period. If the difference between two given delay values is less than the sample period, an error is raised. All samples of the channel impulse response which are not specified as a MPC are set to zero. To calculate one sample of the feedback signal, the vector containing the channel impulse response is transposed and multiplied with the buffered transmit signal vector. Besides the multipath propagation, it is also possible to add white Gaussian noise to the feedback signal. The noise power is specified in dB via the parameter `FB_Noise_dB`.

The main parameters concerning the FIC system are the adaptive filter size and the step size of the LMS algorithm. The current values of adaptive filter coefficients can be read from the property `Filt_coeff`. This property is also writable, to permit different initial conditions. During execution, the implementation constantly checks if the repeater gets instable. This is done by testing if the absolute value of the transmit signal exceeds a certain limit, given by the property `Stability_limit`. If so, the calculation is aborted and the `run` method returns with the parameter `stable` set to `false`. By default, the stability limit is set arbitrarily to 10^8. If it is changed, it should be set to not less than about 50 times the standard deviation of the transmit signal, in order to prevent a false positive classification.

Table C.2 shows the methods defined by the class. To send a signal through the repeater, the method `run` has to be invoked. It takes the input signal and a string specifying the operational mode as parameters and returns the retransmitted signal and a variable indicating if the OCR was stable during calculation. The operational mode can be `LMS` (for operation with FIC switched on), `noFIC` (FIC switched off) or `noFB` (without FB channel) and causes `run` to call the according private method. The method `run_rec` provides the same functionality as the LMS-mode, but additionally records error and deviation values as specified in Section 4.4. It was used to calculate the learning curves presented there.

Further methods exist to reset and train the adaptive filter and to display the status of the OCR. An example on how to use the class is given in Listing C.1. It makes use of the function `createSUIchannel`, which returns a Rician fading channel object with parameters taken from the Stanford University Interim (SUI) model [Erceg et al., 2001]. This function is not part of the OCR class and its code is not listed here. Listing C.2 shows the command line output generated by the `disp` function in Listing C.1.

Listing C.1: Example on how to use the class `OnChannelRepeater`

```
% ~~~~~~ Transceiver Settings ~~~~~~
TRx = WiMAX(); % Create WiMAX object with default settings

% ~~~~~~ Repeater Settings ~~~~~~
RPx = OnChannelRepeater(TRx.channelSamplePeriod);
RPx.Gain_dB        = 30;
RPx.RepDelay       = 3e-6;
RPx.AdaptFiltSize = 30;
RPx.StepWidth      = 1 / (RPx.AdaptFiltSize* RPx.Gain^2 * 100);
RPx.FB_Noise_dB    = -40;
RPx.FB_DelayGain   = [300e-9, 1.2e-6, 5e-6;  % sec
                          -35,    -42,   -45]; % dB
sui_model = 2;
channelAnoise = 30; % SNR
channelBnoise = 30; % SNR
channelA = CreateSUIchannel(sui_model, TRx.channelSamplePeriod);
channelB = CreateSUIchannel(sui_model, TRx.channelSamplePeriod);

disp(RPx);
RPx.train_filter(10000); % Train filter with 10000 random samples

% ~~~~~~~~ TRANSMITTER ~~~~~~~~
data_in = randint(TRx.inputDataSize(1), TRx.inputDataSize(2));
tx_signal = TRx.Transmit(data_in);

% ~~~~~~~~ REPEATER ~~~~~~~~
rx_signal = awgn(filter(channelA, tx_signal), channelAnoise);
[tx_signal, stable] = RPx.run(rx_signal , 'LMS');
if ~stable
  error('Repeater unstable!');
end

% ~~~~~~~~ RECEIVER ~~~~~~~~
rx_signal = awgn(filter(channelB, tx_signal), ...
  channelBnoise, RPx.Gain_dB);
data_out = TRx.Receive(rx_signal, 'post-sync');
nr_errors = biterr(data_in, data_out);
nr_bits_sent = TRx.inputDataSize(1) * TRx.inputDataSize(2);
ber = nr_errors/nr_bits_sent;
```

Listing C.2: Output of the `disp` function of Listing C.1

```
Repeater Gain:    30.0 dB (31.62)
Filter Size:      30
LMS Step Width:   3.333e−07
Repeater Delay:   3.036 us (17 samples)
Sample Period:    0.1786 us
Feedback Channel:
  Path 1:   0.3571 us    2 samples    −35 dB (0.01778)
  Path 2:     1.25 us    7 samples    −42 dB (0.007943)
  Path 3:        5 us   28 samples    −45 dB (0.005623)
```

C.2. Code Listings of the OCR Implementation

This section lists the source codes of the class `OnChannelRepeater`, including the class definition and all methods. Table C.3 gives an overview and references to all listings.

