Design of a Radio Communications Protocol for HAMNET Access in the 70cm Amateur Radio Band

Lukas Ostendorf

Master Thesis

June 15, 2020

Examiners

Prof. Dr. Petri Mähönen
Prof. Dr.-Ing. Marina Petrova

Supervisors

Prof. Dr. Petri Mähönen
Dr. Felix Hoffmann, Dr. Ljiljana Simić

This thesis has been submitted to pursue
a Master's Degree at the

Institute for Networked Systems
RWTH Aachen University

The present work was submitted to the Institute for Networked Systems

Design of a Radio Communications Protocol for HAMNET Access in the 70cm Amateur Radio Band

Master Thesis

presented by
Lukas Ostendorf

Prof. Dr. Petri Mähönen
Prof. Dr.-Ing. Marina Petrova

Aachen, June 15, 2020

_____

(Lukas Ostendorf)

Acknowledgements

I would like to thank Dr. Felix Hoffmann, Christian Obersteiner and Jann Traschewski from Rohde&Schwarz for lots of fruitful discussions throughout the thesis and the insights into the amateur radio community they gave. Furthermore, I would like to thank Prof. Mähönen for supervising this thesis, his availability in case of questions and his helpful comments.

Contents

ABSTRACT

The project "High-speed Amateur-radio Multimedia NETwork" (HAMNET) tries to establish an internet protocol (IP) based network purely based on wireless links. Currently, 2.3 GHz and 5.7 GHz carrier frequencies and channel bandwidths of 10 MHz are used. In combination with high order quadrature amplitude modulation, this gives data-rates of up to 65 Mbps and links covering distances of 50 km and more. However, the usage of such high carrier frequencies limits the system to line-of-sight connections. This limits the coverage and prohibits easy access to the system. In order to provide more coverage and ease the access to the network, we propose a new communication system using the lower 430 MHz frequency band. Leveraging this frequency band, communication without line-of-sight is still possible, yielding better coverage. This thesis will provide an analysis of the expected channel conditions in the 430 MHz spectrum and derive requirements from expected usage scenarios. We then propose our OFDM based solution and implement it on software defined radios. We evaluate our proposal in simulations and under real world conditions.

1

Introduction

Amateur radio is the communication via radio frequencies for non-commercial interests. Amateur radio operators are licensed and have to restrict their transmissions to certain frequency bands defined by federal agencies. However, within the band and transmit power restrictions, they are free to choose any communication protocol or even define completely new ones. Examples of amateur radio services are voice communication, television broadcasting or satellite communication. Digital communication applications have been used for a long time in the amateur radio community. It started in the 1980s with the establishment of packet radio services based on 1200 baud audio frequency shift keying modems. At link level AX.25, a derivative of the X.25 protocol, was used. With AX.25, communication over multiple nodes is possible. The packet header might not only specify the destination of the packet, but also intermediate stations through which the packet shall be routed. These relay nodes – called "digipeaters" – enabled communication over very long distances. In a time, in which the nowadays self-evident internet communication was not widespread or possible at all, this lead to a certain popularity of packet radio in the 80s and 90s. However, with the spread of the global internet, the popularity of packet radio decreased. Except for slightly faster modulation with 9600 baud or even 19200 baud, no big technology advances took place. Furthermore, using the Internet Protocol (IP) stack on top of the AX.25 link layer had a poor performance, which limited the user applications of packet radio.

In order to eliminate the flaws of packet radio, a new project called "High-speed Amateur-radio Multimedia NETwork" (HAMNET) emerged in 2009. Its goal is to use state of the art technology in order to provide fast IP based inter-connectivity over wireless links. Modified WiFi routers are used to create links in the 2.3 GHz and 5.7 GHz bands. Channel bandwidths of up to 10 MHz in combination with usage of the IEEE 802.11 standards can provide data rates of up to 65 Mbit/s. The use of highly directional antennas permit link distances of more than 50 km. There are currently more than 500 HAMNET backbone stations being operated in Germany [1]. However, due to the high frequency bands that are used, getting access to the network is only possible in line-of-sight (LOS) conditions to an existing backbone station. It is often not possible to get in line of sight of a backbone station, e.g. in larger cities. Therefore another technology is needed to provide the "last mile" link. Due to better propagation properties, line of sight is not strictly required if lower frequency bands – such as the 70 cm band – were to be used. A combination of the existing HAMNET backbone and 70 cm access networks could provide both good coverage and good data rates.

This thesis makes a contribution towards a new HAMNET access network by analyzing the channel conditions and the resulting requirements for a HAMNET access system in the 70cm band. Special emphasis is put on whether it would be ben-

eficial to use orthogonal frequency division multiplex (OFDM) in such a scenario. We then compare existing protocol proposals and assess how they fit our derived requirements. Finally, we propose a new system design for HAMNET access in the 70 cm band which we implemented on a software defined radio (SDR) platform. The implementation is validated by simulations and real world tests.

The remaining thesis is structured as follows: Chapter 2 will provide background information on channel modeling, multi-user communication techniques and protocols and orthogonal frequency division multiplex. In the next chapter, we derive requirements for the system and model expected channel properties for our usage scenarios. They are then used to design the physical and MAC layer of the new communication system. Chapter 4 will cover the implementation details of the system design. The system is evaluated in Chapter 5. Here, we verify the overall system functionality and assess the performance of some implementation details. Finally, we summarize our results and give an overview what should be investigated further in the future in Chapter 6.

2

BACKGROUND

This chapter will provide background information on the most important aspects of a communication system, helping the reader to understand the system design process. Before we start with the theoretical background, we will give some more insights on the HAMNET project. We will then introduce the principles of channel modeling and derive models for large scale and small scale fading effects. Then, we will have a closer look at the principles of multi-user communication. We first treat the concept of multiplexing and then look at how access to multiplexed resources can be controlled. The last part of this chapter is dedicated to OFDM modulation and how signal processing steps like timing synchronization and channel estimation can be done in OFDM systems.

2.1 HAMNET

HAMNET is an IP-network based on wireless links. The idea to build such a network emerged in 2009 in Austria [2]. Since it requires much effort and cost to develop own hardware that supports high speed wireless connections, already available products on the market are used to build the network. It was decided to modify WiFi routers, since they support the IEEE 802.11 standards, giving high data rates. WiFi routers usually transmit in the 2.4 and 5 GHz bands, but some can be modified to use the 2.3 GHz and 5.7 GHz amateur radio bands. When operated in these bands, the maximum transmission power is not limited to 20 dBm as in the WiFI ISM bands, this increases the link distance up to several kilometers [2]. A channel bandwidth of 10 MHz enables data-rates of up to 65 Mbps. Use cases for such a network are connecting and controlling other amateur radio services as APRS or Amateur Television. Telephony is made possible by setting up VoIP servers in the HAMNET. Finally, a IP-network that is separated from the commercial internet could serve as a communication backup in case of disaster scenarios like power outages. All links that belong to the HAMNET are managed at the wiki-like website *hamnetdb* [1]. This website gives information on existing links, the link quality and assigned IP addresses to the backbone nodes. An overview of the available links around Munich is given in Figure 2.1.

2.2 COMMUNICATION CHANNELS

Any communication system consists of a transmitter, a receiver and a channel over which information is transmitted. In wireless communications, the channel is the atmosphere. It propagates electromagnetic waves that are radiated by an antenna. There are three propagation types, depending on the used frequency band: groundwave, sky-wave and line-of-sight (LOS) propagation.

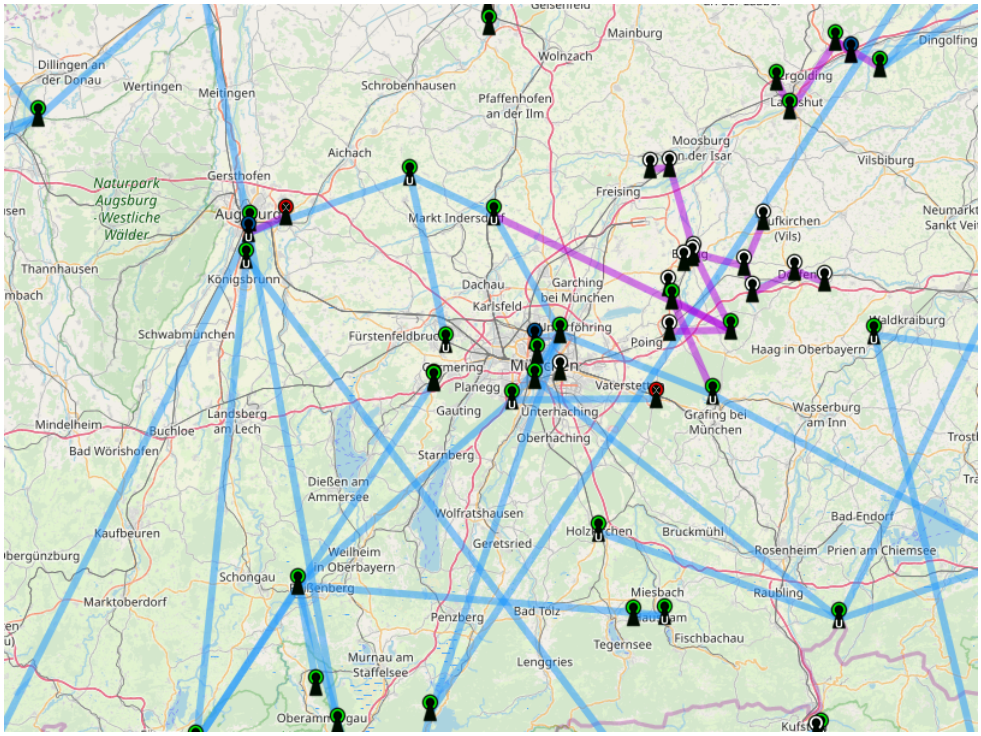


FIGURE 2.1: Map of the realized HAMNET backbone links around Munich. Taken from [1].

In the medium frequency band (MF, 0.3-3 MHz) the dominant propagation type is ground-wave propagation. In ground-wave propagation, the electromagnetic waves follow the curvature of the earth. The range of a transmission system using these frequency bands is therefore only limited by the transmitter power and the noise in the channel. In sky-wave propagation, the electromagnetic waves are being reflected by the ionosphere in the sky. The reflection strength depends on the weather conditions. In general, sky-wave propagation is only possible using frequencies below 30 MHz. Higher frequencies mostly pass through the ionosphere. For frequencies above 30 MHz, the dominant mode of propagation is line of sight propagation. Here, the transmitter and receiver antennas must be in a direct line of sight with little obstruction. The maximum link distance is therefore limited by the height of the transmitter and receiver antenna and curvature of the earth. Thus, broadcasting stations using these frequencies have their antennas mounted to high towers [3]. Also in the ultra-high-frequency band (UHF 300 Mhz-3 GHz), to which the 70cm band belongs, LOS propagation is the predominant propagation mode.

However, in the very-high frequency band (VHF, 30 MHz-300 MHz) and the lower UHF band, communication is still possible if the direct LOS path is obstructed. This can be broken down to mainly three effects: reflection, diffraction and scattering. Reflection occurs when an electromagnetic wave hits a smooth object. Diffraction is the phenomenon that electromagnetic waves bend around sharp edges of obstructing objects. When scattering occurs, an electromagnetic wave is being reflected in many different directions. This happens when a wave hits rough, irregular surfaces [4].

2.2.1 Path Loss modeling

In order to predict the average loss in the signal strength caused by these phenomena, many propagation models were developed. They are either purely analytical, or based on empirical measurements.

Analytical models The first type of models tries to derive mathematical formulas from the physical properties of electromagnetic waves. The most widely known model is probably the free space path loss model. It predicts the received signal strength for an unobstructed transmission. The received signal strength is given by Friis free space equation:

$$P_r = \frac{P_t G_t G_r \lambda^2}{(4\pi d)^2 L}, \tag{2.1}$$

where P_t is the transmitter power, P_r is the received power, G_t and G_r are the transmitter and receiver gains, λ is the wavelength and d is the distance between sender and receiver. The corresponding path loss is given by:

$$P_L(dB) = 10 \log \frac{P_t}{P_r} = -10 \log(\frac{G_t G_r \lambda^2}{(4\pi d)^2}). \tag{2.2}$$

We can see that the received signal power falls of with the square of the distance. This is due to that the transmitter is being modeled as a point source, where waves emerge in a spherical way. The power density at the spherical hull is inversely proportional to the square of the distance. If the aperture to collect the electromagnetic energy is fixed, e.g. the antenna properties are fixed, the received power is also degrades with d^2 [4]. In real world conditions, the predictions of this model are rarely met, the real world performance is often worse [5]. Therefore, this model can be seen as the best case scenario of the encountered path loss. Other analytical models develop expressions for reflection, diffraction and scattering effects. We will not discuss them further.

Empirical models Many path loss models use the empirical modeling approach. In this approach, real world measurements are performed and curve fitted to extract analytical expressions. A simple empirical model is the log distance path loss model. Here, the path loss is given by

$$P_L(d, d_0) = P_L(d_0) + 10n \log(\frac{d}{d_0}). \tag{2.3}$$

The path loss at a distance d is derived from the path loss at another distance d_0, which is previously known. The path loss increases with the distance d, where path loss exponent n states how fast the power decreases. For $n = 2$, the path loss is modeled just as in the free space path loss formula. Measurements showed that a exponent n from 2.7 to 3.5 describes the path loss in an urban area cellular network [5].

A more advanced and widely used empirical model is the Hata model. In 1960, Okumura developed graphical curves to get a correction factor for the free space path loss based on measurements made in Tokyo. The model is applicable for frequencies from 150 MHz to 1920 MHz and distances ranging from 1 km to 100 km. The model parameters are the receiver and transmitter antenna height, the frequency and the

type of terrain. The terrain is split into the three categories urban, suburban and rural [5] [4]. The Hata model (or sometimes Okumura-Hata model) is a set of equations derived from the graphical Okumura path loss data. The median path loss L_{50} in urban terrain is given by

$$\begin{aligned} L_{50}(urban)(dB) = 69.55 + 26.16\log(f_c) - 13.82\log(h_{te} - a(h_{re}) \\ +(44.9 - 6.55\log(h_{te}))\log(d), \end{aligned} \tag{2.4}$$

where f_c is the frequency in MHz (150 MHz-1500 MHz), h_{te} is the transmitter height in meters (30 m-200 m), h_{re} is the receiver height (1m-10m) and d is the link distance in meters. $a(h_{re})$ is a correction factor based on the receiver height and the type of city in which the receiver is located. For a small to medium city it is given by [5]

$$a(h_{re}) = (1.11\log(f_c) - 0.7)h_{re} - (1.56\log(f_c) - 0.8)\,\mathrm{dB} \tag{2.5}$$

for large cities it is given by

$$a(h_{re}) = 3.2(\log(11.75h_{re}))^2 - 4.97\,\mathrm{dB} \quad for f_c \geq 300\,\mathrm{MHz}. \tag{2.6}$$

For suburban areas the path loss formula for urban areas is modified

$$L_{50}(dB) = L_{50}(urban) - 2[\log(f_c/28)]^2 - 5.4. \tag{2.7}$$

Finally the equation for rural areas is

$$L_{50}(dB) = L_{50}(urban) - 4.78(log(f_c))^2 - 18.33log(f_c) - 40.98. \tag{2.8}$$

2.2.2 Small Scale Fading

In the previous section, the median path loss that affects a transmission has been described. As we have seen, it is mainly determined by the distance between the sender and receiver. But the received signal can also vary with very small changes in the receiver or transmitter position. This effect is called small scale fading and will be investigated further in this section. Small scale fading is caused by multi-path propagation, moving transmitters or receivers and moving objects in the channel [5].

2.2.2.1 Multipath Propagation

The effects of scattering, diffraction and reflection cause multiple versions of the signal to arrive at the receiver station. They arrive with different amplitudes, different phases and at different points in time. This channel effect can be modeled as a linear time variant filter

$$r(t) = s(t) * c(\tau; t), \tag{2.9}$$

where $s(t)$ is the time discrete input signal, $c(\tau; t)$ is the possibly time variant impulse response and $*$ denotes convolution. The filter taps in $c(\tau; t)$ model the delayed multi-path signal components. If the number of nonzero taps is greater than one, every

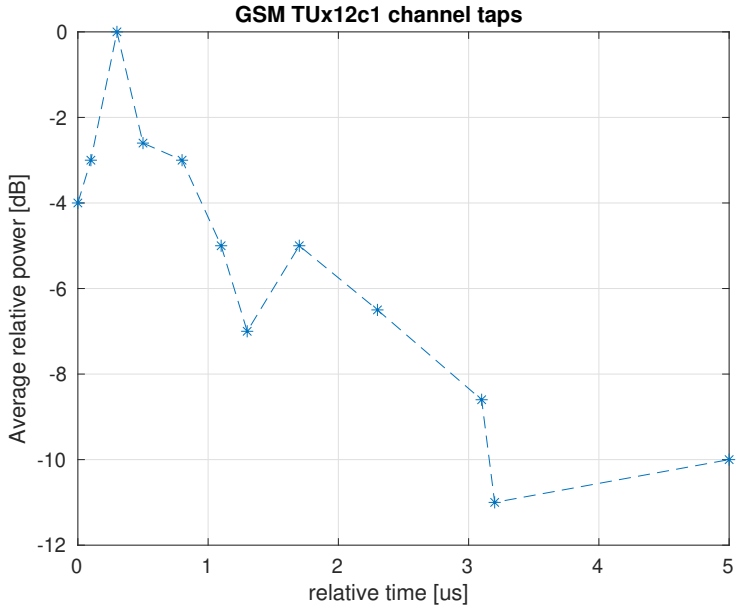


FIGURE 2.2: Power-delay profile of the GSM typical urban 12-tap channel 1 (gsmTUx12c1) as defined in [7].

symbol is interfered by previously sent symbols. This inter-symbol interference (ISI) degrades the system performance. We further define

$$R_c(\tau) = E[c^*(\tau; 0)c(\tau; 0)] \tag{2.10}$$

as the average channel response's power output, which is also called the power-delay profile of the channel. It can be measured by sending short pulses and measuring the received power over time.

Such measurements have been used to model the power delay profile for different channels. Models exist for basically any larger communications standard. As such, the 3rd Generation Partnership Project (3GPP) defined multipath models for GSM, UMTS and LTE. The channel model for the GSM system is based on research carried out by the European research initiative COST (Cooperation in Science and Technology) 207. They developed a average power delay profile for three terrain classes: Rural area, Hilly terrain and Urban area. [6]. The channel taps are defined by the relative time of arrival and relative received power. For every terrain type, two channel configurations are given. The channel taps for the first typical urban scenario are visualized in Figure 2.2.

Later, another joint European research initiative COST 259 developed a more advanced channel model. It was used during the development of the cellular communications standard UMTS. Here, the multipath components are modeled as multiple clusters. The clustering pattern occurs since main components of the signal are being reflected at a few larger objects. Since also scattering and diffraction may occur at this object, many more signal paths are created, but they all are related to one initial

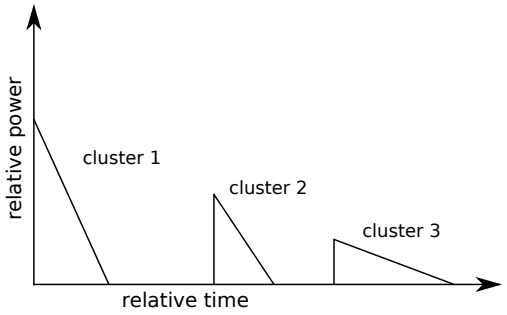


FIGURE 2.3: Example of a clustering power delay profile.

event and therefore can be grouped [6]. Figure 2.3 gives an example of this channel clustering model.

More recent channel models such as the model for LTE [8] put their focus on wide-band cellular systems. Other modern models, such as the models for the IEEE 802.11 standards, are dedicated to indoor environments or outdoor environments with shorter ranges. They are outside of our scope of this thesis and will not be discussed further.

2.2.2.2 Delay Spread

In order to give short descriptions of the channel, some commonly used parameters will be derived from the multipath channel description. The *mean excess delay* is given as

$$\hat{\tau} = \sum_k \frac{R_c(k)k}{R_c(k)}. \tag{2.11}$$

The rms (root-mean-square) *delay spread* is defined as

$$\hat{\tau}_{rms} = \sum_k \frac{R_c(k)k^2}{R_c(k)}. \tag{2.12}$$

We can see that this is the sqare root of the second moment of the power delay profile. At last, the *maximum excess delay* $\hat{\tau}_{max}$ denotes the delay from the first received signal component until the last signal component. These parameters quantify how far the multipath components are spread in the time domain. Using the rms delay spread, the *coherence bandwidth* can be defined.

The definition of the coherence bandwidth varies, as there exists no exact relationship between these values [5]. According to Rappaport [5] and Seybold [4], the bandwidth over which the frequency correlation function is above 0.9 is approximately

$$B_c = \frac{1}{50\hat{\tau}_{rms}}. \tag{2.13}$$

The bandwidth over which the coherence is above 0.5 is defined to be

$$B_c = \frac{1}{5\hat{\tau}_{rms}}. \tag{2.14}$$

In contrast, Proakis estimates the coherence bandwidth in [3] as

$$B_c = \frac{1}{\hat{\tau}_{max}}. \tag{2.15}$$

In the last expression, the maximum excess delay is used instead of the delay spread. Since it will be larger than the rms delay spread, the estimated coherence bandwidth using this equation can nevertheless be in the range of one of Rappaport's definitions.

The coherence bandwidth is used to classify a channel as either flat or frequency selective. When the signal bandwidth B is larger than the coherence bandwidth B_c, the channel is called frequency selective. If $B < B_c$, the channel is called flat. In a flat channel, the frequency response of the channel has approximately the same amplitude and phase over all frequencies of the input signal. In to characterize the delay spread in different environments, extensive measurements have been performed. It is widely perceived that the delay spread is log-normal distributed and increases with the link distance [9] [10]. In 1997 Greenstein *et al.* developed a model based on this assumptions. His results were also used in the COST 259 model [11]. As in most empirical models, he classifies different terrain categories: Urban, Suburban, Rural and Mountainous. The measurement results suggest that the delay spread can be estimated as

$$\tau_{rms}(d) = T_1 * d^{\epsilon}, \tag{2.16}$$

where ϵ is 0.5 in all environments except mountainous terrain. T_1 is the median delay spread at a distance of 1 kilometer.

2.2.3 Doppler Spread

Another widely used parameter to classify the multipath channel is the *Doppler spread*. The Doppler spread is used to describe the time variant component of a channel. It is defined as the range of frequencies over which the Doppler spectrum is nonzero. Let us say that a basestation and a mobile station are moving relatively to each other at velocity v and an angle θ. Then the Doppler shift f_d can be written as

$$f_d = \frac{v \cos(\theta)}{\lambda}. \tag{2.17}$$

The maximum Doppler shift is reached if the nodes move directly towards each other ($\theta = 0$) or apart from each other ($\theta = \pi$).

$$f_m = \frac{v}{\lambda}. \tag{2.18}$$

Using the maximum Doppler shift, the *coherence time* T_c of the channel can be defined. The coherence time defines the time over which the channel impulse response is nearly invariant. The coherence time is inversely proportional to the maximum Doppler shift. Two common approximations are used to estimate the coherence time:

$$T_c = \frac{6}{16\pi f_m}, \tag{2.19}$$

$$T_c = \sqrt{\frac{9}{16\pi f_m^2}} = \frac{0.423}{f_m}. \tag{2.20}$$

The coherence time is used to classify channels as fast or slow fading. In slow fading channels, the symbol time T_s is shorter than the coherence time T_c, e.g. the channel response does not vary over multiple data-symbols. The relation of T_c and T_s can be used to define how often the channel has to be estimated.