Method Name	Reference
Class definition	Listing C.3
`run`	Listing C.4
`run_LMS`	Listing C.5
`run_noFIC`	Listing C.6
`run_noFB`	Listing C.7
`run_rec`	Listing C.8
`run_Equ`	Listing C.9

Table C.3.: References to code listings of the class `OnChannelRepeater`.

Listing C.3: OCR class definition (file `OnChannelRepeater.m`)

```
classdef OnChannelRepeater < handle

%% Properties
properties
  Gain
  Filt_coeff
  Stability_limit
end

properties (Dependent)
  Gain_dB
  RepDelay
  RepDelay_smpl
  AdaptFiltSize
  StepWidth
  SamplePeriod     % read only
  FB_DelayGain     % write only
  FB_ImpulseResp   % read only
```

```
    FB_Att_dB       % read only
    FB_Delay        % read only
    FB_Delay_smpl   % read only
    FB_Noise_dB
    EQU_mu
    EQU_nrTaps
    EQU_weights
  end

  properties (Access=private)
    repDelay_     % in samples
    adaptFiltSize_
    stepWidth_
    fbChanCoef_
    fbDelay_      % vector containing path delays in samples
    samplePeriod_
    fbNoise_
    equMu_
    equWeights_
    doppler_phase_
  end

  %% Methods
  methods

    %% Constructor
    function obj = OnChannelRepeater(sample_period)
      obj = obj@handle();

      obj.samplePeriod_ = sample_period;
      obj.Gain = db2mag(30);
      obj.RepDelay = 1e-6;
      obj.adaptFiltSize_ = 10;
      obj.stepWidth_ = 1 / (obj.adaptFiltSize_* obj.Gain^2 * 1000);
      obj.Filt_coeff = zeros(obj.adaptFiltSize_, 1);
      obj.FB_DelayGain = [2e-6; -35];
      obj.FB_Noise_dB = -inf;
      obj.Stability_limit = 1e8;
      obj.equMu_ = 0.01;
      obj.equWeights_ = zeros(10,1);
      obj.doppler_phase_ = 0; %2*pi*rand(1,1);
    end

    %% Property Access
    % Gain (dB)
    function gain_dB = get.Gain_dB(obj)
      gain_dB = mag2db(obj.Gain);
    end
    function set.Gain_dB(obj, gain_db)
      obj.Gain = db2mag(gain_db);
    end
    function gain = get.Gain(obj)
```

```
      gain = obj.Gain;
    end
    % Repeater delay
    function set.RepDelay(obj, rep_delay)
      obj.repDelay_ = round(rep_delay / obj.samplePeriod_);
    end
    function r_del = get.RepDelay(obj)
      r_del = obj.repDelay_ * obj.samplePeriod_;
    end
    function set.RepDelay_smpl(obj, rep_delay)
      obj.repDelay_ = rep_delay;
    end
    function r_del = get.RepDelay_smpl(obj)
      r_del = obj.repDelay_;
    end
    % Filter Size
    function set.AdaptFiltSize(obj, fsize)
      obj.adaptFiltSize_ = fsize;
      obj.Filt_coeff = zeros(fsize, 1);
    end
    function fsize = get.AdaptFiltSize(obj)
      fsize = obj.adaptFiltSize_;
    end
    % Step Width (mu)
    function set.StepWidth(obj, mu)
      obj.stepWidth_ = mu;
    end
    function mu = get.StepWidth(obj)
      mu = obj.stepWidth_;
    end
    % Feedback Channel Parameters
    function set.FB_DelayGain(obj, dg_matrix)
      if size(dg_matrix, 1) ~= 2
        error(['The property must be a 2xM matrix, with the ' ...
          'first row being the delay times in seconds and the ' ...
          'second row being the corresponding gain values in dB']);
      end
      if ~issorted(dg_matrix(1,:))
        error(['Elements of the delay vector must be sorted in ' ...
          'ascending order!']);
      end
      nr_paths = size(dg_matrix, 2);
      min_del_diff = inf;
      for i = 2:nr_paths
        del_diff = dg_matrix(1,i)-dg_matrix(1,i-1);
        if del_diff < min_del_diff, min_del_diff = del_diff; end
      end
      if min_del_diff < obj.samplePeriod_
        error(['Path delays are too close to be resolved by the ' ...
          'sampling interval!']);
      end
      obj.fbDelay_ = round(dg_matrix(1,:)/obj.samplePeriod_);
      obj.fbGain_ = dg_matrix(2,:);
    obj.fbChanCoef_ = zeros(obj.fbDelay_(nr_paths), 1);
    for i = 1 : nr_paths
      obj.fbChanCoef_(obj.fbDelay_(i)) = ...
        db2mag(dg_matrix(2,i));
    end
  end
  function fba = get.FB_Att_dB(obj)
    fba = mag2db(obj.fbChanCoef_(obj.fbDelay_)).';
  end
  function fbd = get.FB_Delay(obj)
    fbd = obj.fbDelay_*obj.samplePeriod_;
  end
  function fbd = get.FB_Delay_smpl(obj)
    fbd = obj.fbDelay_;
  end
  function fbn = get.FB_Noise_dB(obj)
    fbn = pow2db(obj.fbNoise_);
  end
  function fbi = get.FB_ImpulseResp(obj)
    fbi = obj.fbChanCoef_;
  end
  function set.FB_Noise_dB(obj, fbn)
    obj.fbNoise_ = db2pow(fbn);
  end
  % Equalizer Settings
  function set.EQU_mu(obj, mu)
    obj.equMu_ = mu;
  end
  function mu = get.EQU_mu(obj)
    mu = obj.equMu_;
  end
  function set.EQU_nrTaps(obj, nr)
    obj.equWeights_ = zeros(nr, 1);
  end
  function nr = get.EQU_nrTaps(obj)
    nr = length(obj.equWeights_);
  end
  function hw = get.EQU_weights(obj)
    hw = obj.equWeights_;
  end
  % Sample Period
  function ts = get.SamplePeriod(obj)
    ts = obj.samplePeriod_;
  end


  %% disp %%%%%%%%%%%%%%%%%%
  function disp(o)
    fprintf('  Repeater Gain:   %3.1f dB (%4.2f)\n', o.Gain_dB, o.
       Gain);
    fprintf('  Filter Size:     %d\n', o.adaptFiltSize_);
```

```
      fprintf('  LMS Step Width:  %5.4g\n', o.stepWidth_);
      fprintf('  Repeater Delay:  %5.4g us (%d samples)\n', ...
        o.repDelay_*o.samplePeriod_*1e6, o.repDelay_);
      fprintf('  Sample Period:   %5.4g us\n', o.samplePeriod_*1e6);
      fprintf('  Feedback Channel:\n');
      for i = 1:length(o.fbDelay_)
        delay = o.fbDelay_(i);  % nr of samples of delay
        fprintf('    Path %d:  %6.4g us   %2d samples   %4.2g dB
           (%6.4g)\n', ...
          i, delay*o.samplePeriod_*1e6, delay, ...
          mag2db(o.fbChanCoef_(delay)), o.fbChanCoef_(delay));
      end
      fprintf('\n');
    end

    %% disp_filt_coeff %%%%%%%%%%%%%%%
    function disp_filt_coeff(o, start_str, mid_str, end_str)
      % Print the current weight values of the adaptive filter
      % in the command window. The output format is (without spaces)
      %
      %   start_str coeff1 mid_str coeff2 mid_str coeff3 end_str
      %

      fprintf(start_str);
      for i = 1:o.AdaptFiltSize-2
        value = sprintf('%7.4f', real(o.Filt_coeff(i)));
        fprintf(value);
        fprintf(mid_str);
      end
      fprintf('%7.4f', real(o.Filt_coeff(o.AdaptFiltSize-1)));
      fprintf(end_str);
    end


    %% disp_equalizer_weights %%%%%%%%%%%%%%%
    function disp_equalizer_weights(o, start_str, mid_str, end_str)
      % Print the current values of the coefficients of the equalizer
      % in the command window. The output format is (without spaces)
      %
      %   start_str coeff1 mid_str coeff2 mid_str coeff3 end_str
      %

      fprintf(start_str);
      for i = 1:o.EQU_nrTaps-1
        value = sprintf('%7.4f', real(o.equWeights_(i)));
        fprintf(value);
        fprintf(mid_str);
      end
      fprintf('%7.4f', real(o.equWeights_(end)));
      fprintf(end_str);
    end


    %% reset %%%%%%%%%%%%%%%%
    function reset(o)
      o.Filt_coeff = zeros(o.adaptFiltSize_, 1);
      o.equWeights_ = zeros(size(o.equWeights_));
    end


    %% train_filter %%%%%%%%%%%%%%%%
    function train_filter(o, nr_samples)
      % o.run(randn(1, nr_samples), 'LMS');
      o.run(wgn(1, nr_samples, 0, 'complex'), 'LMS');
    end

    %% train_equalizer %%%%%%%%%%%%%%%%
    function train_equalizer(o, nr_samples)
      tx_data = wgn(1, nr_samples, 0, 'complex');
      o.run_Equ(tx_data, tx_data);
    end

  end  % end methods


  %% Private Methods %%%%%%%%%%%%%%%%
  methods (Access = private)
    [tx, stable] = run_LMS(o, s)
    [tx, stable] = run_noFIC(o, s)
    tx           = run_noFB(o, s)
  end

  end
```