2.2.4 Rayleigh Fading

The time varying nature of a wireless communications channel in non-LOS conditions is typically modeled using the Rayleigh distribution. A Rayleigh distribution can be obtained by taking the square root of two quadrature Gaussian noise signals. If the channel undergoes flat fading, the signal is multiplied with a single Rayleigh distributed random variable. In the case of frequency-selective fading, each of the signal paths is modeled by an independent Rayleigh distribution. Instead of generating the Rayleigh distribution with the sum of squared Gaussian signals, *Jakes' model* can be used. Jakes proposed a deterministic way of generating a Rayleigh distribution by summing up sinusoidal signals in [12]. He assumed that N rays arrive at a moving receiver with uniformly distributed arrival angles $\alpha_n = 2\pi n/N$. The experienced Doppler shift for each ray is $w_n = 2\pi f_m * \cos(\alpha_n)$, where $f_m =$ is the maximum Doppler shift. Let $N_0 = (N/2 - 1)/2$. The total received signal at time instance t is then:

$$\begin{aligned} T(t) &= \frac{1}{\sqrt{2}} \cos(2\pi f_m t + \theta_0) \\ &+ \sum_{n=1}^{N_0} [\cos(\beta_n) + i \sin(\beta_n)] \cos(w_n t + \theta_n). \end{aligned} \tag{2.21}$$

In order to get zero cross correlation between the real and imaginary part, usually $\beta_n = \pi n/(N_0 - 1)$ is chosen. Different phases θ_n should be chosen if multiple waveforms for the frequency selective channel model have to be generated. In case of a flat fading channel model, only a single distribution is needed and θ_n can be set to 0. An example of a Rayleigh channel with 20 Hz Doppler shift is given in Figure 2.4. Characteristically are the short time *deep fades*, in which the signal strength decreases starkly for a moment. It is caused by destructive interference of the scattered signal components.

2.3 MULTIPLEXING TECHNIQUES

In the following we will show different techniques of multiplexing multiple transmissions through one common shared channel. This gives the basis for multi-user communication systems.

The first method is to subdivide the available bandwidth into non-overlapping subchannels, commonly known as Frequency Division Multiple Access (FDMA). Another approach is to divide time domain into multiple repeating intervals, assigning the intervals to different users. This method is referred to as Time Division Multiple

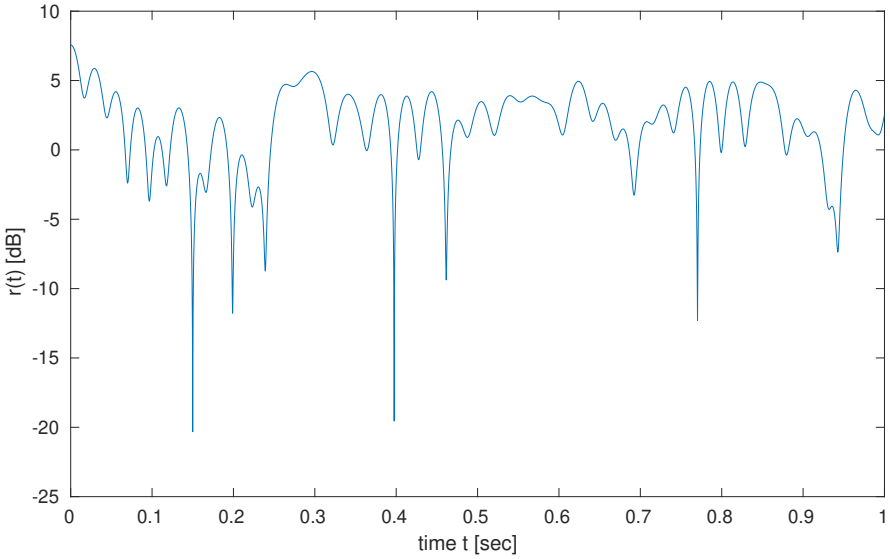


FIGURE 2.4: Rayleigh fading generated with Jakes model. The maximum Doppler shift f_m is set to 20 Hz and $N_0 = 8$ sinusoids have been used.

Access (TDMA) [3]. At last, it is possible use spread spectrum signals to differentiate between different users. With this technique, the bandwidth of the signal is widened by multiplying each sample of the data signal with a spreading sequence. The spreading sequences have good auto-correlation properties and a low cross correlation. Using a correlation receiver, it is possible to detect only one desired signal, even if multiple spreading sequences are transmitted. This multiplexing technique is called Code Division Multiple Access (CDMA) [3]. One should note that the capacity of such a system is not higher than the capacity of TDMA or FDMA systems. If the total bandwidth is fixed, the data-rate when using a spreading sequence of length N is $1/N$ of the channel bandwidth. Furthermore, the number of spreading sequences that can be used simultaneously is limited.

The simplest way of enabling multiuser communication, is to statically subdivide the channel and assign a portion of the channel to each user. Problems arise, if the number of potential users is big and if the network traffic is of bursty nature. The more users a network shall support, the less resources a single user can get. Since traffic usually is bursty, users usually need many resources for a short time, followed by a longer pause without any resource needs. Here, the static resource assignment results in a bad resource utilization [3]. The next section will introduce into more advanced protocols for multiuser access. They still perform channel multiplexing using one of the three presented techniques, but assign the resources dynamically.

2.4 MEDIUM ACCESS CONTROL

In wireless communications, all communication participants broadcast into the same shared medium. When users use the same resources simultaneously, transmission

errors will occur. Therefore the shared medium is subdivided using the previously presented multiplexing methods. Medium Access Control (MAC) protocols manage the usage of the shared resources to enable fair and efficient transmissions [13].

To be able to compare different MAC protocols, we will shortly introduce different performance metrics that are commonly used. Most important are the delay and the throughput. The delay is equal to the average time that a packet spends in the MAC queue, e.g. from the en-queuing till the successful transmission. The throughput is defined as the portion of the channel capacity which is used for data transmission. Moreover, a MAC protocol should be fair, i.e. the bandwidth of the channel should be split up between all clients. The exact definition of fairness can differ, since clients can have different requirements regarding delay and data rate. A good MAC protocol should minimize the delay while maximizing the throughput and give a fair share of the bandwidth to all clients [13].

Next, we will classify different MAC protocols and introduce the most commonly used. Wireless MAC protocols can be classified into distributed and centralized protocols, depending upon the type of network architecture they were designed for. Protocols can then be further classified by the type of access. In random access protocols, all nodes can initiate a transmission in an uncontrolled, random manner. If only one node tries to transmit, the transmission will be successful. When multiple nodes transmit at the same time, a collision will occur. Random access protocols define procedures to resolve this collision. Guaranteed access protocols allocate dedicated resources for every node, so all transmissions are guaranteed to be collision free. Resource allocation is typically done in a master-slave system, where one master is responsible for the allocation. Hybrid access protocols use a mixture of random and guaranteed access. Most hybrid access protocols use a request-grant system. Clients send requests for data-transmissions using a random access protocol. The basestation then allocates dedicated resources based on the requests from the clients [13].

2.4.1 Distributed MAC Protocols

One of the first distributed random MAC protocols is ALOHA. It was designed in the 1970s for packet radio networks. The protocol is defined as follows: When a node has data to send it will transmit it. If a collision occurs, the node will re-transmit after a random period [14]. We can calculate the throughput of the protocol in dependence of a offered load G, where $G = 1$ means that the offered load is just as much as the channel can handle. It can be shown that the throughput S is equal to:

$$S = Ge^{-2G}. \tag{2.22}$$

We can easily see that the maximum throughput is reached for an offered load $G = 0.5$ which is about $S_{max} = 0.184$. The throughput can be improved by adding a slot structure. This addition is called slotted-ALOHA. Defining time slots in which transmissions can occur, decreases the probability of packet collisions. The maximum throughput of slotted aloha is 0.368 for an offered load of $G = 1$, twice as much as for pure ALOHA.

A big drawback of the ALOHA protocol is that transmissions are immediately started, regardless whether the channel is already in use. This observation leads to

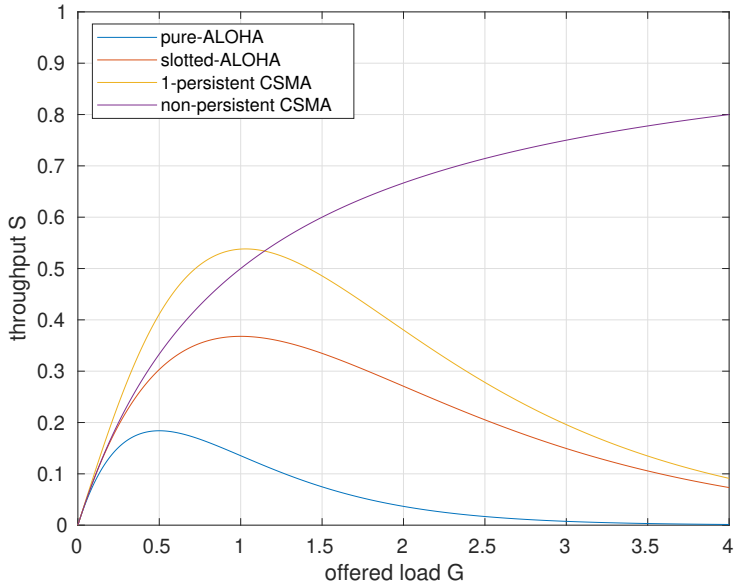


FIGURE 2.5: The maximum throughput of ALOHA, slotted ALOHA, 1-persistent CSMA and non-persistent CSMA for different offered loads.

more advanced Carrier Sense Multiple Access (CSMA) protocols. In CSMA protocols the channel is sensed before a transmission is attempted. Only if the channel is perceived as free, the transmission is started. Otherwise the node waits. In the so called 1-persistent CSMA protocol, each node continuously senses the channel and transmits as soon as the channel is free. This leads to more collisions when the channel utilization is high, since multiple stations start transmitting at the same time. In p-persistent CSMA this problem is mitigated, by only transmitting with a probability p when the channel is sensed free. At last, in non-persistent CSMA, nodes will not continuously monitor the channel. Instead they wait a random period before they sense again, if the channel already is in use. This again results in a better channel utilization under high loads. However, the latter two protocols have the drawback that they increase the delay of the system, even if there is no high load. The throughput for the described ALOHA and CSMA protocols is visualized in Figure 2.5.

2.4.2 Centralized MAC Protocols

In centralized MAC protocols the intelligence of the protocol is moved to the basestation. It is assumed that every client is able to communicate with the basestation. Simple guaranteed access MAC protocols are based on polling. The basestation polls each client regularly, e.g. in a round robin fashion. These protocols give a good channel utilization, if all clients have to transmit data. Since no collisions occur, no bandwidth is wasted for retransmits. However, if some nodes have no data to transmit, slots assigned to them are unused and a waste of resources. Especially under bursty network traffic this leads to a bad resource utilization. Furthermore, the delay

is unnecessarily high under low load scenarios. Due to the round robin assignments, users have to wait for a longer time until they can transmit compared to random access protocols.

These drawbacks lead to hybrid MAC protocols, which try to combine random and guaranteed access for good utilization with low delays. Such protocols usually split the available resources into random access resources and data transmission resources. The random access resources are accessed using previously described random access protocols like ALOHA. Here, all nodes can require resource assignment for transmissions. The basestation then reserves resources for those clients that successfully contented. If the random access resources are small compared to the data transmission resources, these protocols give a good channel utilization.

2.5 ORTHOGONAL FREQUENCY DIVISION MULTIPLEX

As the name suggest, OFDM is a special case of the frequency division multiplexing method. In OFDM, the sub-bands, to which the original frequency band is split, are orthogonal. This is achieved by using the carrier signal

$$s_k(t) = \cos 2\pi f_k t \tag{2.23}$$

for the k-th sub-band, where f_k is the carrier frequency. If we use a symbol rate of $1/T$ at every sub-band and fix the carrier frequency separation of adjacent sub-bands to $1/T$, all sub-bands are orthogonal [3]:

$$\int_0^T a_k \cos(2\pi f_k t + \phi_k) a_i \cos(2\pi f_i t + \phi_i) dt = 0. \tag{2.24}$$

The adjacent channel separation is $1/T$, thus $f_k - f_i = n/T$, where n is integer valued. a_k and ϕ_k are the modulation parameters and constant within one symbol period.

Such a system is commonly realized by using the fast Fourier transform (FFT). A N-point inverse discrete Fourier transform (IDFT) creates a signal from N orthogonal sinusoids, with a frequency separation of $1/T$

$$x(t) = \frac{1}{\sqrt{N}} \sum_{k=0}^{N-1} X_k e^{j2\pi kt/T} \quad 0 \leq t \leq T. \tag{2.25}$$

We can use X_k to represent a complex valued signal constellation that is modulated on each subcarrier. Picking up the previous generic OFDM system definition X_k is given as $X_k = a_k e^{j\phi_k}$. In order to get the constellation points X_k from the signal $x(t)$, a DFT at the receiver can be used. The signal generation in OFDM transceivers is implemented by buffering a complex valued stream of constellation points. Then N constellation points are fed in parallel to a IDFT unit. The output of the IDFT is connected to a parallel to serial mapper. The resulting stream is fed to a digital to analog converter. A block diagram of such an transmitter and the corresponding receiver is given in Figure 2.6. As discussed in the section on multipath, a channel impulse response with $v > 1$ filter taps unequal zero introduces ISI to our system. If we assume that our OFDM system uses many subcarriers, $N >> v$ holds. By

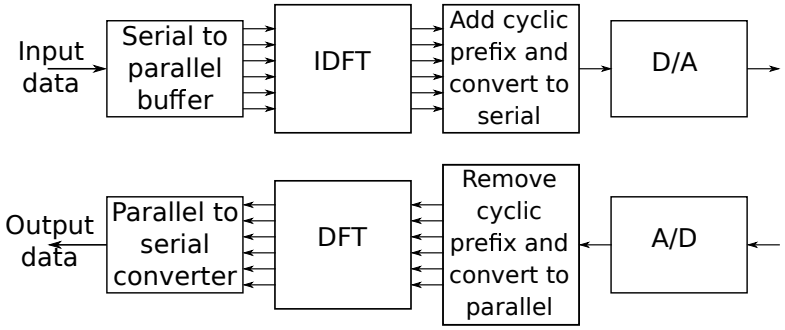


FIGURE 2.6: The Block diagram of an OFDM transmitter and receiver.

introducing a guard time of at least $v+1$ samples, ISI can be eliminated. This happens at the cost of losing bandwidth efficiency: $\frac{v+1}{N+v+1}$ of the system time will not be used for data transmission. However, in the case of $N >> v$ this loss can be neglected. Instead of using a guard interval, also a cyclic prefix can be used. When using a cyclic prefix, the the last v samples of an OFDM symbol are prepended to the same OFDM symbol. Cyclic prefixes are widely used in all modern OFDM systems, i.e. in LTE, IEEE 802.11a, and later 802.11 standards [15] [16].

2.5.1 Synchronization in OFDM

After clearing up the basics of an OFDM transmission system, we will look at some concrete algorithms used for the implementation of an OFDM system. The first task in any communication system is to achieve timing synchronization. Additionally, especially in OFDM systems it is important to have a good estimation of the carrier frequency offset. In OFDM systems, a carrier frequency mismatch between the transmitter and receiver results in a loss of orthogonality between subchannels, which severely degrades the system performance [17].

A widely used timing synchronization algorithm that also estimates the carrier frequency offset has been published by Schmidl & Cox in 1997 [18]. As most timing synchronization algorithms, it is based on searching for a training sequence in the stream of received samples. The training sequence is made out of two identical sequences in time domain. They are generated by transmitting a pseudonoise (PN) sequence at all even subchannels while transmitting zeros on the other subchannels. A second training symbol consists of one PN sequence on the odd subchannels and another one on the even subchannels and is used for a better frequency offset estimation.

The following metric is used to detect the first symbol: Let r_m be the complex valued received symbols and L the number of samples in one half of the first training symbol. Then the sum of the pairs of products is

$$P(d) = \sum_{m=0}^{L-1} (r^*_{d+m} r_{d+m+L}). \tag{2.26}$$

The received energy for the second half of the training symbol is defined by

$$R(d) = \sum_{m=0}^{L-1} |r_{d+m+L}|^2. \tag{2.27}$$

Using $P(d)$ and $R(d)$ a timing metric is defined as

$$M(d) = \frac{|P(d)|^2}{R(d)^2}. \tag{2.28}$$

The maximum value the metric reaches depends on the SNR of the channel and the OFDM symbol length [18]. Synchronization is assumed once the metric grows larger than a fixed threshold level a. The threshold has to be selected in a way that neither the rate of sequence misses, nor the rate of false positives is too high.

A carrier frequency mismatch in the system causes the original baseband samples s_m to be rotated by a phase step f_e.

$$r_m = s_m e^{j2\pi f_e m}. \tag{2.29}$$

If f_e is known, the receiver can compensate the offset by rotating all time domain samples back. As previously said, we get an estimate of f_e by correlating the two identical halves of the training symbol

$$\hat{f}_e = \frac{angle(P(d))}{\pi T} \tag{2.30}$$

since there is a phase difference of $\pi T f_e$ between the two halves.

2.5.2 Channel estimation and equalization

One reason why OFDM has been widely adopted by modern communication standards is the simplified channel estimation for wideband signals. In wideband single carrier communication, the channel impulse response consists of many taps, since the symbol duration is very short compared to the delay spread. The channel is classified as frequency selective fading. In an OFDM system, every subchannel can be estimated independently. Thus, it is sufficient estimate multiple narrow subchannels instead. Each subchannel undergoes flat fading, since it is so narrow. The channel impulse response of these subchannels can be described by a single tap, i.e. only the amplitude and phase of the channel has to be estimated [19]. To get an estimation of each subchannel, typically pilot symbols are added to the signal. These are previously known symbols, which the receiver uses to estimate the channel response. Each subchannel can then be compensated by rotating and scaling all samples against the amplitude and phase of the subchannel.

3

System Design

In this chapter we describe the design process of the communication system. At first we collect the legal requirements that we have to follow. We then determine requirements for the HAMNET access network based on its use-cases. Next, we survey solutions that have been attempted so far and check how they match our derived requirements. We also survey well known waveforms, such as LTE, that have been designed for other use-cases. Here, we check which modifications are needed to deploy them as a HAMNET access network.

Next, we design the new communication system. We start with selecting proper channel models and use them to estimate the link conditions. The results are the basis to decide on the physical layer implementation. Finally, we use the system use-cases to choose a proper MAC protocol and to develop an interface to the IP-stack.

3.1 System Requirements Analysis

Defining clear system requirements is the first important part of the system design phase. The requirements give us the foundation to decide what system components shall be used.

3.1.1 Legal Regulations

At first we have to consider legal regulations. As pointed out in the introduction, we want to use the $70\,\mathrm{cm}$ band to get access to the existing HAMNET. The $430 - 440\,\mathrm{MHz}$ band is dedicated along with other transmissions for amateur radio usage, as stated in the frequency plan of the German Bundesnetzagentur (Federal Network Agency) [20]. The basestation of the access network has to be considered as an autonomously operating amateur radio station (Automatisch arbeitende Amateurfunkstelle). As stated in the German law on amateur radio operation §6 sentence 1 [21], the Federal Network Agency licenses autonomously operating stations. They are doing so in coordination with the German Amateur Radio Club (DARC). The DARC planned the operation of autonomous stations with a maximum bandwidth of $200\,\mathrm{kHz}$ in frequency division duplex mode with center frequencies $434.9\,\mathrm{MHz}$ and $439.7\,\mathrm{MHz}$ [22]. These are the frequencies we are going to use. Additionally, the maximum transmission power is limited to $15\ \mathrm{W}$ ERP[1] for autonomous stations as stated in annex 1 of the Amateurfunkverordnung AFuV ("amateur radio directive") [23]. Clients, on the other hand, can send with up to $75 - 750\ \mathrm{W}$ ERP depending on the amateur radio license class. As it does not seem useful to have such a large difference in the transmission powers for uplink and downlink, we set the requirement to support up to $15\ \mathrm{W}$ ERP for

[1] ERP is a dipole based power measurement. $EIRP = ERP \cdot 1.64$

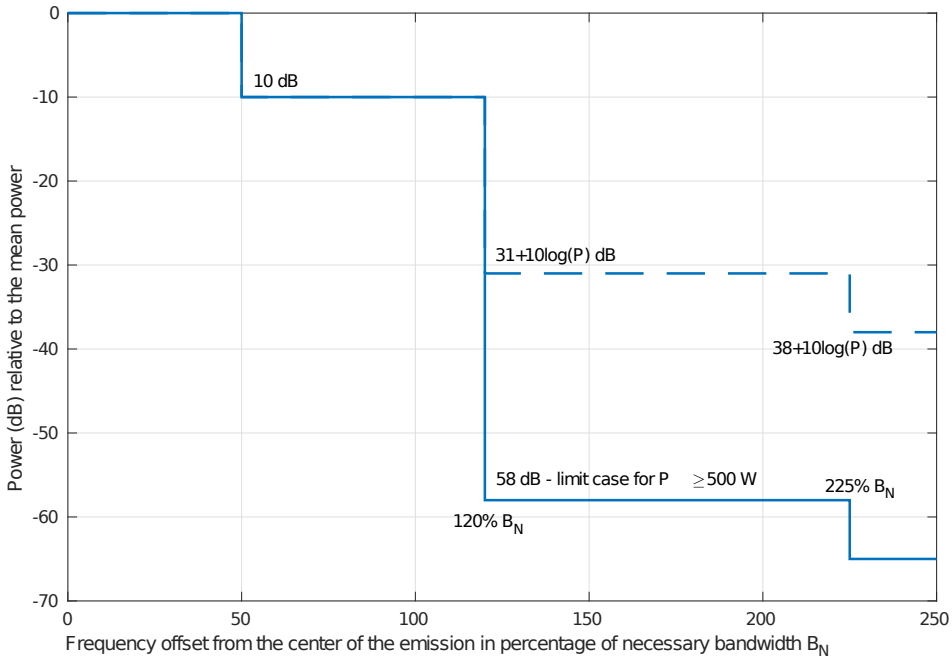


FIGURE 3.1: Spectral mask according to ITU-R SM.1541 for stations operating above 30MHz in the normal band case. Taken from [24]

both basestation and client hardware. Another legal requirement is the out of band spectrum of the developed waveform. ITU Recommendation SM.1541 Annex 9 [24] gives a spectral mask for out of band emissions. According to ITU-R SM.1539 our waveform with the necessary bandwidth $B_N = 200\ \mathrm{kHz}$ should use the normal case mask. The mask is given in Figure 3.1. From this we can derive a required damping of $10\ \mathrm{dB}$ above $100\ \mathrm{kHz}$ from our carrier frequency. Using $P = 15\ \mathrm{W}$, the out-of-band power above $240\ \mathrm{kHz}$ shall be limited to $-43\ \mathrm{dBc}$, above $450\ \mathrm{kHz}$ it shall not exceed $-50\ \mathrm{dBc}$. At last, every client and basestation has to transmit their callsign, an identifier given by the Federal Network Agency. The callsign uniquely identifies a amateur radio operator and autonomous amateur radio stations and it has to be regularly announced in the used frequency band. More precisely, each user has to send the callsign after establishing a connection, every 10 minutes during operation and when ending a connection (AFuV §11 sentence 1 [23]).

3.1.2 Use-case derived requirements

After identifying legal requirements, we can continue with softer requirements that come from desired use-cases. Most important is the support for IPv4 traffic. Since our system shall provide access to the HAMNET backbone which is based on IP traffic this requirement is indisputable. Also fixed is the topology of the network. Since we have a frequency division duplex band and want to give access to the HAMNET backbone,

the network topology is a star with the central basestation node being the coordinator of the network.

The first desired use-case for the Network is Voice over IP (VoIP). Analog and digital voice telephony have been common applications in the amateur radio spectrum, so it would be good if we would be able to run this application over the IP stack. This imposes requirements on the maximum delay. The one way full stack delay should not exceed 150 ms, up to 400 ms is acceptable for long distance calls. The larger the delay gets, the harder it is to communicate with each other without interfering each other [25]. The full stack delay includes packetization time at the sender, which is usually ca. 20 ms. Further delay is introduced by a jitter buffer at the receiver. This buffer is used to capture out of order packets caused by variance in the transmission delay. In accordance with Goode [25] we estimate the jitter buffer as 60 ms. This leaves 70 ms for Network layer delay (we discard processing delays due to encoding and decoding) for a good quality VoIP call. Furthermore, the jitter of our system should not exceed the jitter buffer size, the jitter is thus limited to 60 ms.

Another aspect that should be considered is the bandwidth requirement for a VoIP call. The bitrate of Voice codecs ranges from 5.3 kbps (G.723.1) to 64 kbps (G.711) [26]. Significant overhead is added by the headers of the lower layers: The Real-Time Transport (RTP) Header (12 bytes), User Datagram Protocol (UDP) Header (8 bytes) and the IP Header (20 bytes). Taking the 8 kbps G.729 codec with a packetization interval of 20 ms this results in a bandwidth of:

$$B_{G.729} = \frac{B_{codec} \cdot T_{packetization} + N_{RTP} + N_{UDP} + N_{IP}}{T_{packetization}} = 24 \text{ kbps}. \tag{3.1}$$

The size of a VoIP packet including all headers is:

$$N_{VoIP-packet} = B_{codec} \cdot T_{packetization} + N_{RTP} + N_{UDP} + N_{IP} = 60 \text{ Bytes}. \tag{3.2}$$

Next, one single basestation should be able to give multiple users access to the network. We do not focus on multiple users being able to transmit at the same time. Instead, at least 10 users should be able to idle in the network and get transmission resources within a reasonable amount of time if requested. The behavior should be similar to mobile networks, where cell phones can idle at a basestation.

Another requirement is the supported link distance. It is hard to set fixed values here, since the possible link distance greatly depends on the terrain and the user's antenna. A link budget analysis in the next section shall give more insights on what is possible, so far we only require the system to support long range but slow links. Adding to this, the user should get better data-rates if the link quality is good, the system should support adaptive modulation.

We focus on stationary operation of the system. We assume that the system will be mostly deployed to connect autonomous stations and the homes of radio amateurs. Due to this assumption, the system is allowed to only have limited robustness against Doppler shifts, as these only occur with moving stations.

Furthermore, we restrict the cost of the hardware. The necessary user equipment costs should not be too high, since deploying the system is only a hobby of radio amateurs. A basestation however, could require more expensive hardware. It is a central part of the system that does not need to be deployed as often as clients.

These cost restrictions have implications on the available duplex operation modes. For full duplex mode, e.g. sending and receiving at the same time, a duplexer that separates the uplink and downlink with their 4.8 MHz shift needs to be installed after the antenna. In order to reduce hardware costs of the clients, we omit this duplexer and restrict clients to half-duplex transmission.

One last use-case is the flexibility or portability of the system. The amateur radio bands are regulated differently in other countries, e.g. the usable frequencies within the 70 cm band can vary. But also the maximum bandwidth could be different. It would be good if it required not too much modifications to deploy the system with different frequencies and bandwidths. This feature is also desired, since it raises the acceptance of the system.

3.2 SURVEY OF EXISTING WAVEFORMS

After some requirements have been defined, we will survey existing proposed solutions. Most of these solutions are only idea sketches on how to modify existing waveforms. Notably different is the project "New Packet Radio" by the French radio amateur F4HDK [27], which offers a ready to use solution.

3.2.1 New Packet Radio

New Packet Radio is an open-source project which was first released in April 2019 [27]. It uses an ISM transceiver chip that operates on the 70 cm ISM band using 2 or 4 state Gaussian phase shift keying. Multiplexing was initially done with a TDD TDMA scheme, support for FDD TDMA has been added. It can be deployed with bandwidths ranging from 100 kHz to 1MHz, our desired bandwidth of 200 kHz is also supported. According to spectral power measurements, the signal fulfills the requirements of the ITU spectral mask described in the previous section [27]. The transceiver is connected to an amplifier with an output of up to 20 W, so transmit powers of up to the legal limit are possible. As per his own statement, link distances of up to 80km were tested [28]. However, the channel conditions and the used antennas have not been stated.

So far, New Packet Radio fulfills the legal requirements we have derived. Next, we will use the protocol specification to assess whether the other requirements are met. In the 200 kHz mode using 4GFSK (modulation 22 in the specification) the duration of one TDMA frame is 294.5 ms. Within a TDMA frame up to 16 so called micro slots can be used for data transmission. The maximum payload size that can be transmitted in a microslot is 255 Bytes. Thus, the maximum Layer 1 data-rate is

$$R_{max} = \frac{255Byte \cdot 16}{0.2945s} = 111 \text{ kbps}. \tag{3.3}$$

This data-rate would be sufficient for multiple 24kbps VoIP streams. To further identify the VoIP capabilities of the system we have to estimate the transmission delays. At first, there are two allocation modes for each client: A fast mode and a slow mode. The slow mode is only entered if a client requests at most one uplink slot within 32 TDMA frames (~ 9.4 seconds), so we can assume that a client will always be in the fast mode during a VoIP call. If only a single user is connected to the basestation, the basestation will allocate all micro slots in uplink and downlink to this client. Each

micro slot can contain up to 255 Bytes payload, which is enough to fit in a VoIP packet of $N_{VoIP-packet} = 60$ Bytes. A micro slot is sent every 16.6 ms, so the micro slot rate is higher than the VoIP packetization rate of 20 ms. Under the assumption that the radio frame, which is transmitted during a micro slot, is scheduled just before the micro slot, the delay of the VoIP communication should be at most one micro slot duration. This is an optimistic estimation, as it neglects processing delays. They could be significant on the low power micro-controller hardware. But, as we want to assess the capabilities of the protocol itself, it is a valid simplification. To conclude, a single connected client in fast mode might be able to support VoIP traffic. This changes, once more than one client is connected in fast mode. The micro slot assignment for each user has to be continuous. This creates large timeslots in which no transmission is possible. If two clients get assigned the same amount of slots, the time in which a client cannot transmit is 150 ms (50% of the 294 ms frame duration). The delay is even larger for 4 connected clients, which could support VoIP if only the bandwidth limitation is considered. Taking these delay estimations into account, it is unlikely that the system will deliver a good VoIP performance in a real world scenario. It seems that the design of the TDMA structure was done in a way to maximize the throughput for non delay-critical applications. Thus, the size of one slot is quite large, as is the TDMA frame cycle.

In conclusion, the system meets most of the key requirements, but it is questionable whether it will able to deliver acceptable VoIP performance.

3.2.2 LTE

Long Term Evolution is a cellular network standard released by the 3GPP. The first LTE Release, 3GPP Release 8, was published in 2008 [15]. Later releases offer data-rates of more than 1000 Mbps and a high spectral efficiency. Given the popularity, we had a look whether LTE could be operated on the available 200 kHz channel. The smallest bandwidth configuration of LTE is 1.4MHz, still too large for the available resources. Though, by slowing down the symbol rate of the 1.4 MHz LTE we could narrow the bandwidth even further. If we divide the symbol-rate by 8, this would result in a bandwidth of only 175 kHz, which fulfills our system requirements. However, LTE is a OFDM system with designed subcarrier spacing of 15 kHz. Slowing down the symbol rate would narrow the subcarrier spacing to less than 2 kHz. As described in the previous chapter, such narrow spacings require a highly sophisticated frequency offset estimation and correction. In addition, the impairments due to the Doppler spread are more severe. Thus, such an approach does not seem feasible. Nevertheless notable is the delay of an LTE system. LTE gives a delay of 5ms for user data transmission [15]. This is achieved by using very short scheduling intervals. In the LTE frame structure, one frame has a duration of 10 ms. It consists of 10 subframes, 1ms each, that are split up into a control and a data region. The scheduler is executed on a per subframe basis [29].

In LTE Release 13, a new cellular technology called Narrowband Internet of Things (NB-IoT) was introduced [30]. It was designed for machine to machine communication, thus the required client complexity is low. As such is the data-rate. Interesting for us is the system bandwidth of only 180 kHz, which fits our requirements. Furthermore, NB-IoT allows clients to operate in half duplex mode. Layer 1 peak data-

rates are 250 kbps in uplink and 226.7 kbps in downlink [30]. However, the latency of the system could become a serious issue. In machine to machine communication, the assumed payload data will mostly be sensor measurement reports. Instead of focusing on latency, main design aspects of such systems are power consumption and transceiver complexity. Most papers that assess the latency of NB-IoT, thus assume a regular measurement report as payload, but no streaming applications. In [31], the latency of a measurement report consisting of 80 Bytes transmitted with 160 bps has been calculated. According to the authors, the latency will not exceed 6.6 seconds with 99% confidence. This latency includes synchronization, random access and possible retransmissions, delays that we do not have to consider during streaming. Furthermore, the chosen data-rate is very low, giving a high delay for the actual transmission of the packet. But even if these latencies are omitted, the remaining latency is $\sim 500\,\mathrm{ms}$, and thus most likely too high to support VoIP applications.

3.2.3 Other approaches

For completeness of the survey, we will briefly introduce into two other projects that have been noticed during the initial research. The project *HRD70* by A. Müller and S. Brigmeier was presented at the amateur radio convention in 2018 in Munich [32]. They use a GFSK transceiver similar to *New Packet Radio* and a TDMA based MAC protocol. To this date, no further updates of the project have been found. There are no sources or detailed protocol specifications available which we could analyze. Finally, the project *Charon* uses the Adalm Pluto, a Software Defined Radio (SDR) by Analog Devices. It makes use of the Opensource library liquid-dsp [33] and implements a OFDM based transceiver. The channel bandwidth is 140 kHz and 64 subcarriers are used. This results in a very narrow subcarrier spacing of about 2 kHz. The subcarriers can be modulated using 16-QAM, this leads to a data rate of 272kbps after FEC coding. CSMA is used for medium access control [34]. Thus, delays are expected to be low if the number of users is limited and system load is low. However this will change, if multiple users want to transmit at the same time. Then, CSMA suffers from a lower throughput efficiency and probably higher delays than a collision free MAC protocol. Furthermore, only one out of two available FDD channels is used in this project. Nevertheless, the project is interesting regarding the PHY layer and hardware selection. By using a SDR, higher order modulation schemes can be used, giving us significantly better data-rates than in the project New Packet Radio.

3.3 CHANNEL MODELING

In the following section, we estimate the channel properties for scenarios in which the designed system will be mostly used. At first, we derive the large scale fading parameter using the Hata model that we presented previously in chapter 2.2.1. The path loss estimate will be used to perform a link budget analysis and check possible modulation schemes.

Path Loss in urban areas				
	$h_{re_1} = 1.5$ m		$h_{re_2} = 10$ m	
	$h_{te_1} = 100$ m	$h_{te_2} = 200$ m	$h_{te_1} = 100$ m	$h_{te_2} = 200$ m
5 km	133.3 dB	127.8 dB	114.5 dB	109 dB
10 km	142.9 dB	136.7 dB	124.1 dB	118 dB
30 km	158 dB	151 dB	139.3 dB	132.2 dB

TABLE 3.1: Path loss estimation using the Hata model for urban terrain

Path Loss in suburban areas				
	$h_{re_1} = 1.5$ m		$h_{re_2} = 10$ m	
	$h_{te_1} = 100$ m	$h_{te_2} = 200$ m	$h_{te_1} = 100$ m	$h_{te_2} = 200$ m
5 km	125 dB	119.5 dB	106.3 dB	100.7 dB
10 km	134.6 dB	128.5 dB	115.8 dB	110 dB
30 km	149.8 dB	142.7 dB	131 dB	123.9 dB

TABLE 3.2: Path loss estimation using the Hata model for suburban terrain

3.3.1 Path Loss and Link Budget Analysis

The Hata model distinguishes four different terrain types: Urban, Suburban, Rural and Hilly terrain. We limit our analysis to the Urban and Suburban scenario, as these seem the most common scenarios for a 70 cm HAMNET site. Common location for HAMNET backbone stations are elevated buildings in a city. The basestation of the 70 cm system will be mounted next to existing HAMNET backbone stations. Client antennas will be either mounted on top of houses or be mounted close to windows. According to the HAMNET database [1], 20 backbone stations in Germany are mounted on buildings higher than 150m. 45 stations have heights between 100 m and 150 m and 65 stations are mounted at heights 75 m to 100 m. Therefore we consider the effective antenna height $h_{te_1} = 100$ m and $h_{te_2} = 200$ m. 200 m was chosen since the stations in Munich and Nuremberg have this height and they are most likely to be chosen for first real world tests. The distance is set to 5 km, 10 km, and 30 km. The first two represent typical distances that we estimated to occur most common, 30 km represents a usage scenario in a more rural area.

The receiver antenna height is assumed to be either $h_{re_1} = 1.5$ m or $h_{re_2} = 10$ m above ground. The first case will be likely in urban areas, where amateur radio operators mount antennas in their flat probably in ground floor close to the window. The second case represents operators which mount their antennas on top of their roof.
Using the path loss we can now perform a link budget analysis. For this, we calculate the received signal level at the receiver and the receiver threshold. The signal level at the receiver is given by [4]:

$$RSL = EIRP - L_{path} + G_{Rx}. \tag{3.4}$$

SNR at receiver elevation 1.5m		
link distance	$h_{te_1} = 100\ \mathrm{m}$	$h_{te_2} = 200\ \mathrm{m}$
5 km (urban)	29.6 dB	35.1 dB
10 km (urban)	20 dB	26.2 dB
30 km (urban)	4.9 dB	11.9dB
5 km (suburban)	37.9 dB	43.4 dB
10 km (suburban)	28.3 dB	34.4 dB
30 km (suburban)	13.1 dB	20.2 dB

SNR at receiver elevation 10m		
link distance	$h_{te_1} = 100\,\mathrm{m}$	$h_{te_2} = 200\,\mathrm{m}$
30 km (urban)	23.6 dB	30.7 dB
30 km (suburban)	31.9 dB	39 dB

TABLE 3.3: SNR estimates at the receiver using the Hata model, a receiver threshold of $-114\,\mathrm{dBm}$, transmit power $EIRP = 43.9\,\mathrm{dBm}$, RX antenna gain $G_{RX} = 5\,\mathrm{dB}$

The Effective Radiated Power ERP of the base station is legally restricted to 15W (41.8dBm). The Effective isotropic Radiated Power (EIRP) can be obtained by the following equation: $EIRP = 1.64 \cdot ERP$, thus $EIRP_{max} = 43.9\ \mathrm{dBm}$. The receiver threshold is equal to the thermal noise plus the noise figure L_{nf} of the receiver

$$RX_{thresh} = k_B BT + L_{nf}. \tag{3.5}$$

This is the intrinsic noise power of the receiver. In order to detect a signal, the received signal strength must be above this threshold. As our channel has a bandwidth of 200 kHz, the thermal noise power equals

$$N = k_B BT = 1.38 * 10^{-23}\ \mathrm{J/K} \cdot 200\,\mathrm{kHz} \cdot 290\,\mathrm{K} = -121\,\mathrm{dBm}. \tag{3.6}$$

The noise figure of an Adalm Pluto SDR is stated as $L_{nf} = 2.5dB$ by the manufacturer [35]. Literature on platform independent noise figures suggests to use significantly larger values. For example, a technical report by ETSI on parameters affecting the receiver signal level threshold, recommends to use 4dB + 3dB industrial margin for carrier frequencies above $1.5\,\mathrm{GHz}$ [36]. Suggestions for the $70\,\mathrm{cm}$ band could not be found. In order to get a conservative estimation, we will assume a noise figure of $L_{nf} = 7\ \mathrm{dB}$.

Subtracting the receiver threshold from the received signal level gives us the SNR at receiver:

$$SNR_{receiver} = RSL - RX_{thresh}. \tag{3.7}$$

The SNR at the receiver has been calculated for different path loss scenarios and is shown in Table 3.3. In order to transmit with a desired bit error rate (BER) over the channel, a certain SNR depending on the modulation scheme must be reached. For successful transmission the SNR at the receiver must be larger than the required SNR. The required SNR for a bit error rate of 10^{-6} over an AWGN channel was taken

Modulation Type	BPSK	QPSK	QAM16	QAM64	QAM256
SNR for BER=10^{-6}	10.5 dB	13.5dB	18dB	25.3dB	33dB

TABLE 3.4: Required SNR in order to reach a BER of 10^{-6} over an AWGN channel

Measurement	T_1 (d=1km)	d=5km	d=10km	d=30km
Rappaport, 4US cities Urban	0.94us	2.1us	2.97us	5.15us
Weyman, Denver Urban	0.41us	0.92us	1.30us	2.25us
Sousa, Toronto Suburban	0.28us	0.63us	0.89us	1.53us

TABLE 3.5: Median delay spreads using T_1 value from [10]

from [5] and is shown in Table 3.4. These values represent bit error rates of uncoded modulation. For better performance a coding scheme should be used. We will discuss about this later in this thesis. The bit error rates of the uncoded modulation can be seen as a worst case estimate.

Using Table 3.3 and Table 3.4 we can see that our received signal levels are high enough to at least support QPSK modulation, except for the 30km link distance urban scenario with bad RX antenna setup. In many cases the SNR will be high enough to allow high order modulations, especially if the RX antenna is mounted on top of a roof ($h_{re} = 10\,\mathrm{m}$).

3.3.2 Multipath components

After we characterized possible link distances using large scale path loss we will continue with the multipath parameters. At first, we will gather delay spread estimations by using Greensteins model. The delay spread is modeled using Equation 2.16

$$\tau_{rms}(d) = T_1 \cdot d^{\epsilon}, \tag{3.8}$$

where ϵ is 0.5 in all environments except Mountainous terrain. In the original paper, Greenstein analyzed measurements collected in different cities and calculated T_1 for them. The results are given in Table 3.5. The table also gives delay spread estimations at different distances when applying the formula for $\epsilon = 0.5$. The calculated delay spread is now going to be used to estimate the coherence bandwidth. When using $B_c = \frac{1}{5\tau_{rms}}$, our channel with bandwidth $B = 200\,\mathrm{kHz}$ can be seen as flat, if the link distance is short ($d < 5\,\mathrm{km}$). Using the more strict approximation $B_c = \frac{1}{50\tau_{rms}}$, we have to expect a frequency selective channel already for short distances of $1\,\mathrm{km}$. Our use-case scenario is thus an edge case. If we consider clients with directional antennas mounted on a roof, the delay spread will most likely be short enough to consider the channel as flat. However, in long distance cases without these best case antenna configurations, the channel could be frequency selective.

Using the formula $f_m = \frac{v}{\lambda}$, the maximum Doppler shift in the 70 cm band is $f_m = 20\,\mathrm{Hz}$ at a velocity v of $50\,\mathrm{km/h}$ or $f_m = 40\,\mathrm{Hz}$ at a velocity of $100\,\mathrm{km/h}$. If we look back at our requirements, one might note that we do not require to support moving nodes. However, since Doppler spread is not only caused if the receiver itself is moving, but also by moving objects in the channel [5], we cannot omit these calculations. Using the expected Doppler shift, the coherence time of the channel can be estimated. At $50\,\mathrm{km/h}$ it is $T_c = \frac{0.423}{f_{dmax}} = 21\,\mathrm{ms}$ and at $100\,\mathrm{km/h}$ it is $T_c = 10.5\,\mathrm{ms}$ [5]. The symbol time $1/B$ is short compared to estimated coherence time. Hence, the channel undergoes slow fading, it can be perceived as constant for a relatively long time. We should include channel estimation symbols at least every $10\mathrm{ms}$. These might be training sequences or pilot symbols, depending on the chosen physical layer.

3.4 PHY LAYER DESIGN

After we derived the system requirements and gathered some channel metrics we start developing our own system. Before we go to the PHY and MAC layer design phase, we will shortly discuss which hardware can be used. From the survey of existing waveforms, we know that there are many cheap FSK and MSK transceivers available for the 70 cm band. However, we also saw that the spectral efficiency of these devices is low, they support at most 4 level modulation. Better spectral efficiency can be achieved with SDRs and M-QAM or M-PSK modulation, as we saw in project Charon. Another big advantage of SDRs is that all signal processing can be done on a general purpose processor (GPP). Therefore, the system design can be done without specifying one SDR model, also the implementation is at least partially platform independent.

3.4.1 Modulation and Coding

We can expect a high SNR in many scenarios and want to make use of this by getting high data rates from our limited channel bandwidth. Therefore the system needs to support spectrally efficient modulation schemes. But also more robust schemes in case of large link distances where the SNR is low are required. Two candidates are M-QAM and M-PSK modulation.

In M-PSK modulation, M constellation points are mapped equidistant on the complex unit circle. In M-QAM, both the amplitude and phase are modulated, the constellation points represent a grid on the complex plane. Possible downsides of QAM are the modulated amplitude, which makes QAM unusable in fast fading channels and requires more linear amplifiers [4]. PSK on the other side has a worse bit error rate (BER) given the same average symbol energy [3]. As our channel undergoes slow fading and we need linear amplifiers anyway for the OFDM modulation (see next section), we will use QAM. In order to support robust long distance links, QAM-4 or QPSK is the basic modulation scheme that every client has to support. On top of that, QAM-16 and QAM-64 are defined as optional modulation schemes for our adaptive modulation.

Next, the coding scheme has to be decided. Coding schemes were assessed by looking at the BER performance and the computational complexity. In general, coding schemes can be classified into block codes and convolutional codes. In block codes,

binary sequences of length n are mapped to binary sequences of length m, where $m > n$. This adds redundancy to the information being sent and allows error correction. The code rate $r = n/m$ indicates how much redundancy is added. For convolutional codes, n bits enter a finite state machine at a time, resulting in an output of m coded bits. The current state of the machine depends on the previous input bits, the machine has a memory. The length of the memory is called the constraint length k. The memory property of convolutional codes is the reason that they can achieve better performance compared to block codes at the same coding rate. Decoding performance can be further increased by using soft decision demodulation before the decoder. Here, the demodulator does not make a hard decision on the sent bits. Instead, every decoded bit is given as a multi-level certainty value. For a 8 level soft decision, the value 0 could represent a certain zero, value 7 represents a certain 1. The values in between represent uncertainties in the decision. However, the soft decision decoding requires more computational resources.

In the last years, *turbo codes* became a popular choice to use as a coding scheme. They are used for example in LTE [30]. Turbo codes are a special type of convolutional codes, where two different convolutional codes are concatenated. They are decoded using iterative decoding. The performance of turbo codes is close to the theoretical shannon limit, if the number of iterations is sufficiently high [37]. However, the iterative decoding process is computationally very expensive. With our chosen SDR, we do have enough resources for iterative decoding. Therefore, these codes will not be considered further. Instead, we will use convolutional codes with relatively short constraint length ($k = 7$). If feasible, decoding will be performed by soft decision Viterbi. Using [3], we get the upper bound of the coding gain for a $r = 1/2$ code as $5.44\,\mathrm{dB}$. Longer constraint lengths result in higher coding gain, but the computational complexity also rises fast [3]. We have seen from the SNR estimation, that a SNR of up to $30\,\mathrm{dB}$ can be expected. We do not need robust coding schemes in such scenarios. To achieve higher data rates, convolutional codes with high rates ($3/4$, $5/6$ etc.) could be used. However, such codes result in very complex Viterbi decoders, which are infeasible to use. Instead code puncturing is commonly used to trade in coding gain for higher data-rates. In code puncturing, our code for $r = 1/2$ is used but certain bits at the output of the decoder are removed. This gives the desired higher coding rates of $2/3$, $3/4$ etc. Having chosen possible modulation and coding schemes, a fixed set of such combinations, so called modulation coding schemes (MCS) will be defined. Client and basestation are able to to agree on a MCS during runtime, enabling adaptive modulation. Table 3.6 shows the MCS definitions.

In wireless communication, bit errors are typically not uniformly distributed. Since the received signal power can be modeled as Rayleigh distributed, we experience short deep fades. They result in multiple biterrors within a short time, called burst errors. Coding schemes however are designed to effectively correct statistically independent bit errors. They are not able to correct such burst errors. In order to cope with this problem, an interleaver will be used. An interleaver reorders blocks of incoming bits and forwards them to the encoder. This way the encoder does not directly work on the bitstream that is sent out in time domain, but on a more randomized one.

MCS idx	modulation	coding rate	logical channel size
0	QPSK	1/2	62
1	QPSK	3/4	93
2	16QAM	1/2	125
3	16QAM	3/4	187
4	64QAM	1/2	188
5	64QAM	3/4	282
6	256QAM	1/2	250

TABLE 3.6: MCS definitions and the resulting payload size in Bytes.

3.4.2 OFDM

As we saw during channel modeling, there are cases in which the channel has to be perceived as frequency selective. Also ISI will occur. Using OFDM would yield us multiple flat subchannels, and ISI could be handled by using a cyclic prefix. This makes OFDM look like a promising candidate to use on the PHY layer. In the following we will have a closer look what advantages OFDM will bring and which problems could arise when using it.