Listing C.4: Public method `run` (file `run.m`)

```
function [tx, stable] = run(o, s, mode)
stable = true;
switch mode
  case 'noFB'
    tx = o.run_noFB(s); % this mode is always stable
  case 'LMS'
    [tx, stable] = o.run_LMS(s);
  case 'noFIC'
    [tx, stable] = o.run_noFIC(s);
  otherwise
    error('Error calling method run(): Invalid Mode!');
end
```

Listing C.5: Private method `run_LMS` (file `run_LMS.m`)

```
function [tx, stable] = run_LMS(o, s)

```

```
  s = [s, zeros(1, o.repDelay_)];
  N = length(s);
  mu = o.stepWidth_;
  stable = true;

  fb_len  = length(o.fbChanCoef_);
  w_len   = length(o.Filt_coeff);
  tot_len = max(w_len, fb_len);
  % Pad h_fb or w with zeros to a common length
  h_fb = [o.fbChanCoef_; zeros(tot_len-fb_len, 1)];
  w    = [o.Filt_coeff;  zeros(tot_len-w_len, 1)];

  rep_buf = zeros(1, o.repDelay_);
  tx = zeros(1, N);
  tx_buf = zeros(tot_len, 1);
  shift_i = 1:tot_len-1;
  fb_noise = sqrt(o.fbNoise_) * randn(1,N);

  for i = 1 : N

    % Buffer transmit signal (FIFO)
    tx_buf(shift_i+1) = tx_buf(shift_i);
    if i==1, tx_buf(1) = 0; % very first run
    else , tx_buf(1) = tx(i-1);
    end

    % Calculate FB and estimated FB signal
    fb   = h_fb.' * tx_buf + fb_noise(i);
    fb_est = w' * tx_buf;  % Hermitian transp. here!!

    % Get receive signal
    rx = s(i) + fb;

    % Repeater delay
    s_estim = rx - fb_est;
    if o.repDelay_ == 0
      tx(i) = s_estim * o.Gain;
    else
      tx(i) = rep_buf(mod(i, o.repDelay_)+1);
      rep_buf(mod(i,o.repDelay_)+1) = s_estim * o.Gain;
    end

    % Check for stability
    if abs(tx(i)) > o.Stability_limit
      stable = false;
      return
    end

    % Adapt filter coeff
    w(1:w_len) = w(1:w_len) + mu*conj(s_estim)*tx_buf(1:w_len);
  end

  % remove initial samples to keep frame synchronization:
  tx = tx(o.repDelay_+1 : end);
  o.Filt_coeff = w(1:w_len);  % save for next call

end
```