As already pointed out, the first benefit is the robustness against ISI. If we set our symbol rate somewhat near $200\,\mathrm{kSym/s}$, the symbol duration will be $5\,\mu\mathrm{s}$. A cyclic prefix of 2 or 4 symbols should be enough to overcome any ISI. The deployment of an OFDM system would be more flexible than the deployment of a singlecarrier system. One requirement was a flexible bandwidth deployment. Using an OFDM system will give us a very flexible deployment. If we want to narrow the bandwidth of the channel, we can easily disable subchannels at the outer edges of the spectrum. By assigning zeros to them no energy will be put to the corresponding frequencies. Such subchannels (or subcarriers) are commonly known as guard subcarriers. On the other hand, if the system shall be deployed on a wider channel, the total numbers of subcarriers can be increased by increasing the symbol rate and FFT size at the same time. Then, the width of each subcarrier will stay constant and therefore also the physical channel properties. Thus, the same channel equalization technique can be used for different bandwidth deployments. Due to the narrow subchannels, there will also be no problem to deploy the system at higher carrier frequencies, whereas a single carrier system will have problems coping with the more severe frequency selectivity. The channel equalization will be simpler in some cases. As every subchannel is flat, it is enough to calculate just one complex coefficient to equalize the channel. Although, also in a single carrier system, an equalizer with just a few filter taps would be required. ISI will span only over a few symbols, the complexity for e.g. a least mean squares equalizer will not be too high. Again, since we want to keep the bandwidth flexible, an OFDM equalizer would help us here. At last, OFDMA could be used. OFDMA is a multiple access scheme based on OFDM. The users are assigned different subcarriers in the uplink, which allows them to transmit at the same time. This could enable low delays, which we require in order to support VoIP.

On the downside, OFDM requires a more accurate carrier frequency offset estimation, since the channel bandwidths are smaller compared to a single carrier system. Another downside is the higher peak to average power ratio (PAPR). This requires

amplifiers with a large linear working range. This can be either achieved by using more costly amplifiers, or lowering the transmit power and using amplifiers at a lower but typically more linear output level. The PAPR in OFDM systems is proportional to $\sqrt{N}$, where N is the number of subcarriers [3]. Furthermore, OFDM is more sensitive to Doppler spread. The Doppler shift imposes a uncorrectable frequency offset on our system. This does not impact the system performance if the bandwidth is large compared to the Doppler shift. But since narrow subcarriers are used, Doppler shifts can severley degrade system performance in OFDM systems. Since the subcarriers have a rather small bandwidth, the symbol time is longer compared to single carrier systems. Therefore, we should check whether we can still expect a slow fading channel. The previously estimated coherence time of $T_c = 10.5\,\mathrm{ms}$ will only give us a fast fading channel for very narrow subcarrier spacings of less than $1\,\mathrm{kHz}$. At last, OFDM has higher out of band emissions compared to many single carrier systems. In single carrier systems, pulse shaping is used to increase spectral efficiency. This is done by low-pass filtering the signal at the transmitter. A matching filter at the receiver can restore the original signal. This is not possible in OFDM. What is used in OFDM instead is the concept of guard bands. The outermost subcarriers are modulated with zeros, resulting in spectral nulls at these carriers.

After all, we decided to stick to an OFDM system. The flexibility and simplicity of the channel equalization outweigh the downsides, which can mostly be mitigated by putting more effort into the signal processing. We chose $N_{FFT} = 64$ subcarriers and a subcarrier spacing of $B_{SC} = 4\,\mathrm{kHz}$. The subcarrier spacing is wide enough to accurately estimate the frequency offset. At the same time, the FFT size is large enough that adding a cyclic prefix does not affect the usable bandwidth too much. We will add a cyclic prefix of length $L_{CP} = 4$ samples. The symbol duration of the system is $T_s = \frac{1}{N_{FFT} * B_{SC}} = 3.90625\,\mu\mathrm{s}$. Therefore, all ISI caused by delayed multipath of up to $15.6\,\mu\mathrm{s}$ will be canceled. Of all 64 subcarriers, 40 will be modulated with data or pilots, the other carriers are used as nullified guard bands. This gives a usable bandwidth of $160\,\mathrm{kHz}$. The resulting spectrum is given in Figure 3.2 and compared to the ITU spectral mask. The figure only displays the baseband signal up to a frequency $1/T_s$, we will design digital filters that ensure that the interpolated and upmixed signal also fulfills the spectral mask (see chapter 4.1.1).

At last we will define the pilot distribution in a OFDM symbol. A straight-forward method is a equidistant distribution of pilots across all subcarriers. We additionally define two types of OFDM symbols, symbols containing pilots and OFDM symbols without any pilots. The two subcarrier allocation for the two types are shown in Figure 3.3. Furthermore, by defining an allocation of the introduced OFDM symbol types in the time domain, we get a 2D grid over time and frequency in which pilot tones can be defined. In the time domain, the delay between to pilot tones should be shorter than the channel coherence time. We estimated the coherence time T_c as $10\,\mathrm{ms}$, thus pilot tones in every 10th OFDM symbol should be sufficient. However, first tests showed us that channel estimation would not work properly, if the timespan between to pilot tones is this large. We therefore designed the 2D allocation grid to contain a pilot tone in every second OFDM symbol. Figure 3.4 shows the grid for a PHY layer data slot.

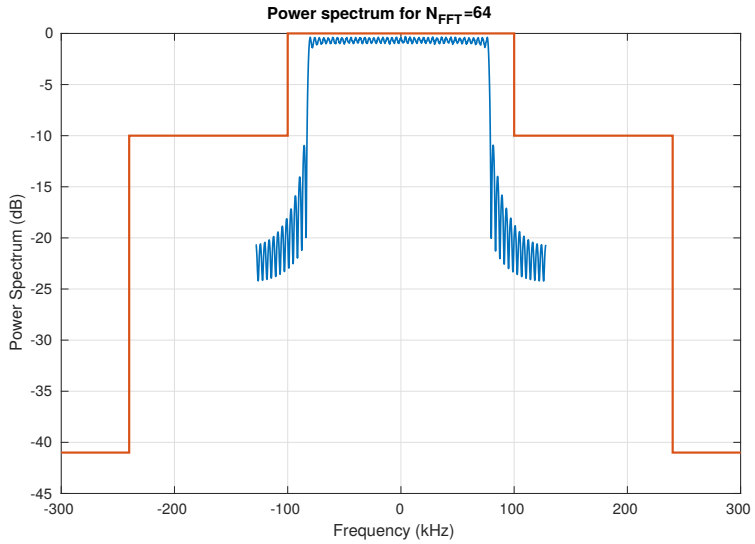


FIGURE 3.2: OFDM power spectrum with FFT size 64, 4 kHz subcarrier spacing and 40 used data carriers

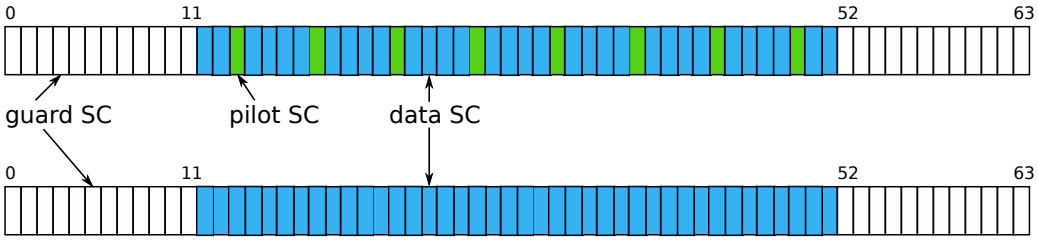


FIGURE 3.3: OFDM subcarrier allocation including pilot symbols (top) and without any pilot symbols (bottom).

3.5 MEDIUM ACCESS CONTROL

For medium access, a centralized hybrid TDMA protocol will be designed. After surveying different MAC protocol types in Section 2.4, we concluded that a pure random access scheme will not be sufficient for our needs. The throughput efficiency of ALOHA or CSMA is just too low in a high load scenario. Since our resources are limited to $200\,\mathrm{kHz}$ channels, the goal is to find a protocol with at best close to one throughput. CSMA and ALOHA achieve a throughput of at most 0.5 at maximum load $G = 1$ (see Fig. 2.5). In addition, a centralized TDMA protocol will result in a more deterministic latency, with a low jitter. For VoIP applications, users have to be granted transmission possibilities on a regular basis. This cannot be guaranteed when using a random access MAC like CSMA. We designed a hybrid access protocol, in which a limited number of user gets guaranteed access and new users can try to get into the guaranteed access pool by using a random access channel. The random access scheme is based on slotted ALOHA.

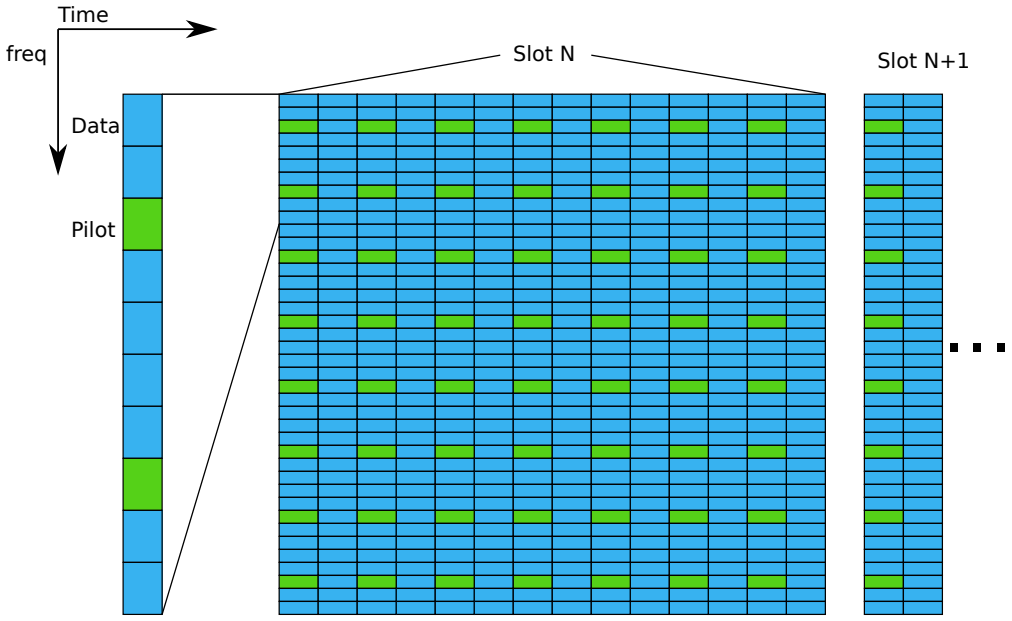


FIGURE 3.4: The resulting OFDM pilot allocation grid for a data slot.

Next, the multiplexing technique that is used has to be decided. The three techniques FDMA, TDMA, and CDMA have already been presented. Since the use of OFDM is desired, the special case of FDMA, OFDMA is an interesting candidate for the access scheme. Using OFDMA would allow multiple users to transmit data at the same time. However, when multiple clients are sending, all transceivers modulate the signal with a slightly different carrier frequency offset. This destroys the orthogonality between the subcarriers. El-Tanany et. al. showed in [17] that the intercarrier interference caused by different frequency offsets limits signal quality in every subcarrier. An uncorrected relative offset of $\sigma = 0.02$ gives a signal to interference ratio (SIR) of $28\,\mathrm{dB}$, an offset of $\sigma = 0.05$ results in a SIR of $21\,\mathrm{dB}$. There are two ways to compensate this frequency offset. A control loop that measures the offset and tells the clients to correct the carrier frequency can be added. Or, the basestation compensates the offset internally. To do so, the time domain samples have to be corrected for every client as shown in chapter 2.5.1. This however, requires high computational resources. If there are P simultaneous users, The time domain samples have to be compensated P times and the FFT has to be performed on each compensated sample stream. In conclusion, OFDMA needs tightly controlled client hardware in order to work properly. Any client that does not follow the specifications on e.g. frequency accuracy will degrade the performance of other clients. The amateur radio environment however, often uses custom amplifiers, antennas and other equipment. Thus, we cannot require client hardware to follow tight specifications. Therefore, an OFDMA implementation was discarded.

Using CDMA would require similar restrictions on the client hardware. One challenge in CDMA systems is the near-far effect. The signal of users closer to the basestation will have more power than users that are further away. Since users are transmitting at the same time, one users signal is interference or noise to another user.

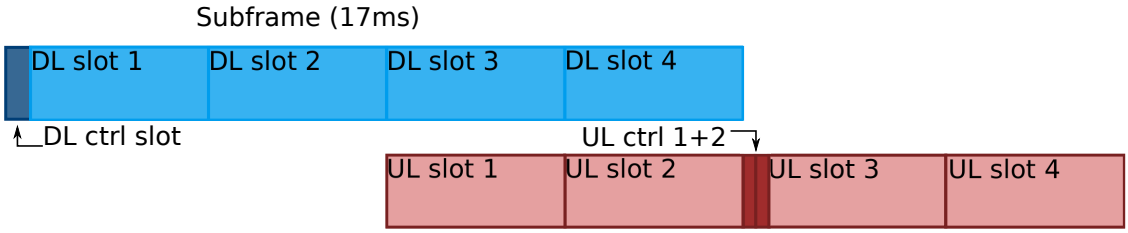


FIGURE 3.5: The structure of one subframe. The upper part defines the structure for the Downlink band, the lower part defines the structure for Uplink.

Therefore, farther away node might not be able to transmit at all, if closer users are transmitting [3]. This problem can be solved by adding a power control loop. This however again restricts the used amplifiers at the client side.

The downsides of OFDMA and CDMA lead us to the remaining technique: TDMA. In a TDMA system we do not have to tightly restrict the used amplifiers and oscillators. If one client uses a non linear amplifier, or a inaccurate TCXO, only this user's connection quality is affected. Other users are not impacted.

3.5.1 Frame Structure

The design of the TDMA frame structure has to fit to the VoIP requirement and the half-duplex requirement for clients. In order to keep the system responsive and the delay low, the interval in which the basestation can define resource allocations was chosen close to the $20\,\mathrm{ms}$ packetization time of VoIP applications. This unit is called a *subframe*, its structure is given in Figure 3.5. Within a subframe we have to define resource units that can be assigned. They are called (time-)*slots*. There are four timeslots per subframe in both uplink and downlink. A timeslot should be long enough to accommodate a single VoIP packet even if a slow modulation scheme is used. This way, a user that is assigned one slot per subframe iteration (<20ms) is able to transmit VoIP traffic. We define a slot as consisting of 14 OFDM symbols, or $T_{slot} = 3.718\,\mathrm{ms}$. A guard interval has to be added between each slot, because the client's amplifiers require a certain RX to TX switching time. For simplicity we set the guard interval T_{GI} to $265.6\,\mu\mathrm{s}$ the duration of one OFDM symbol ($68T_s$). In the downlink, an additional slot that indicates the slot assignments per subframe is required. It is located at the beginning of every subframe and has a length of 2 OFDM symbols. In the uplink, two short control slots are added. They are assigned to all users on a round robin basis and can be used to request uplink resources. Uplink and downlink subframes are shifted by $\sim 1/2$ subframe length, i.e. 8 ms. This is done to give clients the time to receive the slot allocation at the start of the subframe and prepare data to send. So far, the system does not include any random access slot for clients. There is also no synchronization sequence transmitted for timing and frequency offset estimation. We define a larger structure, the *frame* consisting of 8 subframes. All subframes except the first subframe per frame have the structure as described before. In the first subframe, we add a synchronization sequence in the downlink and a random access slot in the uplink. Figure 3.6 shows the frame structure and the position of the random access

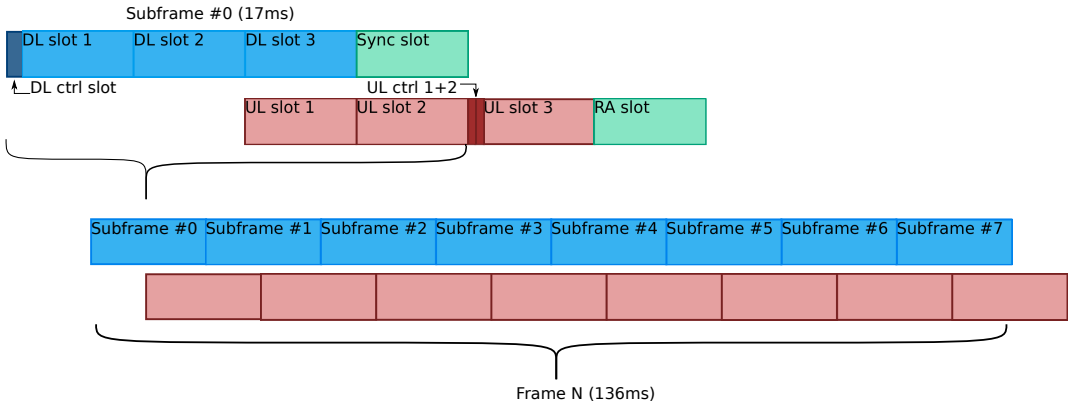


FIGURE 3.6: Frame structure of the system. The first subframe contains the special random access slot and the synchronization slot.

and synchronization slot within the frame. A more detailed definition of all slot types is given in Appendix A.

3.5.2 MAC Procedures

This section will give an overview of the most important procedures that the MAC layer has to support. These are namely

1. Getting Initial Access to the system
2. Idling: How client and basestation monitor the connection status
3. Uplink requests: How clients can get uplink resources
4. Switching the modulation coding scheme
5. Sending and Receiving frames

As part of these procedures, a set of MAC messages have been defined. Their description can be found in Appendix B.

Initial Access The first step when getting initial access is the timing synchronization in downlink. This is done with the Schmidl & Cox synchronization algorithm. Next, the frequency offset is estimated using the same sequence. Now the client is able to decode the downlink control slot and knows the position of the random access slot. It can send an association request via the random access slot and will wait for a association response. It is sent via a downlink slot assigned for broadcasting, since the client and basestation do not have defined any user identification, yet. After decoding the association response, the user knows its allocated *userid* and can start requesting data transmissions.

Idling After a user has successfully associated to the basestation, it has access to the dedicated resources. The basestation will assign uplink control slots in a round robin fashion to all connected users. These can be used to request uplink slots for data transmission. In order to free the limited amount of active connections when they are not used anymore, any client has to regularly answer to the basestation. This is done by keepalive messages that are regularly transmitted via the uplink control slots. If the basestation did not successfully receive any message from a client for a while, it will end the connection. During such an event, it informs the client of the connection end and immediately removes the client afterwards. If the client receives such a message, because maybe only the uplink was disturbed, it will also close the connection. The client furthermore monitors the assignments from the basestation. If there are no assignments for a while, it assumes that the connection has been closed.

Uplink requests Clients that are connected and idling, regularly get assigned to uplink control slots. Each client is assigned one control slot per frame, i.e. every $136\,\mathrm{ms}$. A client that has data to transmit can use the slot to send a uplink-request message. This message contains the number of bytes that the user wants to transmit. The basestation will store the requested amount of bytes from the client. At the beginning of the next subframe, the basestation will schedule uplink data-slots for the corresponding client. It will continue so over the next subframes until enough slots to transmit the requested data size have been granted.

Switching the MCS We required the modulation to be adaptive, therefore it must be possible to switch the MCS while a client is connected. Switching the MCS can be both requested from the client and basestation. The basestation initiates the MCS change by sending a *mcs-info* message. If a client receives such a message, it will switch to the new MCS and send an acknowledgment. As soon as the basesation receives the ACK, it will start using the new MCS.

Sending and receiving data The MAC layer has to be able to transport Ethernet frames as payload (see Section 3.6). Since a time-slot is only able to transport 60 Bytes in the worst case, we have to split the Ethernet frames into multiple slots. This is done by fragmentation. All MAC layer SDUs are transmitted via MAC data messages. The header of each data message contains a *fragmentation number* and a *sequence number*. The fragmentation number is increased for every fragment of a SDU being sent. The sequence number is increased for every completed SDU transmission, to ensure that the reassembler does not mix up fragments from different SDUs.

3.5.3 PHY layer interface

The PHY layer interface defines how the MAC layer messages can be mapped to the physical layer timeslots. This is achieved with the concept of *logical channels*. To keep it simple, only one single logical channel type is defined. The size of the channel is determined by the size of the timeslot. MAC messages are put into the channel one after each other. The MAC message sizes are either fixed or the length is indicated in the message header, allowing the receiver to unpack the messages again. At the end

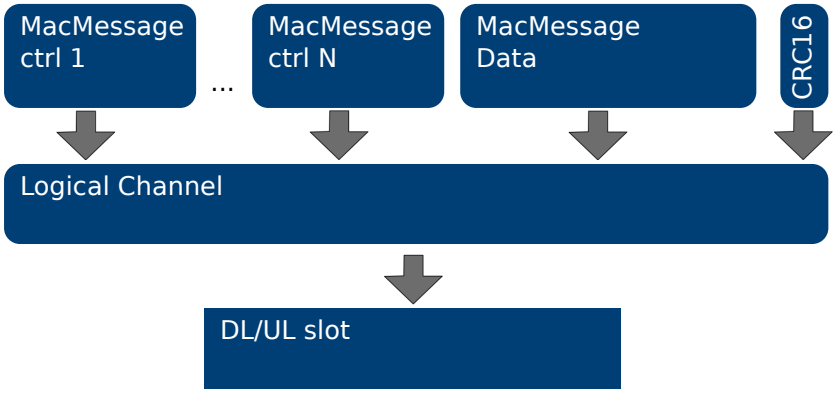


FIGURE 3.7: Overview of the MAC to PHY layer interface. Multiple MAC messages are put into a logical channel and a CRC is added. The resulting bytestream is modulated on a physical layer slot.

of each logical channel, a cyclic redundancy check (CRC) is added. The CRC is used by the MAC layer to check for data integrity.

3.5.4 ARQ

In order to guarantee data integrity to the higher layers, automatic repeat request (ARQ) schemes are used. They consist of three parts: Error checking at the receiver side, requesting retransmission at the receiver side and keeping track of what has been received at the transmitter side. A basic ARQ algorithm is the *Stop-and-wait* ARQ, in which the transmitter sends one frame and waits for an ack before it sends the next frame. If a timeout occurs, the frame is assumed as lost and will be transmitted again. However, when transmitting VoIP such a behavior can be unwanted. We do not need to guarantee correctness of all transmissions, instead we have to guarantee a certain delay of the transmissions. The VoIP codecs can handle a certain frame loss and do not benefit from retransmitted and delayed frames. Thus, an acknowledged transmission mode is unfavoured for VoIP applications. We will get better VoIP performance by only checking the correctness of a logical channel, without acknowledging and retransmitting it.

The remaining traffic will mostly be using TCP at the transport layer. TCP implements a ARQ scheme itself, so data integrity will be guaranteed by higher layers. Nevertheless, in such transmissions, a ARQ implementation could be useful. TCP segments have a size of up to $1460\,\mathrm{Bytes}$ when being encapsulated in Ethernet frames. In high load scenarios, a single client might only be able to transmit with a few kilobits per second. It will take hundreds of milliseconds to transmit single a TCP segment, furthermore the transmission will be fragmented over many timeslots. The delay until a lost segment is detected is therefore large, this will slow down the effective throughput even more. Employing a ARQ scheme at link level can help significantly in this case. Here, we can request retransmission of a single missed timeslot and do not have to wait for the complete TCP segment to arrive and be retransmitted.

Our system therefore is designed to support both an unacknowledged and an acknowledged mode. We can signal successfully received data fragments by sending

the segment and fragment number of it. In the unacknowledged mode the receiver only verifies the CRC and drops frames that could not be received. As part of the thesis we only implemented the unacknowledged mode.

3.6 NETWORK LAYER INTERFACE

The last part of the system design phase was building an interface to allow IP traffic pass through the system. This is achieved by so called *Ethernet transparency*. In short, we will forward IEEE 802.03 packets as our MAC layer SDUs. Every client and the basestation run a Ethernet stack, which interfaces to our designed MAC layer and the Network stack. The key benefit of this approach is that we only have to design a simple interface from our MAC to the Ethernet stack. The interfacing protocols and functions from the Ethernet stack to IPv4 or IPv6 already exist. It should be noted that we do not reimplement the complete Ethernet topology, where all clients are within one local area network and can directly communicate with each other at Ethernet level. Clients in our network will see a local Ethernet network consisting of the client itself and the basestation. The basestation on the other hand sees one large Ethernet network with all clients connected. The reason is that at our MAC layer, the client always communicates via the central basestation. Nevertheless, communication between different clients will be possible. It just has to go through the network stack at the basestation.

The implementation of the MAC to Ethernet interface does not require any additional procedures at the client side. Whenever a Ethernet frame shall be sent at the client side, it is put into one of our designed MAC frames. The client sends all packets to basestation. The basestation however has to keep track of the *userid* which is used at the MAC layer and the corresponding Ethernet address to this ID. When forwarding a Ethernet frame, the userid has to be looked up. A table of the userid to Ethernet mapping can be obtained by monitoring all uplink traffic. All source Ethernet addresses in the uplink can be mapped to the userid from which the frame originated. If a Ethernet packet is designated to an Ethernet address where no userid is known for, it can be sent as a broadcast MAC message. The downside of the Ethernet transparency is the overhead caused by Ethernet headers. A Ethernet header has a size of 18 Bytes. For voice applications that create very short packets of 60 Bytes this overhead reduces the user data-rate significantly.

At last, the assignment of a virtual MAC address per station allows a simple way of the legally required station identification. Every amateur radio station has to identify itself by regulary transmitting its callsign. The callsign can be encoded into the 48bit Ethernet address. Proposals for the encoding schemes can be found in [38] [39].

4

Implementation

In this chapter we will go through the implementation of the communication system. The structure of the final system is depicted in Figure 4.1. The system will be implemented on an Adalm Pluto SDR. The main application consists of a platform abstraction, a physical layer module, a MAC layer module and an interface to the Linux IP stack. The platform abstraction either interfaces with the Adalm Pluto FPGA and transceiver, or simulates a communication channel. The interface to the IP stack is realized by using Linux TAP devices. These are virtual Ethernet interfaces, that forward traffic between the Linux network stack and an userspace application. The application will be written in C language.

4.1 Platform

We decided to use the Adalm Pluto from Analog Devices as our SDR platform. It costs less than 200 Euro, which makes it cheaper than most other SDRs and fulfills the cost requirements we set at the beginning. The Adalm Pluto comes with a Xilinx Zynq 7010 SoC for signal processing. It consists of a field-programmable gate array (FPGA) and an ARM cortex CPU in which a Linux system is run. We can therefore implement our whole application on the SDR and make use of the Linux network stack. End-users will be able to connect to the Pluto via USB and an Ethernet interface will be provided to the host PC. No signal processing has to take place on the host PC. This also reduces the delay between the signal processing application and the transceiver. Samples do not have to be transferred via USB or Ethernet to the host PC before they are processed, they are transferred immediately via direct memory access (DMA) to

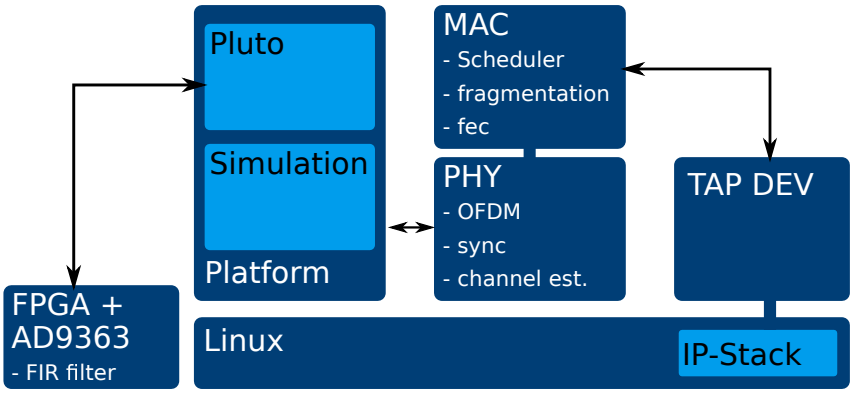


Figure 4.1: System block overview of the implemented communication system on the Adalm Pluto SDR platform.

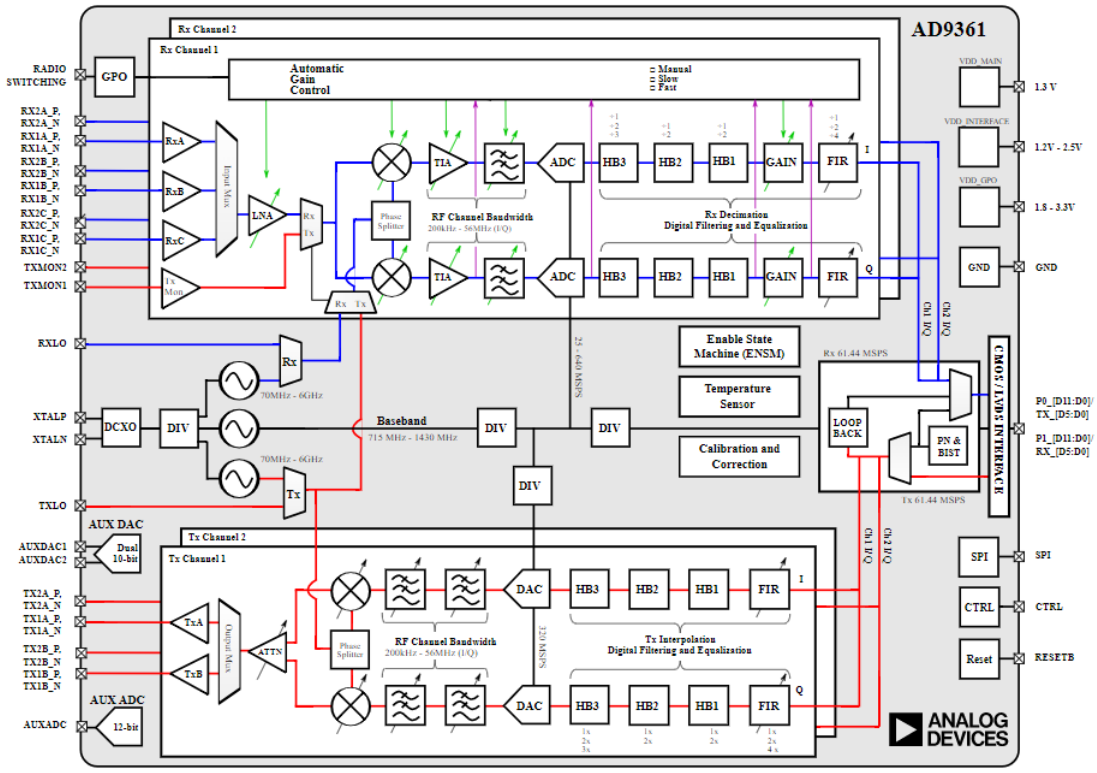


FIGURE 4.2: Structure of the AD9361 transceiver. It has the same structure as AD9363 transceiver that is used, except there is only one TX and one RX channel. Figure taken from [40].

the ARM system. The transceiver on the Pluto is the AD9363[1] by Analog Devices. It has a tuning range of 325 MHz to 3.8 GHz and a sampling rate of 520 ksamps/s up to 61.44 Msamps/s. Lower sampling rates can be achieved by using an additional decimation filter on the Pluto FPGA.

4.1.1 *AD9363 Transceiver configuration*

The structure of the AD9363 transceiver is given in Figure 4.2. It is a direct conversion transceiver, the carrier frequency is immediately mixed down to the baseband. The analog baseband signal is then converted to a 12bit digital signal stream, which is filtered and down-sampled by multiple filter stages. We can interface to the transceiver via the *libiio*[2] library. At first, the interpolation and decimation filters of the transceiver were designed. The first filtering step is done by an analog lowpass filter. It is positioned after the mixer and before the analog to digital converter (ADC). The ADC on the AD9363 has to run with at least 25 Msamps/s. After the ADC, three half-band (HB) decimation filters with predefined coefficients can be activated to lower the symbol rate. Another FIR filter with up to 128 custom filter taps can optionally

[1] Early versions of the Pluto shipped with an AD9364. All devices we had in use were AD9363's.
[2] https://github.com/analogdevicesinc/libiio

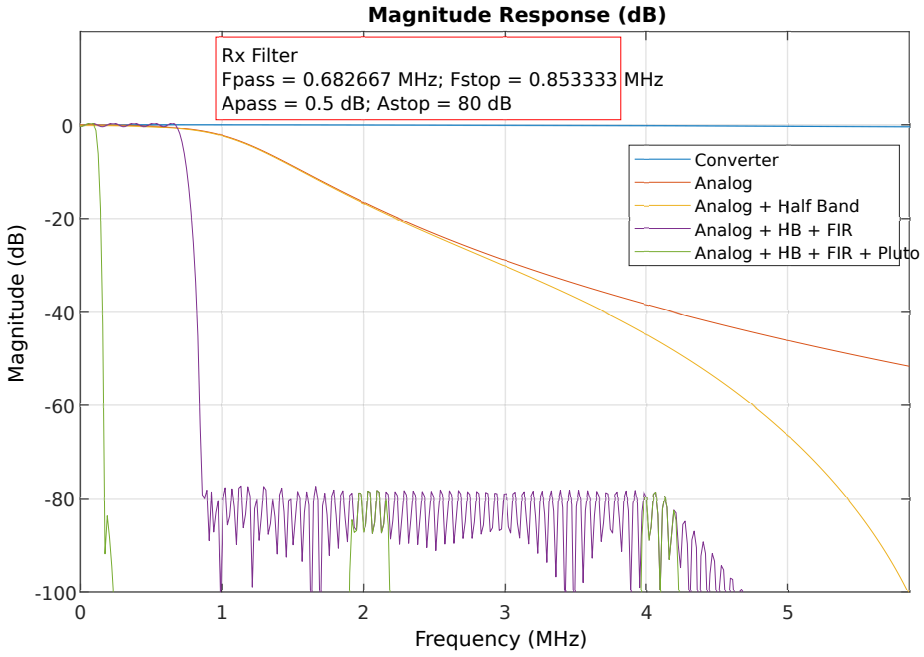


FIGURE 4.3: Magnitude response of the different RX filter stages. Fpass and Fstop values represent the filter stage before the 8 times decimation filter on the Pluto FPGA.

decimate the symbol rate by factor 4. We have to use the additional decimation filter on the FPGA of the Adalm Pluto, which decimates by factor 8, to get to our desired sampling rate of $256\,\mathrm{kHz}$. The custom FIR filter can be designed with a matlab tool. The resulting magnitude response of all filters is displayed in Figure 4.3. It can be seen that we get a resulting pass band of $85.3\,\mathrm{kHz}$ after the Pluto filter stage. This is wide enough, so that our used signal bandwidth of $160\,\mathrm{kHz}$ is not affected. The expected output signal including all filter was simulated using matlab, the results are shown in Figure 4.4. All requirements are fulfilled by a large margin. The attenuation in the out of band spectrum is expected to be $80\,\mathrm{dB}$.

The transceiver includes an automatic gain control (AGC) loop. It can be set to three presets *fast-attack*, *slow-attack* and *manual*. The fast attack mode is designed for burst transmission which occur when using for example CSMA. The slow attack mode is designed for FDD applications where data is constantly being sent. However, we are not able to use any of these two presets. Both presets have a constant gain update interval in which the gain is adjusted. In both cases, the update often occurs in the mid of receiving an OFDM symbol. This leads to large distortions in the signal after calculating the FFT, so that no subcarriers can be decoded. We instead use the manual mode and implement the AGC in software. The application estimates the gain every time the synchronization sequence is received, i.e. every 136 ms. The hardware gain of the Pluto is adjusted just before the next synchronization sequence is received, since there is a large enough gap to adjust the gain without destroying any RX data.

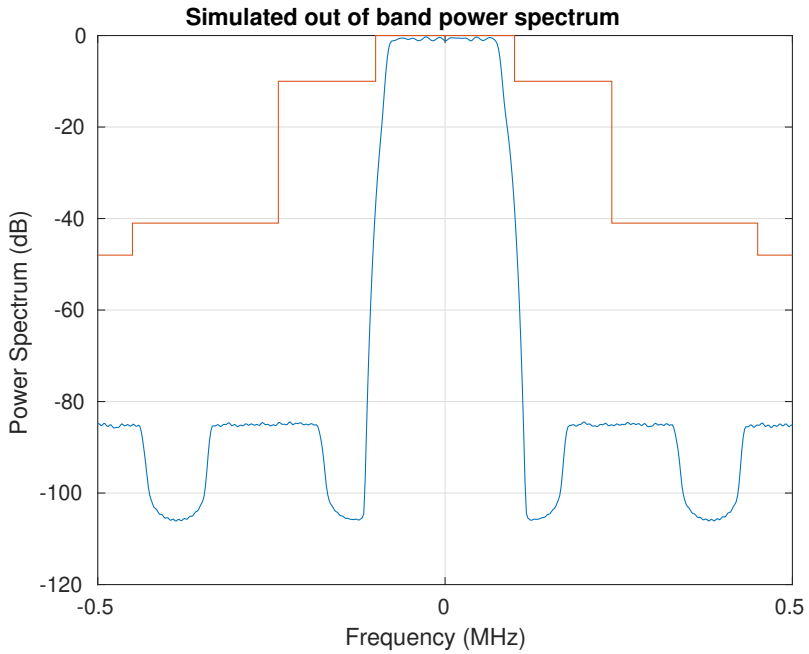


FIGURE 4.4: Simulated power spectrum including all Adalm Pluto filter stages. The orange line shows the ITU spectral mask for amateur radio services, assuming $10\,\mathrm{W}$ output power.

In order to transfer samples between the transceiver and the Linux system, a buffer structure is used. In order to keep the buffering delay low, we try to use very small buffers. This however increases the risk of buffer over- and under-flows. Buffer overflows will occur in the receive chain, if our application is not fast enough to process incoming buffers. Buffer underflows can occur in the transmit chain, if the application does not generate new sample blocks at the rate of the buffer transmit rate. Libiio implements a queue of buffers for both TX and RX chain to loosen the timing constraints imposed from the buffer length. In the receive chain, the transceiver can fill up to N buffers with samples. Every time the Linux application reads from the buffer queue, this buffer becomes free and is ready to be enqueued again by the transceiver. We implement the system with a buffer length of $136\,\mathrm{samples}$. This equals a delay of $136T_s = 531.25\,\mu\mathrm{s}$ per buffer. The queue size is set to $N = 4$. Since the transmitter side will try to keep the buffer queue full to prevent underflows, we expect a introduced delay of $2.125\,\mathrm{ms}$ at the transmit chain. On the RX chain, the delay is equal to a single buffer most of the time, since the RX processing will constantly poll for new buffers. It should be noted that these introduced delays are shorter than the $8\,\mathrm{ms}$ shift between downlink and uplink. This is important in order for clients to be able to correctly use the uplink slots.

4.1.2 Modifying the TCXO

The Pluto uses a $40\,\mathrm{MHz}$ temperature controlled crystal oscillator (TCXO) with an accuracy of $25\,\mathrm{ppm}$ by default. Thus, the local oscillator (LO) frequency at $440\,\mathrm{MHz}$

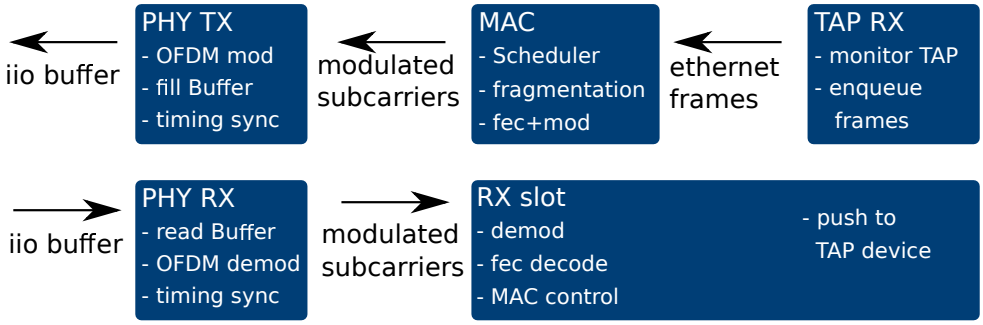


FIGURE 4.5: Threading structure of the application.

can be off by up to $11\,\mathrm{kHz}$. This width is larger than the width of two subcarriers. Additionally, the symbol timing could be off by up to $25\,\mu\mathrm{s} \approx 6T_s$ every second, which equals more than 6 symbols offset per second. We expect that the oscillator is too inaccurate to be used, so we changed it against a oscillator with $0.5\,\mathrm{ppm}$ accuracy. During evaluation, we will measure the accuracy of both the new and old TCXO.

4.1.3 Application

Our configuration of the transceiver buffers puts tight restrictions on the timing of our application. To guarantee a stable operation, the transceiver buffers must be filled and read every $531\,\mu\mathrm{s}$. The default Linux kernel is not able to guarantee a high enough availability of a process to perform such a high frequent I/O. Interrupts from for example the USB I/O cause the processes to wait for up to milliseconds. We therefore use the widely known Linux realtime patch[3]. It solves the latency issues by employing threads for all interrupt routines. The interrupt routines are then scheduled by the operating system and can be delayed if some realtime task has important work to do. Using the realtime Linux kernel we were able to solve some initial timing issues that we came across. As a second step, we use multiple threads for the different application tasks. We use five different threads in total as displayed in Figure 4.5. Two threads are responsible to interface with the transceiver buffers. They run on an isolated CPU core to guarantee CPU resource availability. The other threads do not have as tight timing requirements. They run on the other core of the ARM CPU, where also the remaining Linux system is executed. Figure 4.5 shows the threading structure and the main tasks of each thread. The MAC thread executes the MAC scheduler once per subframe. The RX slot thread is called every time the PHY RX thread received all samples of one slot. It starts the demodulation, runs the Viterbi decoder and handles the contained MAC messages in the slot. It also reassembles the higher layer data and forwards it to the Linux TAP device. Finally, the TAP RX thread continuously monitors the TAP device and adds incoming Ethernet frames to the MAC dataQueue.

[3] https://wiki.linuxfoundation.org/realtime/start

4.1.4 Simulation

In order to verify our application under known environmental constraints, a simulation environment has been implemented. The simulation platform provides the same interface functions as the previously described Pluto platform abstraction. This way, we can use the same PHY and MAC layer source code during the simulation and during deployment. The simulation platform does not simulate the Pluto hardware, i.e. none of the transceiver FIR filters are simulated. It is merely a simulation of the transmission channel at the baseband level. The simulation includes a AWGN channel, a constant carrier frequency offset, a multipath model, and flat rayleigh fading. The channel simulation is done using the library liquidsdr[4], which provides many signal processing routines. For channel simulation the *channel_cccf* object can be used. We use the GSM typical urban channel profile from Figure 2.2 as multipath channel filter taps. The library does not yet provide a Rayleigh fading channel, so we implemented one using Jakes model as presented in Section 2.2.4.

4.2 PHY LAYER

At the Physical layer, the timing synchronization, the frequency offset compensation, OFDM modulation and demodulation and the slot en/decoding has to be implemented. Most of this was realized by again using liquidsdr. The library provides two objects *ofdmframesync* and *ofdmframegen* for OFDM modulation. We expanded them by a more advanced frequency offset tracking and gain estimation algorithm.

4.2.1 Timing and Frequency synchronization

The initial timing and frequency estimation is based on the Schmidl & Cox synchronization algorithm presented in Section 2.5.1. The first part of the synchronization sequence consists of two identical sequences, just as in the presented algorithm. It is used to gather a coarse timing and frequency offset estimation. Another, longer training sequence is sent immediately after the first synchronization symbols. This one is used to get a finer timing and frequency offset estimate. In difference to original proposal by Schmidl & Cox, the samples are not correlated with a time-shifted version of the received samples. Instead, the expected training sequence is used for correlation. After synchronization is achieved, the repeating synchronization signal is used to track the timing and coarse frequency offset. This builds an outer synchronization loop that ensures a coarse system synchronization. Although only giving a coarse synchronization, this loop is able to correct large frequency offsets.

The frequency offset estimation from the synchronization sequence is not sufficient for demodulating high order QAM symbols. We therefore use pilot symbols within each slot to constantly adjust the frequency offset. Here, the phase offset between two received OFDM symbols is compared to give the frequency mismatch. Let the old

[4] https://liquidsdr.org/

phase offset be ϕ_{old} and the new estimated phase offset be ϕ_{new}. The additional phase rotation needed per time domain symbol is

$$d\phi_{corr} = \frac{\phi_{new} - \phi_{old}}{t_p \cdot N_{fft}}. \tag{4.1}$$

where t_p is the number of OFDM symbols between two symbols containing pilots (in our case $t_p = 2$). In order to smoothen the estimates, we apply lowpass filtering. Let $d\phi_{old}$ be the already corrected phase step per sample. The new phase step is

$$d\phi_{new} = d\phi_{old} + \alpha \cdot d\phi_{corr}, \tag{4.2}$$

where $\alpha \in [0, 1]$ can be used to tune the filtering, we set it to value $\alpha = 0.1$.

4.2.2 Channel equalization

Channel equalization is performed by estimating a phase and gain correction value for each subcarrier. Every subcarrier symbol is then corrected by the estimates. By default, liquidsdr estimates the gain every time the synchronization sequence is received. Since this estimation occurs less often than the minimum coherence time of our channel and we have to expect frequency selective fading, a more advanced gain estimation has to be implemented. We implement a gain estimation based on pilot symbols, similar to the already implemented phase offset estimation on pilot basis. The received pilot subcarriers are compared to the expected pilot sequence and gain correction values are calculated for each pilot subcarrier. In order to correct the gain of data subcarriers, these estimates are interpolated using a second order polynomial filter. OFDM symbols without any pilots use estimates from previous symbols. The phase offset estimation works likewise.

4.2.3 Convolutional coding

Slots are encoded using convolutional codes. The library libfec implements different convolutional codes and Viterbi decoders. We use a code with constraint length $k = 7$ and rate $r = 1/2$. Viterbi decoding is a computationally intensive task, first performance measurements showed that Viterbi decoding requires more resources than all other system components. To speed up the decoding process, SIMD (single instruction multiple data) operations can be used. Libfec already makes use of SIMD on Intel CPUs with Streaming SIMD Extensions 2 (SSE2) support. Arm NEON, the ARM SIMD instructions, is not supported yet. To our knowledge, no open source implementation of a $r = 1/2$ decoder with Neon instructions exists. Therefore we derived our own Viterbi decoder based on the SSE2 implementation in libfec. We will evaluate its performance in the next chapter.

4.3 MAC LAYER

At the MAC layer, the main component to be implemented is the scheduler. We have to implement two separate scheduling algorithms. One, which runs on the client side and one that runs at the basestation. The client scheduler will be executed once per

subframe (i.e. every 17 ms) after the downlink control slot is received. It uses the uplink data and uplink control message queue as input and maps the messages to the slots that the client is allocated to. Every client scheduler has to follow the allocations made by the basestation. If a client is assigned some uplink slot, even if there is no data to be transmitted, the client has to generate a dummy message and send it instead. This is necessary, because the basestation will expect useful data in order to perform channel estimation. The pseudocode of the algorithm is given in Alg. 4.1. The client first iterates over all data slot allocations and maps control and data messages to these slots. The check in line 2 enables a fast way of allocating more data slots. If the client still has data to send in the last assigned slot of the subframe, it will piggyback a request for more slots to this data slot. At the end, the client will either add uplink requests or a Keepalive notification to allocated control slots.

Algorithm 4.1 Pseudocode of the client scheduling algorithm

```
1:  for all slot in assignedUplinkDataSlots do
2:      if isLastAssignment(slot) then
3:          addUplinkRequest(controlQueue)
4:      end if
5:      mapControlMessages(slot,controlQueue)
6:      if hasMessage(dataQueue) then
7:          mapDataFragment(slot,dataQueue)
8:      else
9:          mapKeepaliveMessage(slot)
10:     end if
11: end for
12: for all slot in assignedUplinkControlSlots do
13:     if hasMessage(dataQueue) then
14:         addUplinkRequest(controlQueue)
15:         mapControlMessages(slot,controlQueue)
16:     else
17:         mapKeepaliveMessage(slot)
18:     end if
19: end for
```

The basestation scheduler needs a more advanced algorithm, since it has to split up the limited resources in a fair way to all clients. As stated earlier, we will use a round robin scheduler to guarantee a fair resource sharing. At first, the scheduler allocates the uplink control slots. These are allocated on a pure round robin basis. Next, the broadcast queue is checked and at most 1 slot is allocated for broadcasting. This gives broadcasting priority over user-data transmissions, but we limit the broadcasting traffic to 25% of our bandwidth to ensure that users are still able to get some bandwidth. The remaining downlink slots are allocated to clients with nonempty downlink queues. At last, the uplink data slots are assigned. This assignment is also done in round robin fashion over the set of users that have requested uplink transmissions. The requested packet size is updated after each assignment. This way the basestation automatically stops assigning slots to clients that got enough slot assignments to transmit their data. Before any slot assignment is made, the scheduler

has to check that the assignment does not create any time domain overlap in the uplink and downlink. Otherwise the half duplex constraint is violated. The pseudocode for the algorithm is given in Alg. 4.2.