Listing C.6: Private method `run_noFIC` (file `run_noFIC.m`)

```
function [tx, stable] = run_noFIC(o, s)

  s = [s, zeros(1, o.repDelay_)];
  N = length(s);
  stable = true;

  fb_len  = length(o.fbChanCoef_);
  h_fb = o.fbChanCoef_;
  rep_buf = zeros(1, o.repDelay_);
  tx = zeros(1, N);
  tx_buf = zeros(fb_len, 1);
  shift_i = 1:fb_len-1;
  fb_noise = sqrt(o.fbNoise_) * randn(1,N);

  for i = 1 : N

    % Buffer transmit signal (FIFO)
    tx_buf(shift_i+1) = tx_buf(shift_i);
    if i==1, tx_buf(1) = 0; % very first run
    else, tx_buf(1) = tx(i-1);
    end

    % Calculate fb and rx signal
    fb  = h_fb.' * tx_buf + fb_noise(i);
    rx = s(i) + fb;

    % Repeater delay
    if o.repDelay_ == 0
      tx(i) = rx * o.Gain;
    else
      tx(i) = rep_buf(mod(i, o.repDelay_)+1);
      rep_buf(mod(i,o.repDelay_)+1) = rx * o.Gain;
    end

    % Check for stability
    if abs(tx(i)) > o.Stability_limit
      stable = false;
      return
    end
  end

  % remove initial samples to keep frame synchronization:
  tx = tx(o.repDelay_+1 : end);
```

```
end
```

Listing C.7: Private method `run_noFB` (file `run_noFB.m`)

```
function tx = run_noFB(o, s)

  s = [s, zeros(1, o.repDelay_)];
  N = length(s);
  rep_buf = zeros(1, o.repDelay_);
  tx = zeros(1, N);

  for i = 1 : N
    % Repeater delay
    if o.repDelay_ == 0
      tx(i) = s(i) * o.Gain;
    else
      tx(i) = rep_buf(mod(i, o.repDelay_)+1);
      rep_buf(mod(i,o.repDelay_)+1) = s(i) * o.Gain;
    end
  end

  % remove initial samples to keep frame synchronization:
  tx = tx(o.repDelay_+1 : end);
end
```

Listing C.8: Public method `run_rec` (file `run_rec.m`)

```
function [tx, le, dev, re] = run_rec(o, s)
%
% le ........... LMS error:          |rx[n]-fb_est[n]|^2
% dev .......... Squared deviation:  || w[n] - h_fb ||^2
% re ........... Residual error:     |tx[n] - G*s[n-D]|^2
%

  s = [s, zeros(1, o.repDelay_)];
  N = length(s);

  rep_buf = zeros(1, o.repDelay_);
  mu = o.stepWidth_;

  fb_len  = length(o.fbChanCoef_);
  w_len   = length(o.Filt_coeff);
  tot_len = max(w_len, fb_len);
  % Pad h_fb or w with zeros to a common length
  h_fb = [o.fbChanCoef_; zeros(tot_len-fb_len, 1)];
  w    = [o.Filt_coeff;  zeros(tot_len-w_len, 1)];

  tx = zeros(1, N);
  tx_buf = zeros(tot_len, 1);
  shift_i = 1:tot_len-1;
  le = zeros(1, N);
  dev = zeros(1, N);
  re = zeros(1, N-o.repDelay_);

  for i = 1 : N

    % Buffer transmit signal (FIFO)
    tx_buf(shift_i+1) = tx_buf(shift_i);
    if i==1, tx_buf(1) = 0; % very first run
    else , tx_buf(1) = tx(i-1);
    end

    % Calculate FB and estimated FB signal
    fb   = h_fb.' * tx_buf;
    fb_est = w' * tx_buf;  % Hermitian transp. here!!

    rx = s(i) + fb;
    s_estim = rx - fb_est;
    if o.repDelay_ == 0
      tx(i) = s_estim * o.Gain;
    else
      tx(i) = rep_buf(mod(i, o.repDelay_)+1);
      rep_buf(mod(i,o.repDelay_)+1) = s_estim * o.Gain;
    end

    % Adapt filter coeff
    e = rx - fb_est;
    w(1:w_len) = w(1:w_len) + mu*conj(e)*tx_buf(1:w_len);

    % Learning curves:
    le(i) = abs(e)^2;                          % LMS error
    dev(i) = (w-h_fb)' * (w-h_fb);             % Squared deviation
    i2 = i-o.repDelay_;
    if i2>0, re(i2) = abs(tx(i) - o.Gain*s(i2))^2; % Residual error
    end
  end

  % remove initial samples to keep frame synchroniztation:
  tx = tx(o.repDelay_+1 : end);
  le = le(1 : end-o.repDelay_);
  dev = dev(o.repDelay_+1 : end);
  o.Filt_coeff = w(1:w_len);  % save for next call
end
```

Listing C.9: Public method run_Equ (file run_Equ.m)

```
function [tx, stable] = run_Equ(o, s, trainingSeq)

  stable = true;
  s = [s, zeros(1, o.repDelay_)];
  N = length(s);

  train_len = length(trainingSeq);
  fb_len  = length(o.fbChanCoef_);
  fb_noise = sqrt(o.fbNoise_) * randn(1,N);
  h_fb = o.fbChanCoef_;

  equ_mu = o.equMu_;
  equ_w = o.equWeights_;
  nrEquTaps = length(equ_w);

  rep_buf = zeros(1, o.repDelay_);
  tx = zeros(1, N);
  tx_buf = zeros(fb_len, 1);    % For convolution with h_FB
  tx_shift_i = 1:fb_len-1;

  rx_buf = zeros(nrEquTaps, 1); % For equalizer
  rx_shift_i = 1:nrEquTaps-1;



  for i = 1 : N

    tx_buf(tx_shift_i+1) = tx_buf(tx_shift_i);
    if i==1, tx_buf(1) = 0;
    else tx_buf(1) = tx(i-1); end

    fb  = h_fb.' * tx_buf + fb_noise(i);

    rx = s(i) + fb;
    rx_buf(rx_shift_i+1) = rx_buf(rx_shift_i);
    rx_buf(1) = rx;
    equ_out = equ_w' * rx_buf;

    % Check for stability
    if abs(rx) > 1e40
      stable = false;
      return
    end

    % Update Equalizer Weights
    if i <= train_len
      equ_e = trainingSeq(i) - equ_out;
      equ_w = equ_w + equ_mu * conj(equ_e) * rx_buf;
    end


    if o.repDelay_ == 0
      tx(i) = equ_out * o.Gain;
    else      % delay before retransmitting
      tx(i) = rep_buf(mod(i, o.repDelay_)+1);
      rep_buf(mod(i,o.repDelay_)+1) = equ_out * o.Gain;
    end

    end

  % remove initial samples to keep frame synchroniztation:
  tx = tx(o.repDelay_+1 : end);
  % store equalizer weights for next call
  o.equWeights_ = equ_w;
end
```

Bibliography

Jeffrey G. Andrews, Arunabha Ghosh, and Rias Muhamed. *Fundamentals of WiMAX*. Communications Engineering and Emerging Technologies Series. Prentice Hall, 2007. (Cited on pages 51 and 52.)

Axell RCB. *Rural Coverage Brochure*. Axell Wireless, Buckinghamshire, UK. URL `http://www.axellwireless.com`. (Cited on page 8.)

Su Chang Chae et al. Direct relay using M&F. IEEE C802.16j-07/526r3, September 2007. (Cited on page 15.)

Channelot WP. *Transmitters and Repeaters as Digital and Mobile TV Gap Fillers*. Channelot, Tel Aviv, Israel, 2009. URL `http://www.channelot.com`. White Paper. (Cited on page 2.)

Vinko Erceg et al. Channel models for fixed wireless applications. IEEE 802.16.3c-01/29r4, July 2001. (Cited on pages 41, 42 and 86.)

Vasken Genc, Sean Murphy, Yang Yu, and John Murphy. IEEE 802.16j relay-based wireless access networks: An overview. *IEEE Wireless Commun. Mag.*, 15(5):56–63, October 2008. (Cited on pages 12 and 15.)

Simon Haykin. *Adaptive Filter Theory*. Information and System Sciences Series. Prentice-Hall, fourth edition, 2002. (Cited on page 28.)

IEEE 802.16-2009. Air interface for broadband wireless access systems, 2009. URL `http://standards.ieee.org/getieee802/802.16.html`. (Cited on pages 51 and 58.)

Christian F. Lanzani, Georgios Kardaras, and Deepak Boppana. Remote radio heads and the evolution towards 4G networks. Technical report, Radiocomp, Altera, 2010. URL `http://www.radiocomp.com`. (Cited on page 1.)

Peter Larsson and Mikael Prytz. MIMO on-frequency repeater with self-interference cancellation and mitigation. In *Workshop on Wireless Ad-hoc & Sensor Networks*. Ericsson Research, 2008. (Cited on page 20.)