Algorithm 4.2 Pseudocode of the basestation scheduling algorithm

```
1: setUplinkControlAssignments();
2: slotidx = 0;
3: if hasMessage(broadcastQueue) then
4:     mapDataFragment(slotidx,broadcastQueue);
5:     slotidx = slotidx + 1;
6: end if
7: while slotidx < totalSlots do
8:     user = getNextActiveUser();
9:     mapControlMessages(slotidx,user.controlQueue);
10:    mapDataFragment(slotidx,user.dataQueue);
11:    slotidx = slotidx + 1;
12: end while
13: slotidx = 0;
14: while slotidx < totalSlots do
15:    user = getNextActiveUser();
16:    setUplinkAssignment(slotidx,user);
17:    slotidx = slotidx + 1;
18: end while
```

4.4 INTEGRATION TO EXISTING HAMNET

Figure 4.6 shows a deployment scenario of the designed communication system and the integration into the existing HAMNET. The access network which was designed in this thesis forms a local area network between all Adalm Plutos. Host PCs will connect to the Pluto over Ethernet. The Pluto acting as a basestation is connected via another host PC to the existing HAMNET. A simple way of letting host PCs communicate with the backbone is to create a network bridge between the two Ethernet interfaces at the Adalm Pluto Linux system. This will bring all connected Host PCs into the same LAN, allowing communication at Layer 2 within the new communication system. The backbone PC has to act as a Layer 3 router to route traffic between the 70 cm network and the existing HAMNET. IP address distribution within the 70 cm network could easily be done by hosting a DHCP server at the backbone PC which binds to *eth0*.

Another deployment scenario is to create separate subnets for each Adalm Pluto and connected client hardware. Then, corresponding IP routes at the Adalm Plutos and the backbone PC have to be created, every Adalm Pluto now acts as a Layer 3 router. However, assigning subnets to every connected client requires more manual work. Furthermore, the set of clients which connect to the 70 cm basestation should be relatively static, because of the required subnet assignment. In a slight variation of the latter approach, the Adalm Plutos are assigned only one public IP address and they create their own subnet which is not visible to the public. The Adalm Plutos will run a Network address translation (NAT) service to let the users access the public network.

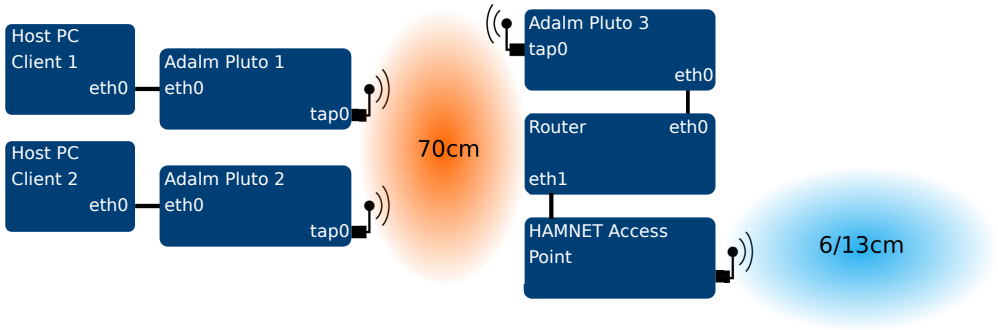


FIGURE 4.6: Possible deployment scenario of the designed communication system.

A practical deployment example of the first and last approach is given in Appendix C.

5

EVALUATION

In this chapter, the previously implemented system will be evaluated. We first simulate the PHY and MAC layer to ensure they work as expected. We then go on to test the implementation on the SDR. Here, we first check how the RF characteristics match our requirements. We also will assess the overall performance of the system and try to identify limiting factors in the signal processing chain. We finally test the performance of applications like VoIP.

5.1 SIMULATION

As stated in Section 4.1.4, our channel simulation consists of an AWGN channel, a multipath model based on the GSM typical urban power delay profile, a constant carrier frequency offset, and a Rayleigh fading model. We will start by simulating the bit error rate (BER) at the physical layer. We then put a special focus on the frequency offset estimation accuracy since it is crucial for the OFDM system. At last, the MAC layer delay will be assessed.

5.1.1 Biterror rate

The bit error rate simulations give an overview of the total performance of the physical layer. In our first test, we use the AWGN channel and the GSM typical urban power-delay profile. Downlink and uplink tests were run separately, in each test $30 \cdot 10^6$ bits have been simulated. The BER for the downlink is shown in Figure 5.1, the uplink BER is given in Figure 5.2. Most downlink biterror rates are as expected. When comparing the $r = 1/2$ coded modulation to the theoretical error rates for uncoded modulation (see table3.4) the simulated BER of 10^{-6} has a gain of $3 - 5\mathrm{dB}$ compared to uncoded modulation. This is the coding gain of the 1/2 rate code, it matches to our calculated maximum coding gain of $5.44\mathrm{dB}$ from the system design chapter. The first interesting thing to note, is that the BER of MCS2 (16QAM - r=1/2) drops faster than the BER of MCS1 (4QAM - r=3/4). This result could be reproduced in different tests. It could be caused by the different coding schemes, where the higher coding gain of the r=1/2 code causes the BER to drop faster. However, this phenomena can not be seen when comparing MCS3 (16QAM - r=3/4) and MCS4 (64QAM - r=1/2). Furthermore, the uplink BER of MCS1 and MCS2 also do not produce the same result. Secondly, we observe that the BER for very high order MCS (64QAM and r=3/4 coding) reaches a limit for SNR higher than $25\,\mathrm{dB}$. This is most likely caused by a limit of the pilot based channel estimation. As we will show in the next section, the pilot based frequency offset estimation does not produce better estimations after a certain SNR level. This limits the BER performance. Furthermore, we see that the uplink BER for the same SNR and MCS is higher than the comparable downlink BER. This is likely caused by

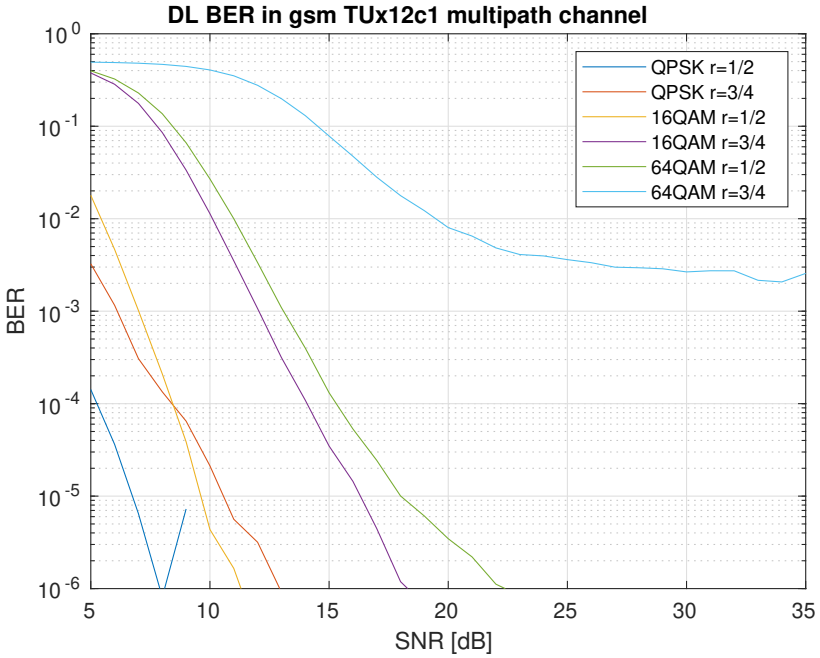


FIGURE 5.1: Bit error rate for different modulation schemes in downlink multipath channel.

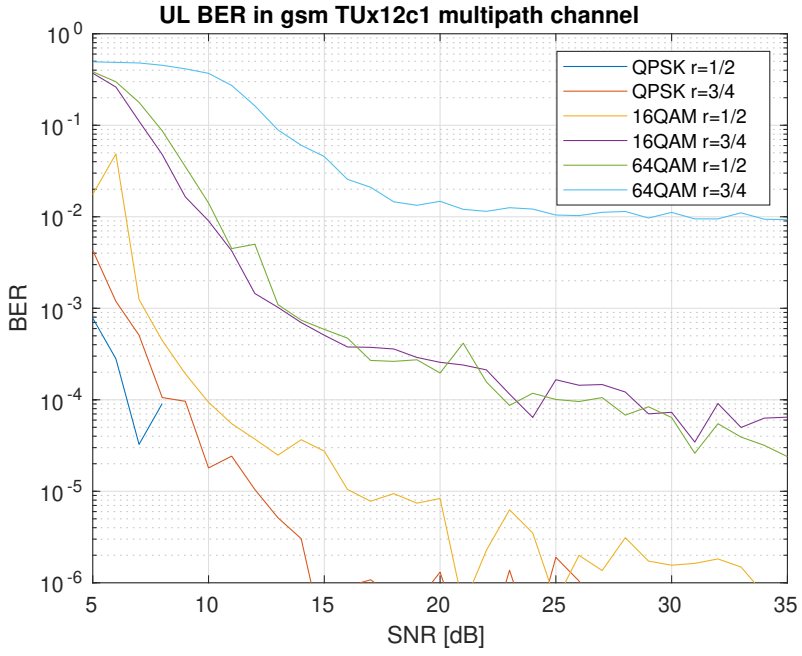


FIGURE 5.2: Biterror rate for different modulation schemes in uplink multipath channel.

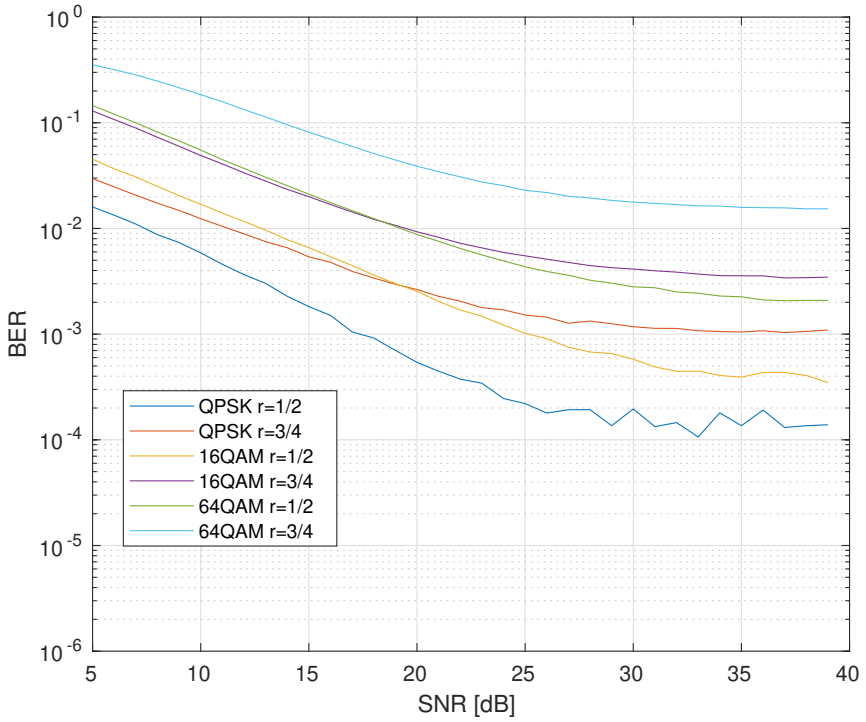
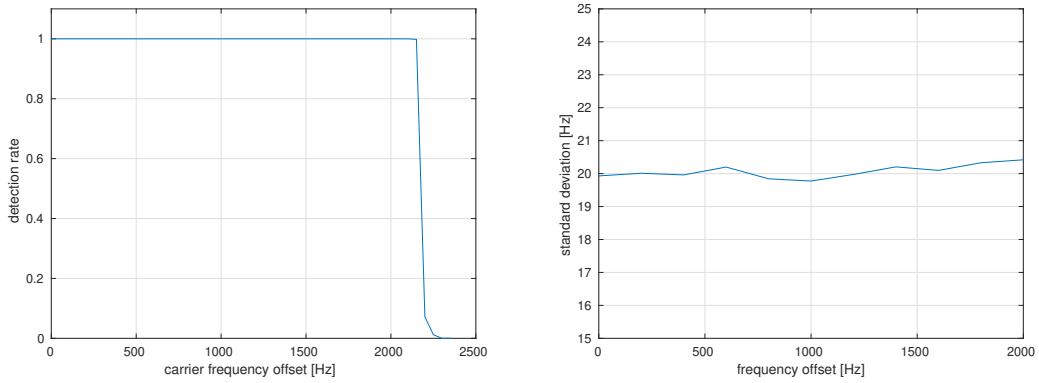


FIGURE 5.3: Bit error rate for different modulation schemes in the downlink multipath channel including flat Rayleigh fading with $f_d = 20\,\mathrm{Hz}$.

the better possibilities of channel estimation in downlink. There is no equivalent of the coarse carrier frequency offset estimation loop in the uplink. This loop helps to regain a coarse frequency offset estimation if the pilot based tracking loop looses the offset at one point. Instead, in the uplink the client will loose the connection until the basestation detects the client as disconnected. It then tells the client reconnect and the client performs the coarse frequency offset estimation again.

In a next step, we included Rayleigh fading to our channel model. The maximum Doppler shift is set to $20\,\mathrm{Hz}$, equal to clients moving with speeds of $50\,\mathrm{km/h}$. The results of the downlink simulation are given in Figure 5.3. The BER gets significantly worse when simulating the Rayleigh channel. Even with the most robust MCS the BER does not fall below 10^{-4}. Similar to the previous test, we observe that the BER reaches a limit for high SNRs. We can furthermore see that for a SNR greater than 20, modulation coding schemes using $r = 1/2$ codes outperform $r = 3/4$ codes. The MCS using 64-QAM and $r = 1/2$ code rate achieves better BER than 16-QAM $r = 3/4$ coding, while being more spectrally efficient. The same can be witnessed for 4-QAM $r = 3/4$ and 16-QAM $r = 1/2$ coding. We could already see this in the simulations without Rayleigh fading. The additional coding gain of the stronger convolutional coding seems to outperform the loss due to the use of higher order modulation. With these results, it would be interesting to see the performance of 256-QAM with $r = 1/2$ coding. One should also consider testing more robust coding using $r = 1/3$ codes.



(a) (b)

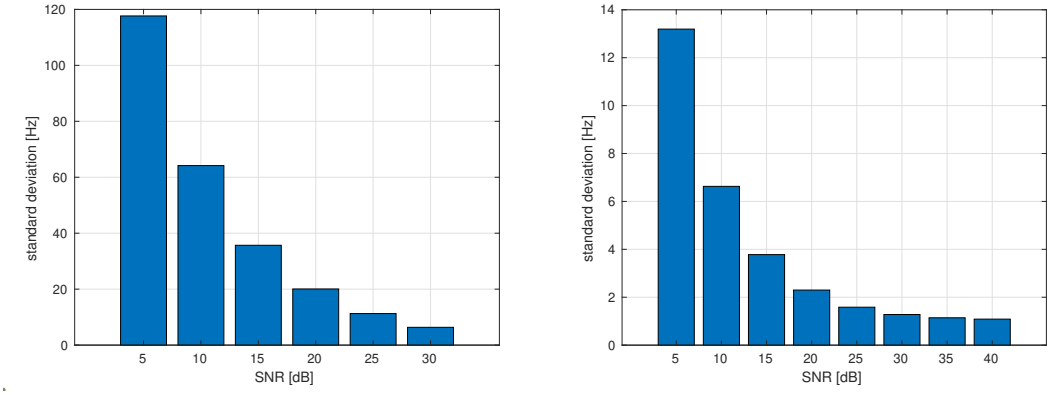
FIGURE 5.4: (a): Synchronization sequence detection rate for different simulated carrier frequency offsets. (b) Standard deviation of the frequency estimation error at different simulated carrier frequency offsets.

We made an interesting observation when looking at the slots that the client failed to decode. In all slots where the decoding failed, i.e. slots that contain at least one biterror, the average number of biterrors was 40. In other words, whenever a slot contained errors, it contained many errors at once. That is different to the biterror distribution in the simulation without Rayleigh fading. This is can be attributed to burst errors caused by deep fading of the Rayleigh distribution. The interleaving scheme should already compensate for this, but probably the interleaver block length is too small. A further investigation of this issue by testing interleaving schemes with more interleaving rounds would be interesting.

5.1.2 Frequency offset estimation

Due to our usage of OFDM with considerably small subcarrier spacing, accurate frequency offset compensation algorithms were required. In the following, we will test the coarse frequency offset estimation that uses the synchronization sequence. Then we evaluate the fine frequency offset tracking based on the pilot symbols.

First, we check the maximum offsets that the coarse frequency offset estimation is able to detect. For this test, the AWGN channel with a fixed SNR of $20\,\mathrm{dB}$ and the GSM typical urban multipath profile was enabled. We tested $n = 10000$ synchronization sequences, with varying frequency offsets from 0 to $2500\,\mathrm{Hz}$. Figure 5.4 (a) shows the results. We can see that up to an offset of about $2200\,\mathrm{Hz}$ all synchronization sequences are detected. The detection rate drops sharply for larger frequency offsets. If we consider TCXO with $0.5\,\mathrm{ppm}$ accuracy, this detection range is large enough for the application to find a sync immediately. With the default $25\,\mathrm{ppm}$ TCXO however, the carrier frequency mismatch could be up to $11\,\mathrm{kHz}$. Thus, the synchronization sequence might need to be searched at different LO frequency tunings until it is found. Next, we wanted to see whether the performance of the estimation degrades with the frequency offset. Therefore, we calculated the standard deviation of the estimation



(a) coarse offset estimation

(b) fine offset estimation

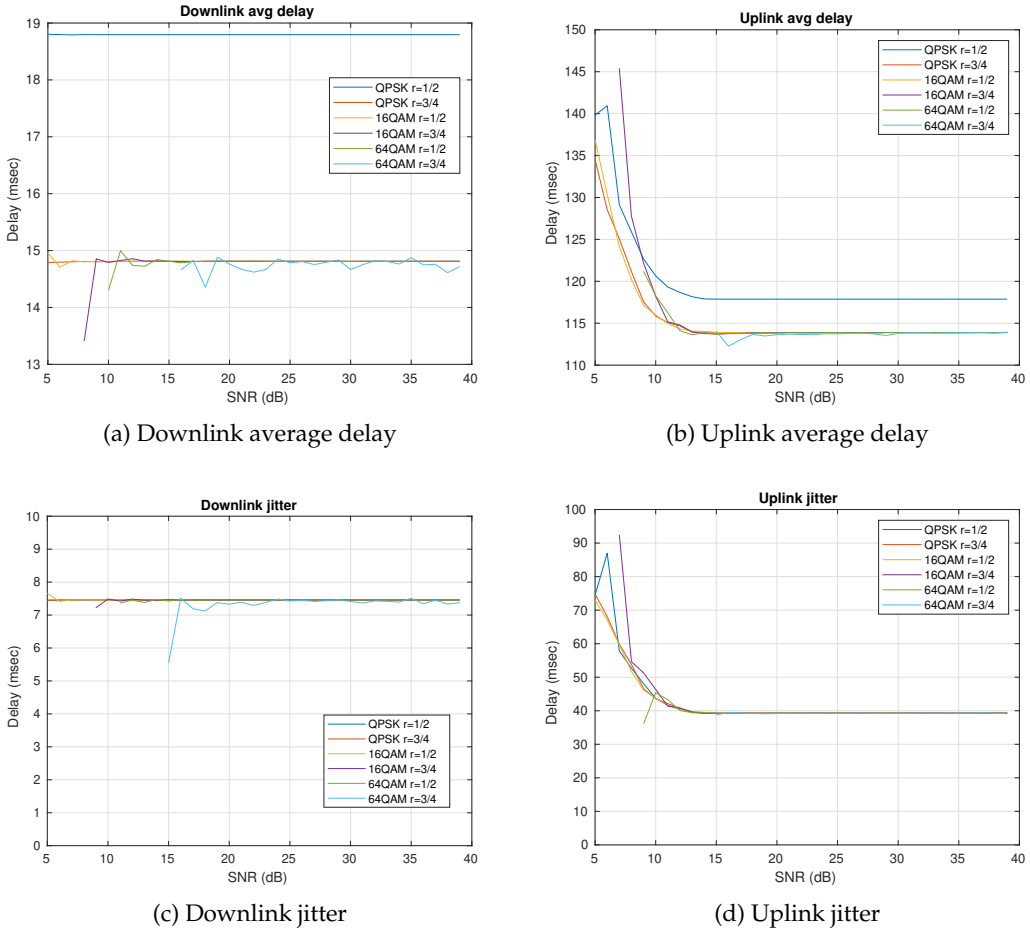
FIGURE 5.5: Standard deviation of the coarse (a) and fine (b) frequency offset estimation error for different SNR.

error using the same simulation parameters as in the previous test. Figure 5.4 (b) shows that the estimation performance does not degrade with the carrier frequency offset. We continued to measure the estimation performance for different SNR values. We kept the simulated frequency offset at 0Hz, as the absolute value does not impact the estimation performance. The results are shown in Figure 5.5 (a). For a SNR of 20dB or higher, the standard deviation is less than 0.005 of the subcarrier spacing. We then looked at the pilot based offset estimation. Figure 5.5 (b) shows that the pilot based estimation under the same channel conditions performs significantly better. We also see that the standard deviation does not decrease much for SNR higher than 30dB.

5.1.3 MAC Layer Delay

Keeping the delay low was one of the key requirements for our system. In this section we will derive theoretical values for the delay and jitter before verifying them in a PHY+MAC layer simulation. In the first scenario, the client is in the idle mode when a packet arrives, meaning that there are no current pending uplink requests or scheduled data transmissions at the arrival time. For a uplink transmission the client will wait for the next uplink control slot, request a data-transmission, wait for the slot assignment and finally transmit the packet. Every client is assigned one uplink control slot every 136 ms, therefore the initial waiting time can be seen as a uniformly distributed random variable X_1 over the interval 0 ms-136 ms. A fixed time of 17 ms (one subframe) goes by from the announcement of the uplink control slot until the actual transmission of the uplink request. When the request is received at the basestation, again 17 ms pass by until the next downlink control slot in which the basestation can announce a slot assignment. The first uplink slot starts 1/2-subframe after the downlink control slot, adding 8 ms to the delay. The actual transmission of one slot takes 3.9 ms. The average delay is thus

$$\bar{T}_{ul} = E[X_1] + 17\,\text{ms} + 17\,\text{ms} + 8\,\text{ms} + 3.9\,\text{ms} = 113.9\,\text{ms}. \tag{5.1}$$



(a) Downlink average delay

(b) Uplink average delay

(c) Downlink jitter

(d) Uplink jitter

FIGURE 5.6: Latency and Jitter simulation for Uplink and Downlink using a Packetization rate of 200 ms. Uplink and Downlink data was simulated seperately.