Gerry Leavey. Enabling distributed base station architectures with CPRI. White paper, PMC-Sierra, Inc., Santa Clara, CA, USA, February 2006. (Cited on page 1.)

Young-Jun Lee, Ho Min Eum, Yong-Tae Lee, Kyung Sik Son, and Hyoung-Nam Kim. Performance of feedback cancellers for T-DMB on-channel repeaters. *IEEE Trans. Broadcast.*, 55 (4):810–817, 2009. (Cited on pages 20, 31 and 45.)

George Moschytz and Markus Hofbauer. *Adaptive Filter*. Springer-Verlag, 2007. (Cited on pages 29 and 34.)

Karim M. Nasr, John Cosmas, Maurice Bard, and Jeff Gledhill. Performance of an echo canceller and channel estimator for on-channel repeaters in dvb-t/h network. *IEEE Trans. Broadcast.*, 53(3):609–618, August 2007. (Cited on page 26.)

NOWTEL 1000S. *WiMAX Small RF Repeater*. NOWTEL, R. of. Korea. URL `http://www.now-tel.com`. (Cited on page 20.)

NOWTEL 100S. *WiMAX Ultra Pico RF Repeater*. NOWTEL, R. of. Korea. URL `http://www.now-tel.com`. (Cited on page 20.)

Loutfi Nuaymi. *WiMAX: Technology for Broadband Wireless Access*. John Wiley & Sons, first edition, January 2007. (Cited on page 51.)

Sung Ik Park, Homin Eum, So Ra Park, Geon Kim, Yong-Tae Lee, Heung Mook Kim, and Wangrok Oh. Novel equalization on-channel repeater with feedback interference canceller in terrestrial digital multimedia broadcasting system. *ETRI Journal*, 31(4):357–364, August 2009. (Cited on pages 17, 20, 26 and 31.)

Steven W. Peters and Robert W. Heath, Jr. The future of WiMAX: Multihop relaying with IEEE 802.16j. *IEEE Commun. Mag.*, 47(1):104–111, January 2009. (Cited on pages 12 and 15.)

Renaissance Rep. *LTE/WiMAX Repeaters*. Renaissance Electronics Corp., Harvard, MA. URL `http://www.rec-usa.com`. (Cited on page 20.)

RFI-CE900. *900 MHz GSM Medium Power Repeater*. RFI, North Rocks, Australia. URL `http://www.rfi.com.au`. (Cited on page 8.)

Taneli Riihonen, Stefan Werner, and Risto Wichman. Optimized gain control for single-frequency relaying with loop interference. *IEEE Transactions on Wireless Communications*, 8(6):2801–2806, June 2009. doi:`10.1109/TWC.2009.080542`. (Cited on page 20.)

W.T. Slingsby and J.P. McGeehan. Antenna isolation measurements for on-frequency radio repeaters. In *Ninth International Conference on Antennas and Propagation*, volume 1, pages 239–243, 1995. (Cited on pages 20 and 21.)

Hiroshi Suzuki, Kazuhito Itoh, Yoshio Ebine, and Mitsuo Sato. A booster configuration with adaptive reduction of transmitter-receiver antenna coupling for pager systems. In *Proc. IEEE 50th Vehicular Technology Conference (VTC-Fall 99)*, volume 3, pages 1516–1520, September 3 1999. (Cited on page 20.)

Jerry Sydir et al. Harmonized contribution on 802.16j (mobile multihop relay) usage models. IEEE C802.16j-06/015, September 2006. (Cited on pages 2, 11 and 12.)

TTT WiMAX Rep. *WiMAX Repeater*. TTT, Las Rozas (Madrid), Spain. URL http://www.ttt.es. (Cited on page 20.)

Adam Wiewiorka and Peter N. Moss. Digital on-channel repeater for DAB. White paper whp 120, BBC R&D, September 2005. (Cited on page 17.)

WMF-FAQ. Technology FAQ. URL http://www.wimaxforum.org/resources/frequently-asked-questions. (Cited on page 52.)

WMF-T23. Mobile system profile specification – common part, August 2009. URL http://www.wimaxforum.org/resources/documents/technical/T23. WMF-T23-001-R015v01. (Cited on page 54.)