The jitter is defined as the standard deviation of the delay

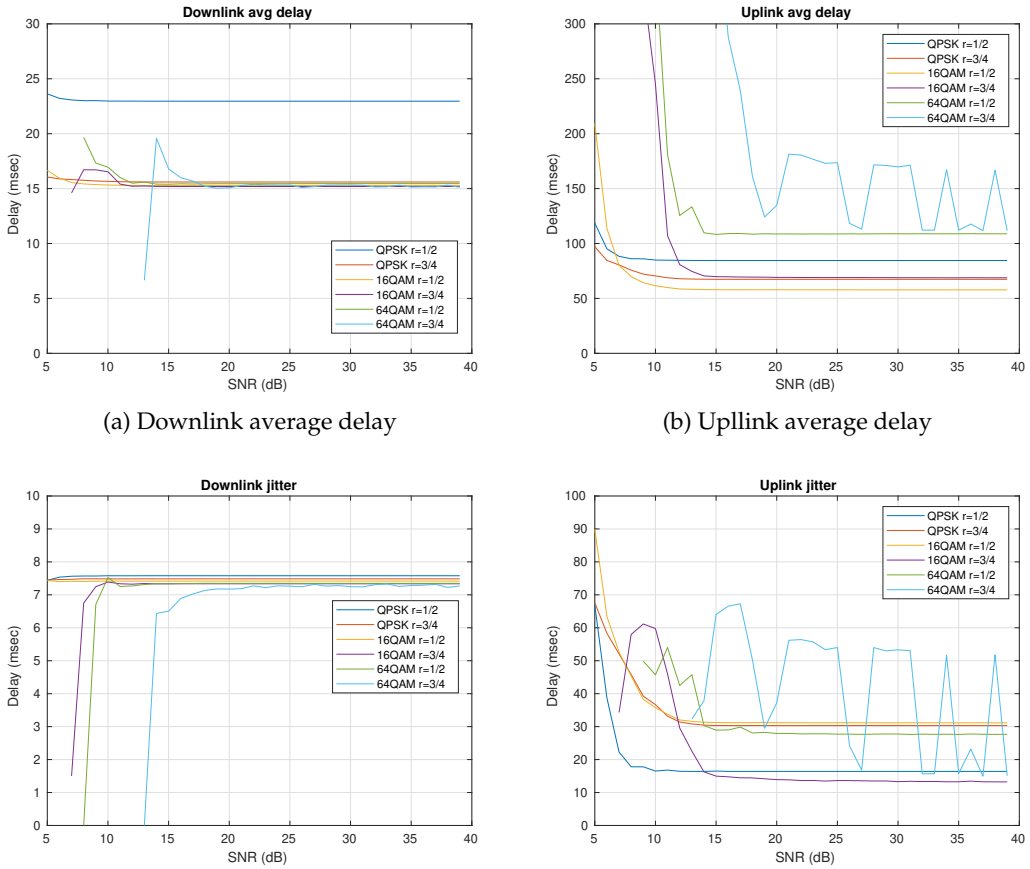
$$J_{ul} = \sqrt{E[X_1^2]} = [\int_0^{136} \frac{(x - E[X_1])^2}{136} dx]^{1/2} = 39.3\,\text{ms}. \tag{5.2}$$

In the downlink, the packet will be queued until the next scheduler iteration and is then immediately sent. The waiting time is uniformly distributed in the interval 0-17 ms, The delay until the packet is received is one slot duration. Thus we have

$$\bar{T}_{dl} = E[X_2] + 3.9\,\text{ms} = 12.4\,\text{ms}, \tag{5.3}$$

$$J_{dl} = \sqrt{E[X_2^2]} = [\int_0^{17} \frac{(x - E[X_2])^2}{17} dx]^{1/2} = 4.9\,\text{ms}. \tag{5.4}$$

For the simulation, we created a MAC data frame with a size of $N_{VoIP-packet} = 60\,\text{Bytes}$ every 200 ms. In total, 6500 packets were simulated. Figure 5.6 shows the results of our simulation. The uplink delay and jitter are as expected for high SNR. At



(a) Downlink average delay

(b) Upllink average delay

(c) Downlink jitter

(d) Uplink jitter

FIGURE 5.7: Latency and Jitter simulation for uplink and downlink using a packetization rate of 20 ms.

low SNR, the jitter and delay increase due to packet loss of the uplink requests. The downlink jitter is higher as expected. This is, because we did not consider that a client is not able to receive every 8th subframe, due to the half-duplex constraint. The client has to transmit in the uplink control channel and thus cannot receive the downlink control slot which overlaps with the uplink control in time domain. This increases the average delay and jitter. Furthermore, we can see that using MCS0, the average delay is slightly higher. When using MCS0, the payload per slot is only 58 Bytes, so two slots are required to complete the transmission of our payload. In another test scenario we want to simulate a VoIP application scenario. We thus will send MAC frames of $N_{VoIP-packet} = 60\,\text{Bytes}$ every 20 ms in the uplink and downlink at the same time. In total, 65000 frames were sent each in the uplink and downlink. The results are shown in Figure 5.7. Comparing these results to the previous simulation, we see that the average delay and jitter in the uplink performs better except when 64QAM r=3/4 modulation is used. The performance advantage differs from MCS to MCS, there is no clear trend visible. Our method of a fast uplink request that can be appended to an uplink data slot seems to depend on the connection speed and the application layer

data rate. It performs best if there is a constant but small queue at the client MAC layer, because in this situation, the MAC layer makes use of the fast uplink requests the most often. Faster connection speeds are able to empty this queue immediately and then fall back to the default and slower uplink request method. This gives a higher delay and jitter in the end. Nevertheless, the requirements for the jitter to be lower than 60 ms are always fulfilled. The desired one way delay of less than 70 ms is not always met in the uplink. But as pointed out in the section on the system requirements, even with slightly higher delays, the system can be seen as VoIP capable. The simulations using 64QAM r=3/4 modulation yield inconsistent results in the uplink. This might be because the BER is considerably high and thus there is a very high frame error rate, giving only few arrived packets.

5.2 SYSTEM TESTING

After evaluating the system in the simulation, we are going to perform tests including the Pluto hardware. In particular, we verify that the out of band emissions are within the requirements, evaluate the effect of changing the TCXO and finally conduct application layer data-rate tests.

5.2.1 Out of Band emissions

The measured spectrum of the basestation without any additional amplifier is shown in Figure 5.8. As we could already see in the simulated spectrum, the signal still has a margin to the ITU spectral mask for amateur radio transmissions. At 100 kHz from the central frequency, the power is down by 16 dB, 10 dB are required. At 240 kHz the power is down by 58 dB, 43 dB are required when transmitting with maximum power (P=15 W). We could not yet perform tests that include an additional power amplifier. Since we have a margin of at least 6 dB at the outer bands, it is likely that the spectrum will fit the spectral mask when an amplifier is used. The maximum output power that we measured during this test was -6 dBm.

When evaluating the spectrum of the system, we saw a significant output at harmonic frequencies of our 439.7 MHz carrier frequency. We therefore conducted further evaluation of the emissions in the spurious domain. The spurious domain is defined as the frequency band that is further away from the carrier frequency than the out-of-band spectrum. Our mixer generates strong intermodulation products at combinations of the carrier frequency and the mixing frequency. In our case the mixing frequency equals our carrier frequency. The Adalm Pluto lacks an analog filter after the mixing stage which would filter the intermodulation products. Figure 5.9 shows a quite strong 3rd order intermodulation product at 1.319 GHz with -10 dBc power. The 5th order intermodulation product has a strength of -18 dBc, also 2nd and 4th order intermodulation is visible. According to ITU recommendation SM.329.12 on emissions in the spurious domain, the emissions should not exceed -55 dBc (considering Category A, P=15 W) [41]. When using an antenna dedicated for the UHF band, the spurious emissions disappear at our receivers. Before deploying the system with an antenna and amplifier, it should be further evaluated whether the harmonics meet the requirements. Otherwise an analog filter is required.

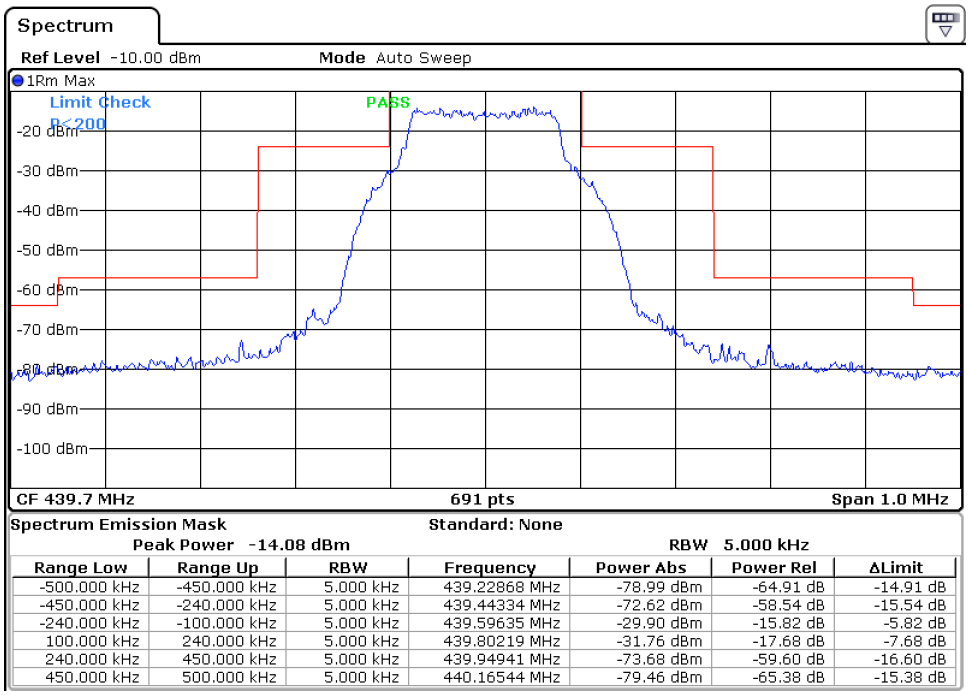


FIGURE 5.8: Out of band emissions of the downlink signal without any additional amplifier. Pluto TX gain is set to 0 (max).

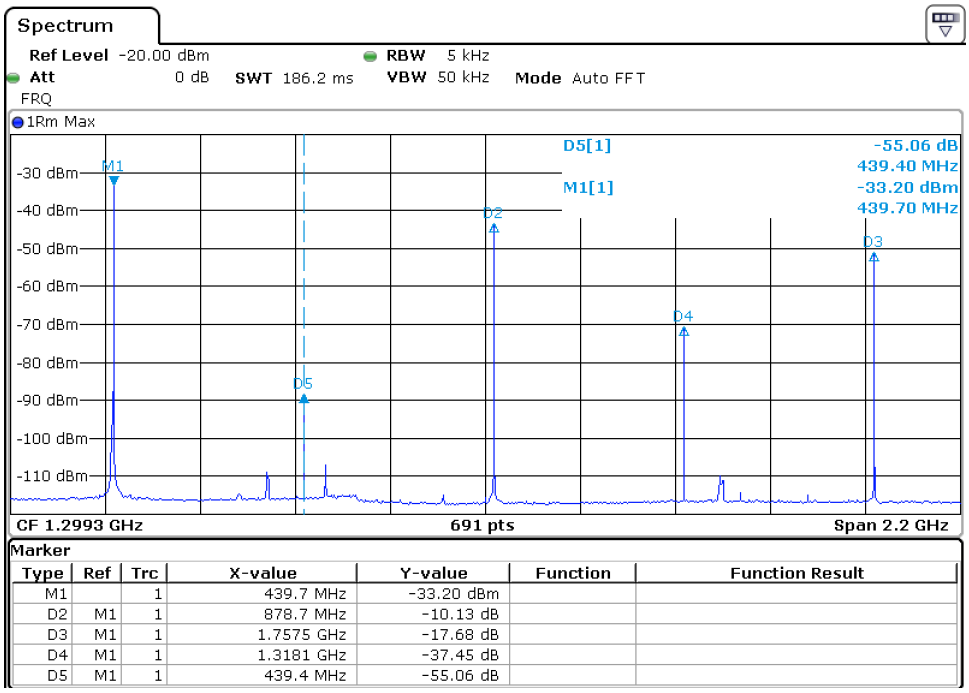
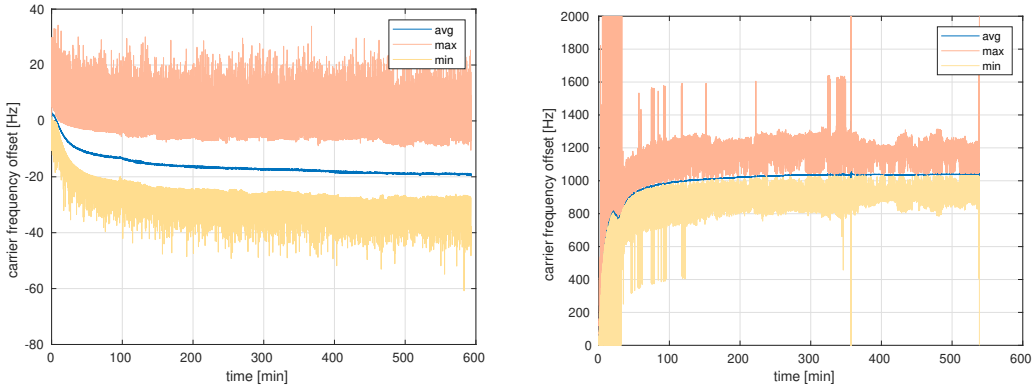


FIGURE 5.9: Spurious emissions of the basestation signal. Wanted signal is located at the M1 marker, all other markers represent intermodulation products of this signal.



(a) 0.5 ppm TCXO (b) 25 ppm TCXO

FIGURE 5.10: Estimated carrier frequency offset over time using the 0.5 ppm TCXO (a) and the 25 ppm TCXO (b).

5.2.2 *XO comparison*

After the start of our system, the carrier frequency should not deviate more than the maximum offset that our frequency offset estimator is able to detect. From the previous simulation, we know that this maximum offset is $f_{max} = 2200\,\mathrm{Hz}$. Thus, the minimal required oscillator accuracy must be $5\,\mathrm{ppm}$. The default TCXO that is mounted to the Pluto only has a accuracy of $25\,\mathrm{ppm}$, which confirms our theory that a better TCXO is required for the system to work. The $0.5\,\mathrm{ppm}$ TCXO is sufficiently accurate. In order to obtain real world measurements, we ran the basestation application on a Pluto with accurate TCXO and a client application on both the 25ppm TCXO and the 0.5ppm TCXO. Client and basestation stood 1 m apart, the basestation sends with maximum transmit power ($\sim 0\,\mathrm{dBm}$). The test is performed indoors at constant room temperature. The clients log the estimated frequency offset over time. Figure 5.10 shows the average, minimum and maximum estimated frequency offset over blocks of 1000 OFDM symbols. We can see that the frequency drifts the most in the first hour after the test started. This is most likely caused by the rising temperature within the Plutos, due to the system execution. Once the temperature settles, the frequency offset stays constant. However, we can see that the default TCXO is way more dependent on the temperature change, as it causes a drift of 1000Hz over time. The drift is in fact 50 times worse compared to the other TCXO, just as the indicated accuracy is 50 times worse. We can also see that the frequency drift itself seems to cause problems in the offset estimation, as the min and max estimations vary wildly at the beginning with the default TCXO. But also after the offset settled at about 1000Hz, the estimations have a greater deviation compared to the estimations with the accurate TCXO.

5.2.3 *Issues with the carrier synchronization*

During the system testing, some problems in the carrier frequency offset estimation arised. The offset estimation issues were observed first when using antennas and transmitting over the air.

The first problem can be described as follows: During the initial system start, the client's frequency offset estimation algorithm would lock to an unexpected frequency. After some time, the client's local oscillator is retuned to compensate the offset. After the retune, another, even larger offset was estimated. Analyzing the spectrum of the received baseband signal verified that the estimation algorithm was indeed correct. When introducing a carrier offset of f_e to the system, the receiver will see an offset of $-3 * f_e$, not the expected f_e. What is received in this case is the 3rd order intermodulation product instead of the original signal. The AD9361 transceiver on the Adalm Pluto is a direct conversion transceiver, it directly mixes the desired carrier frequency down to the baseband. Due to nonlinearities, the mixer will also mix down harmonics of the LO frequency. Thus, a signal at $3 \times 439.7\,\mathrm{MHz} = 1.32\,\mathrm{GHz}$ is also shifted to the baseband. We have seen in section 5.2.1 that our transceiver produces a quite strong 3rd intermodulation product at 1.32 GHz, which we can see again at the receiver. Depending on the used antenna, either the harmonic frequencies or the original signal is attenuated. In our case, antennas for the 2.4 GHz band were used, which attenuated our original signal. Only the upper harmonics are received. This problem can be solved by either ensuring that the antenna damps the unwanted intermodulation products, or by using a band-select filter at the transceiver. Since this filter will be allowed to have a very large transition window (440 MHz must be passed through, frequencies above 1.3 GHz should be blocked), the cost for such a filter is not high.

A second issue is related to the continuous frequency offset compensation. We use a numerically controlled oscillator (NCO) object in our application to remove the detected frequency offset in software. The oscillator is controlled by the frequency offset estimation, which is based on the estimated phase errors in pilot symbols. However, when the NCO is compensating a constant frequency offset, the phase estimation is not constant as expected. It follows a sine wave, were the frequency of the sine increases with the compensated frequency offset. Figure 5.11 shows this behaviour. If the carrier frequencies of the basestation and client match and the NCO is disabled, the phase is constant. No sine distortion is visible. This distortion is likely caused in the transceiver itself. It could be caused by some phase locked loop that generates the local oscillator frequency for the mixer, but further study is needed to find the exact cause of this.

Next, the frequency offset control loop is closed. The phase estimates are now used to adjust the compensation at the NCO object. Figure 5.12 shows the estimated phase errors and the frequency offset compensation over time. The oscillation in the phase offset causes an oscillation of the compensated carrier frequency offset. Tests show that the deviation around the actual frequency offset increases with the actual frequency offset. But in all cases the average frequency offset matches the real frequency offset. We use this to mitigate the oscillation in the compensation loop. After startup, each client measures the average frequency offset for a while. Then it retunes the transceiver to this offset. After the retuning the carrier frequencies of the basestation and client match closely. Under this condition, the oscillation in the phase estimation is negligible.

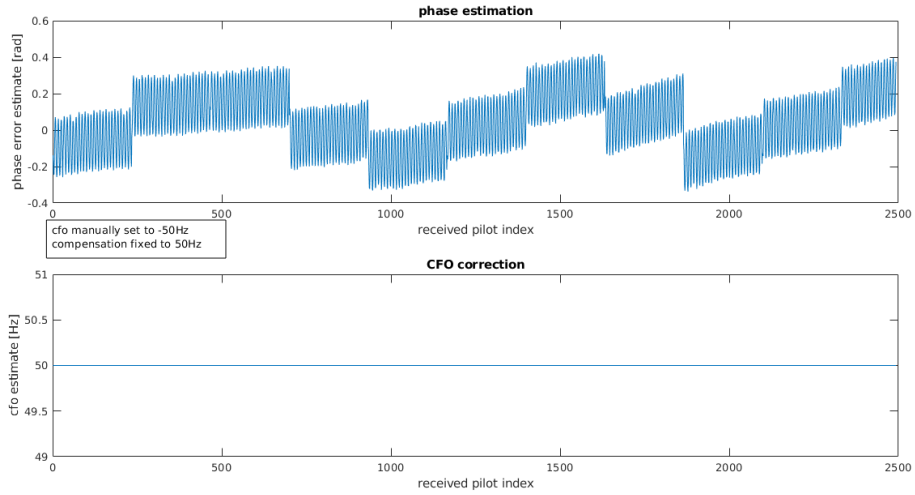


FIGURE 5.11: Phase error estimates over time (top) with a fixed frequency offset compensation of 50 Hz at the NCO (bottom). Discontinuities in the phase estimation are caused because the receiver object is initialized with the phase offset once per frame. Only the difference from this initial estimate is plotted.

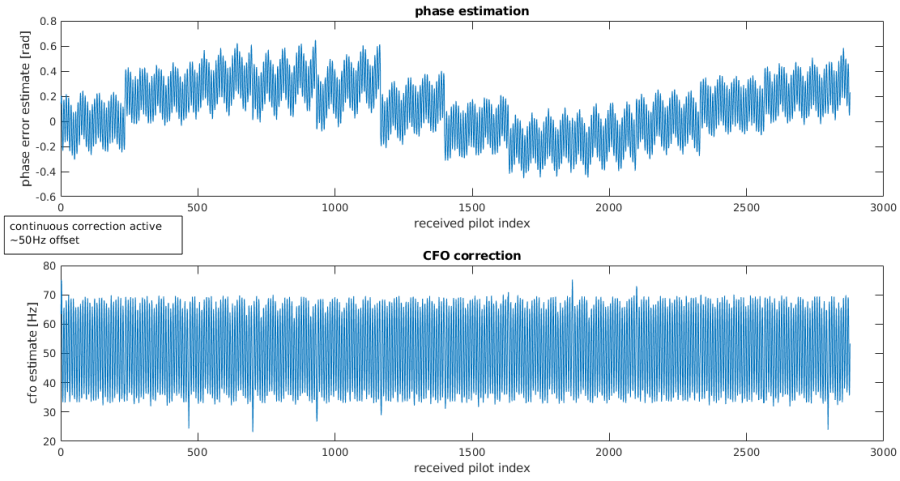


FIGURE 5.12: Top: Phase error estimates over time with the continuous frequency offset compensation. Bottom: Carrier frequency offset that is compensated using the NCO object. Both phase and frequency offset have a high frequent sinusoidal distortion.

MCS	total RX slot		demodulation		Viterbi		deinterleaver	
	avg	max	avg	max	avg	max	avg	max
0 (4QAM r=1/2)	463	745	109	192	290	342	35	96
1 (4QAM r=3/4)	654	808	106	183	479	649	35	74
2 (16QAM r=1/2)	1323	1955	614	1213	569	668	94	691
3 (16QAM r=3/4)	1704	2351	604	1220	946	1499	100	654
4 (64QAM r=1/2)	1887	2659	849	1482	844	943	137	237
5 (64QAM r=3/4)	2506	3302	792	982	1430	2026	143	630

TABLE 5.1: RX slot processing times for different MCS. All values in μs.

5.2.4 Performance Tests

We are going to measure the resource utilization of the Pluto, since this will enable us to estimate whether scaling the system to wider bandwidths is possible. Being able to deploy the system on other bandwidths was one of the desired features that we derived earlier. To assess the system utilization, we ran an iperf test and measured the processing times at the receiver plus the processing time to run the scheduler and prepare the samples. We measure the average and maximum delay over 10000 iterations. RX slot processing includes demodulation, Viterbi decoding, deinterleaving and processing at the MAC layer (Reassembling, forwarding to TAP device). The Viterbi decoder operates on soft bits and uses our implemented arm Neon SIMD optimization. The timings are shown in Table 5.1. It can be seen that Viterbi decoding takes most of the time, followed by the demodulation. Interesting is the 6 times increase in the demodulation time when changing from 4QAM to 16QAM. We can also see that the maximum values are spiking in some cases. We did not expect this, since the system already runs on a realtime Linux with FIFO scheduling and high priority. Our system is written in a way that memory allocations are performed frequently, the latency spikes might be caused by paging. Another cause could be the DMA transfer of samples between the FPGA and the ARM system. The preparation of a TX slot consists of convolutional coding, interleaving and modulation to complex samples. The timing for a TX slot is given in Table 5.2. We see that preparing a slot is not as computationally extensive as receiving the slot. Most of the time is used to modulate the complex samples. From the maximum values we can see that the processing time is spiking in some cases, similar to the RX slot processing. At maximum load, the basestation has to transmit and receive in every slot. One slot lasts $\sim 3.9\,\mu$s, the combined TX and RX slot processing times should not exceed this limit, since these tasks are executed on the same CPU core. This shows us that the system is already pushed to the limit when MCS5 is used. It will not be possible to deploy the system on wider bandwidths, without additional optimization of the Viterbi and/or the modulation.

Next, we check the processing time of the FFT and the channel equalization. These tasks are performed by the threads that transfer the samples between the FPGA. There are always two OFDM symbols transferred within one buffer element, the maximum processing time is thus $136T_s = 530\,\mu$s. The receive time is on average $109\,\mu$s, at maximum $225\,\mu$s. Preparing the buffer takes $30\,\mu$s on average with a maximum of

MCS	total TX slot		modulation		encoding		interleaver	
	avg	max	avg	max	avg	max	avg	max
0 (4QAM r=1/2)	244	353	175	255	46	198	16	97
1 (4QAM r=3/4)	296	413	138	225	121	205	29	102
2 (16QAM r=1/2)	389	500	241	312	96	172	44	140
3 (16QAM r=3/4)	527	675	248	335	242	339	29	78
4 (64QAM r=1/2)	553	744	347	423	157	285	41	89
5 (64QAM r=3/4)	747	841	338	423	343	439	56	123

TABLE 5.2: TX slot processing times for different MCS. All values in μs.

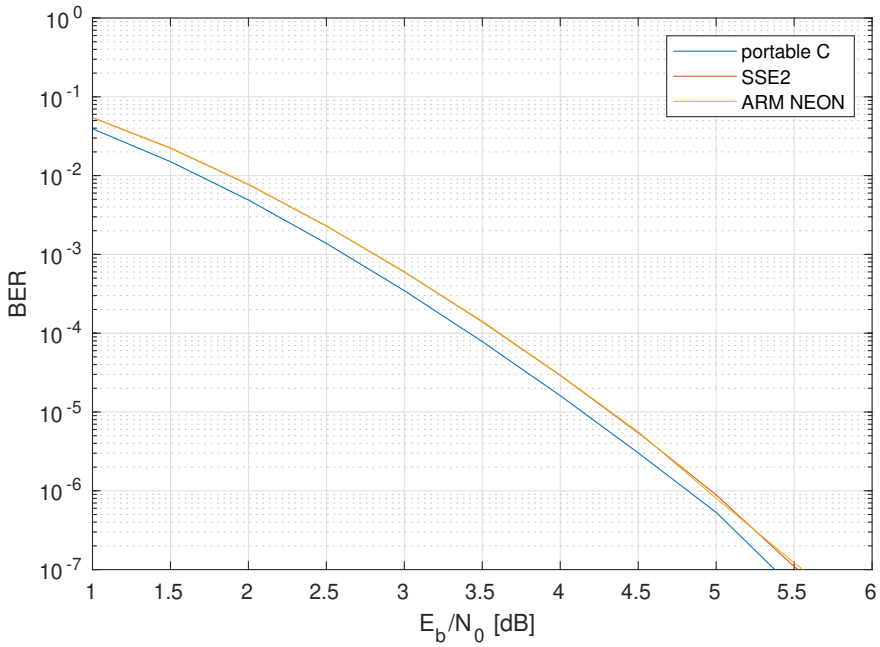


FIGURE 5.13: Decoding performance of the platform independent, the SSE2 optimized and the ARM Neon optimized viterbi decoder.

150 μs. The average from the TX side does not represent the correct FFT processing time, as there some slots in which no data is being sent and no FFT performed. These lower the average. Receiving the buffer takes more time than preparing it, since the carrier frequency offset and the phase and amplitude of all subcarriers have to be compensated. Using the average values, it is clear that there is enough processing time left that could be used when deploying the system with wider bandwidths and e.g. a 128 point FFT.

At last, we evaluate the performance gain of our Viterbi implementation. By default, the libfec library uses platform independent C code when executed on ARM systems. To assess the performance, a test tool *test27* is part of the library. We simulated using 10^9 bits to validate our implementation, the results are given in Figure 5.13. Our ARM Neon software port performs as good as original SSE2 optimized version.

MCS idx	UDP segment size	UDP data rate
0	1408	86kbps
1	1366	130kbps
2	1398	184kbps
3	1414	284kbps
4	1422	284kbps
5	1343	434kbps
6	1428	380kbps

TABLE 5.3: UDP test configuration for different MCS.

The portable C version has a gain of 0.2 dB compared to the optimized versions. This is likely due to the higher quantization of metrics within the decoder. The ARM Neon code provides us a factor of 2.8 speedup, the SSE2 optimization has a speedup of 7.57 (executed on Intel i5-4200u CPU). So the ARM CPU cannot utilize the SIMD instructions as good as the Intel CPU utilizes the SSE2 instructions, even though both work on 128bit vector registers. Still, when looking at the slot processing times we see that the optimization is required to be able to run the MCS3 and higher.

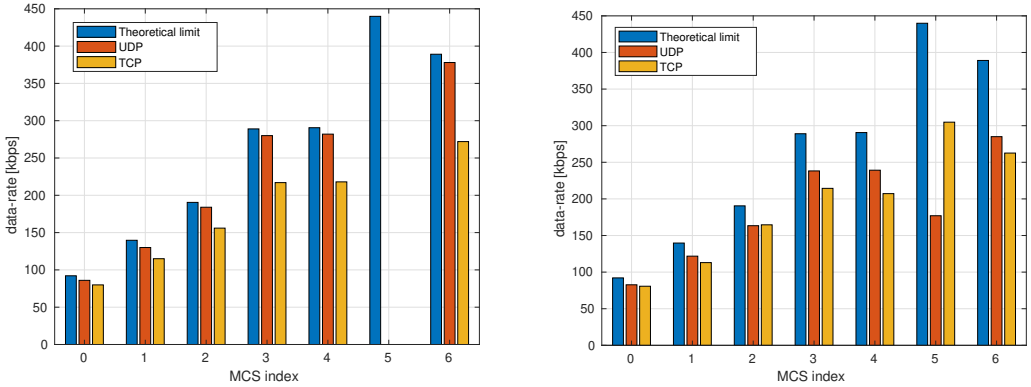
5.2.5 Application Layer Tests

At last we will determine the overall performance of the system by transferring higher layer data. We use the tool iperf3[1] to generate TCP and UDP data streams between the Adalm Plutos. For each measurement, iperf was run for 5 minutes. The Plutos are connected via HF cable, the transmit power is set to -20 dBm. When testing UDP, the parameters given in Table 5.3 were used. The maximum theoretical data rate can be calculated by taking the slot payload size times the number of slots per frame, divided by the frame duration:

$$R_{max} = \frac{\text{slotsize} \cdot 27}{136\,\text{ms}}. \tag{5.5}$$

There are 32 slots per frame, but only 27 can be assigned to a single user due to the half duplex constraint. The payload sizes of the PHY slots for different MCS have been listed in Table 3.6. The logical channel and MAC layer header require another 5 Bytes per slot. For UDP tests we set the packet size including all headers to match a multiple of our MAC data message size. This way, the best slot utilization is achieved. When testing TCP, the MSS size was kept at the default, as this depicts a more relevant scenario. Figure 5.14 compares the TCP, UDP and theoretical data-rates for different MCS. Using UDP, the theoretical limit is nearly met. TCP data-rates are lower, since acknownledgements have to be transmitted. During the ACK transmission, no data can be transferred due to the half duplex property. Furthermore the Ethernet frame size is not matched to our MAC message size. We could not get consistent results when using MCS 5, sometimes no transmission was possible, sometimes the frame error rate was close to 50%. The Plutos are not reliably keeping up with the coding and decoding. As our previous timing measurements show, MCS 5 is hitting the computational limits of the Plutos. In the uplink, the achieved data-

[1] https://iperf.fr/



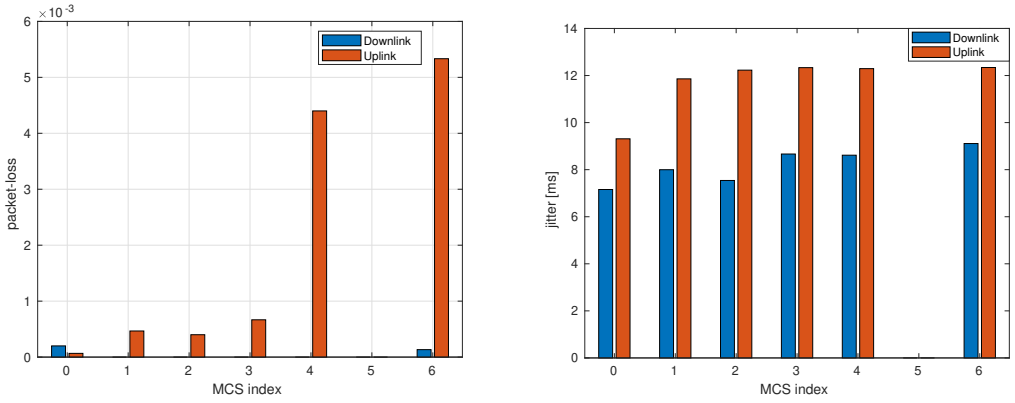
(a) Downlink data-rates (b) Uplink data-rates

FIGURE 5.14: Comparison of the theoretical and measured data-rates in the downlink (a) and uplink (b).

rates are slightly lower, due to uplink requests that are additionally transmitted. They lower the payload size that can be used for the Ethernet frame per slot. In our UDP tests, the packet size is thus not perfectly matched anymore, lowering the efficiency.

Next, we will use iperf to generate VoIP like traffic. We set the UDP packet size and desired data-rate according to the VoIP packetization of the G.729 codec. The data-rate is fixed to $R = 8\,\mathrm{kbps}$, the packet size is 20 Bytes. We start an iperf client both at the basestation and the client. The VoIP simulation runs for 5 minutes, during this time 15.000 Packets are transmitted in each direction. Figure 5.15 shows the packet loss and the calculated jitter during the test. Interestingly, the measured jitter is lower than what we would expect from the simulation. This is because apparently iperf uses a different definition of jitter. RFC3393 states that jitter is both used to describe the packet delay variance and the instantaneous packet delay variation. The second is defined as the maximum delay difference between two consecutive packet arrivals and used by iperf. Nevertheless, the packet loss is well below the maximum packet loss of 1% that VoIP applications require.

At last, we evaluate the delay of the system using the *ping* command. *Ping* will not reveal separate latencies for uplink and downlink, but only the round-trip time in our system. Using the results for MCS0 from the previous simulation shown in Figure 5.6, we expect an average round-trip time of 137 ms. After 3600 transmitted packets, the average measured round-trip time is 145.1 ms, slightly more than expected. This is because the waiting time after the basestation received the packet to the transmission of the reply is not random anymore. The measured jitter is 39.9 ms, slightly less than the expected 47 ms. Again, this is due to the reply not being inserted at a random point of time to the basestation MAC layer. These measurements confirm that the developed system can be used for VoIP applications.



(a) packet-loss (b) Jitter

FIGURE 5.15: VoIP traffic test for different MCS, (a) shows the packet-loss during the test, the jitter is shown in (b).

6

Conclusion

Throughout this thesis, the system design and the implementation of a new communication system for a 200 kHz frequency division duplex channel in the 70 cm band was presented. The implementation is fully integrated on the Adalm Pluto SDR platform.

After weighting the advantages and disadvantages of an OFDM based system we chose to implement an OFDM system with 4 kHz subcarrier spacing and a FFT size of 64. The usage of OFDM makes it easier to deploy the system with other bandwidths and at other frequency bands. Therefore, the system design and implementation was done in a way that only the subcarrier allocation and FFT size has to be redefined to support different channel bandwidths. The usage of OFDM happens at the cost of increased signal processing efforts. The narrow subcarrier spacing requires a finer carrier frequency offset estimation compared to a single carrier system. To accomplish this, we built a compensation loop based on tracking the phase of BPSK pilot tones. At the MAC layer, the design focus was to achieve low latency in order to support VoIP. Our approach was to select a rather short frame structure with short scheduling periods. This way, no sophisticated proactive reservation of resources has to be designed. Our scheduler is purely based on the client's queue length, it schedules slots at a period of 17 ms. This guarantees that VoIP and other low latency streaming applications are possible. In order to support such small delays at the SDR implementation level, we integrated all functionality directly on the ARM CPU of the Adalm Pluto SDR.

Our implementation approach was to realize the complete system and all signal processing in C language. The application uses a platform abstraction layer, which either connects to the SDR transceiver or to a simulation environment. This helped to speed up the implementation process, since the same code could be used for simulating the system and for deployment on the SDR itself. During the implementation, we first struggled to achieve stable real-time performance of the system. The problems could be solved by using a realtime Linux kernel and taking care of distributing the MAC and PHY layer processing across the CPU cores of the SDR. We furthermore had to use SIMD optimization to speed up the Viterbi decoding.

We then extensively studied the throughput and the delay of the system. First tests show that our system design is capable of delivering up to 400 kbps application layer data-rate and a round-trip time of less then 150 ms. At most 14 users can connect to the basestation at the same time. This will allow multiple users to use VoIP services simultaneously. However, our tests so far were all performed under laboratory conditions. The SDRs were connected via HF-cables, or when using antennas they stood in the same room during the test. Additional real world tests with greater distances have to be performed to properly assess whether the system works as expected. Our tests

do not cover the robustness of the system against higher delay spreads, inter-symbol interference, and the handling of higher propagation delays.

The real world tests will require a proper power amplifier. Our measurements showed that the power output at the Adalm Pluto is only -6 dBm, less then what we expected. It will be hard to find amplifiers that are sensitive enough to automatically trigger on this power output. Therefore an additional push-to-talk signal has to be generated at the SDR which activates the amplifier. On top of that, the impact of the higher PAPR due to OFDM is significant for the maximum power output we can achieve. If the PAPR is ~ 10, a 150 W power amplifier is required to yield the desired output of 15 W. Still, also tests with lower than expected output power will reveal whether the estimated link distances can be achieved in the real world.

Finally, we will present further implementation enhancements that could be of interest for further studies. When measuring the computational load on the ARM CPU, we saw that with 64-QAM and $3/4$ rate convolutional coding, no stable operation is possible. Especially the demodulation of the QAM signal and the Viterbi decoder took up too much time. The FPGA on the Adalm Pluto could be used to offload the Viterbi decoding process. This would free computational resources and allow the use of high MCS also on wider system bandwidths. Furthermore, the Rayleigh channel simulations showed that it could be beneficial to run $r = 1/3$ codes with QAM-16 or QAM-64 modulation instead of higher rate coding with lower order modulation. Testing such combinations of modulation and coding would be interesting to see. Next, the channel estimation for each subcarrier is currently done by linear interpolation of the phase and gain of the pilot tones. Switching to quadratic or cubic interpolation should benefit the BER, especially if wider bandwidths are used. Furthermore, there is no interpolation performed in the time domain. The estimates are kept the same until the next pilot tones are received. Interpolating the time domain should also make the estimation more accurate. The computational costs of this should not be too high. Due to the significance of the high PAPR it would be interesting to have a look at PAPR reduction approaches that have been developed for OFDM. A reduction of the PAPR will enable higher transmit powers.

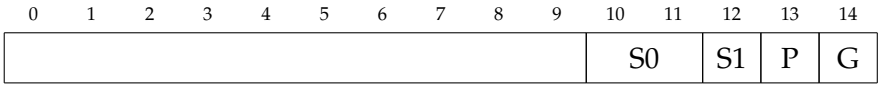
Appendices

A

PHY SLOT DEFINITIONS

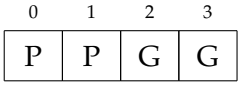
In the following we will present the definition of the different slot types and how higher layer data is transported via them.
The physical layer is based on OFDM. The basic unit of each PHY layer slot is one OFDM symbol length, which equals $68/256000\,\mathrm{s} = 265.625\,\mu\mathrm{s}$. The subcarriers within an OFDM symbol are either fully allocated for higher layer data or contain a pattern of pilot symbols. The exact definition was presented in Figure 3.3. In the following we use the shortname P for a OFDM symbol containing pilots and D for OFDM symbols solely consisting of data subcarriers. G shall denote a guard interval of one OFDM symbol length in which no transmission takes place.

A.1 SYNCHRONIZATION SLOT



The synchronization slot has a duration of 14 OFDM symbols. It is located in subframe 0, in the OFDM symbols 49 to 63 of the downlink. $S0$ and $S1$ represent a special synchronization sequence for the Schmidl & Cox synchronization. Then an OFDM symbol including pilots follows.
The data subcarriers in this slot are modulated using MCS0 (QPSK) and encoded using 1/2 code rate. This gives 3 Bytes of payload data that can be transmitted during the slot. This is used to transmit a logical channel with 8bit CRC. The logical channel carries information on the transmit power of the basestation in the first byte and the current receive sensitivity of it in the second byte. The client uses this information to initially set its transmit power in the uplink.

A.2 DOWNLINK CONTROL SLOT



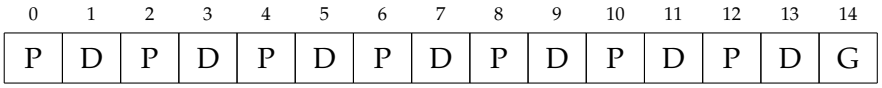
The downlink control slot has a duration of 2 OFDM symbols. Both symbols contain pilot tones to make this slot the most robust. This slot is used to define the allocation of the 4 downlink data slots, 4 uplink data slots and 2 uplink control slots in the current subframe. It is located in OFDM symbol 0 and 1 within a subframe. Two guard intervals separate it from the next slot.
The subcarriers of this slot are modulated using MCS0. This gives 6 Bytes payload.

MCS idx	modulation	coding rate	logical channel size
0	QPSK	1/2	62
1	QPSK	3/4	93
2	16QAM	1/2	125
3	16QAM	3/4	187
4	64QAM	1/2	188
5	64QAM	3/4	277
6	256QAM	1/2	256

TABLE A.1: MCS definitions and the resulting payload size in Bytes.

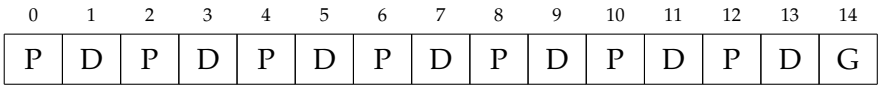
This is used to host a logical channel with 8bit CRC. The remaining 5 Bytes are allocated as follows: Each Byte is split up into four bit fields. The first 4 fields contain the user ID to which the downlink data slots in this subframe have been allocated. The next 4 fields contain the assigned user IDs of the uplink data slots, the last two fields contain assigned user IDs for the uplink control slots. If a slot has not been allocated, the corresponding field is set to 0x0. Before the data is encoded, scrambling of the logical channel bytes is performed. This increases robustness if many fields in the slot are set to zero.

A.3 DOWNLINK DATA SLOT



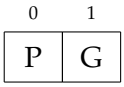
Each downlink data slot consists of 14 OFDM symbols with data transmission, followed by a guard interval. Every second OFDM symbol contains pilot tones. The slot is modulated using an MCS from table A.1. This table also states the resulting sizes for the logical channel. Downlink data slots use logical channel objects with 16bit CRC. There are four downlink data slots per subframe.

A.4 UPLINK DATA SLOT



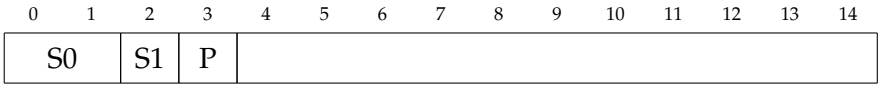
Uplink data slots are defined similar to downlink data slots. Each uplink data slot consists of 14 OFDM symbols with data transmission, followed by a guard interval. Every second OFDM symbol contains pilot tones. The slot is modulated using an MCS from table A.1. Uplink data slots use logical channel objects with 16bit CRC. There are four uplink data slots per subframe.

A.5 UPLINK CONTROL SLOT



Uplink Control slots consist of only a single OFDM symbol including pilot tones followed by a guard interval. They are modulated using MCS0, giving 3 Bytes of payload data per slot. This is filled with a logical channel using 8 bit CRC. There are two uplink control slots per subframe.

A.6 RANDOM ACCESS SLOT



The random access slot is used by clients to get first access to the system. It is located in subframe 0 in each frame. The overall length is 15 OFDM symbols, the clients are expected to start sending at the beginning of the slot. The clients start by sending the same synchronization sequence that is transmitted by the basestation in the synchronization slot. This is used by the basestation to calculate the timing advance. Then a single OFDM symbol including pilot tones follows. It is modulated with MCS0, giving 3 payload bytes. They are used to transmit a logical channel with 8 bit CRC. The first data byte of the channel is used to transmit a temporary user-ID called *rachuserid*. The second byte is used to transmit the number of previous association requests.

B

MAC MESSAGES DEFINITIONS

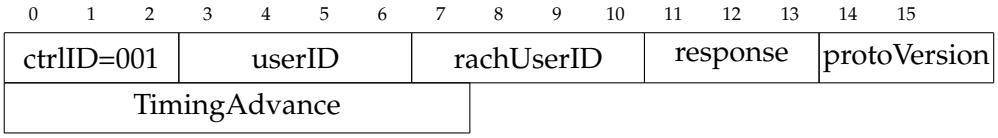
Multiple control messages and the header for data messages have been defined as part of the system design. Their purpose and the data which is exchanged is listed here.

B.1 MAC LAYER MESSAGES SENT BY THE BASE STATION

One or multiple MAC Layer messages are mapped to a Data Channel or Control Channel. Mapping is described in the previous section.

B.1.1 associate_response

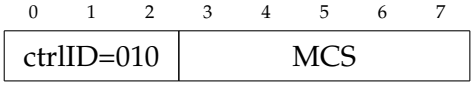
This message is sent by the BS through the Broadcast Channel, to answer an UE's association request.



rachUserID is the random ID which was transmitted by the UE in the Random Access Burst.
userID is the ID which will identify the UE from now on. Is only set if response field is 0x0, aka success.
response Indicates the status of the association. Is set to 0x0 if the user was successfully added. Set to 0x1 if the network is full (no free user ID). Other states can be defined later.
protoVersion indicates which version of the protocol is used by the base station. This protocol described here is Version 0.
TimingAdvance the calculated timing advance. It is sent even though a separate TimingAdvance message exists. The UE initially does not know the userid and we cannot guarantee that BS sends the AssocResponse before the TimingAdvance message which requires awareness of the userid.

B.1.2 dl_mcs_info

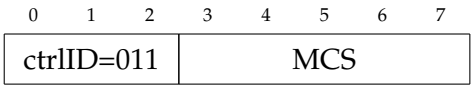
This message is sent by the BS through the Data Channel. It is used to tell a UE which MCS will be used from now on. The UE shall send a control_ack packet upon reception of this info. The BS will change to the new MCS as soon as it receives the ack. If no ack is received, it can send the dl_mcs_info packet again.



MCS is the Modulation Coding Scheme index that will be used.

B.1.3 ul_mcs_info

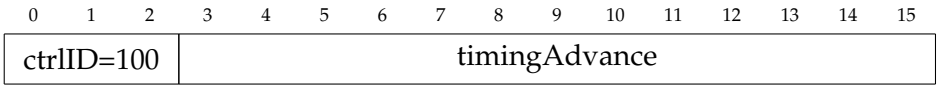
This message is sent by the BS through the Data Channel. It is used to tell a UE which MCS will be used from now on. The UE shall send a control_ack packet upon reception of this info. The BS will change to the new MCS as soon as it receives the ack. If no ack is received, it can send the ul_mcs_info packet again.



MCS is the Modulation Coding Scheme index that will be used.

B.1.4 timing_advance

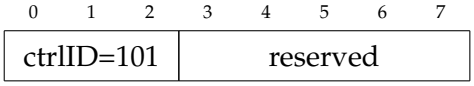
This message is sent by the BS through the Data Channel. It is used to tell a UE which timing advance should be used starting with the next slot. The TA is valid until the UE receives the next timing_advance message. The round trip time at a distance of 30km is 200µs. If the timing advance is signaled in integer multiples of the symbol time $T_s = 3.90625\mu$s, at least 52 steps are needed, which equals a 6 bit integer. We do not want to limit maximum system range by the timing advance, but by the antenna equipment that is used and the channel properties. Furthermore, control messages shall be fixed on byte size. Therefore, 13bit timingAdvance signaling was chosen, even if a smaller number would be sufficient.



userID identifies the user.
timingAdvance defines the timing advance. It is transmitted as an unsigned integer and has to be multiplied with the symbol time $T_s = 3.90625us$ to get the timing advance in seconds.

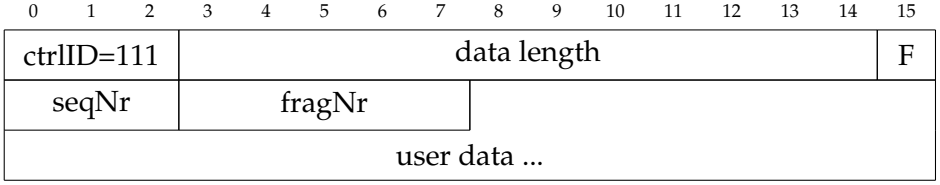
B.1.5 session_end

This message is sent by the BS through the Data Channel. It is used to tell a UE that the connection will be terminated, e.g. because the BS is not able to decode any data from the user anymore. Upon reception of this message, a UE must reset its connection state and stop using the currently assigned userid. It can obtain a new userid by undergoing the initial access procedure.



B.1.6 dl_data

This message is sent by the BS through the Data Channel. It is used to transmit data from higher layers. For more details on fragmentation, see the corresponding section.



data length data size in bytes.
seqNr sequence number of the packet.
fragNr fragment number within a packet.
F Final flag. is set if this is the last fragment of a packet.
user data carries the data of variable size.

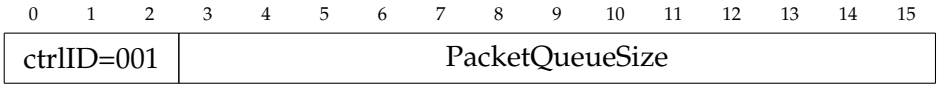
B.2 MAC LAYER MESSAGES SENT BY THE UE

B.2.1 Random Access Burst

For initial Access to the system the UE can send a random access burst. Since neither the UE nor the base station have any information on frequency offset, symbol timing etc., a preamble has to be prepended in front of the control information. We use the same preamble as in the synchronization message: Three OFDM symbols, where the first two are identical. As for control information, the UE selects a random 4 bit *rachUserID* which acts as a temporary identification. This information can be packed into one OFDM symbol.
This message is mapped to the Physical Random Access Channel.

B.2.2 ul_req

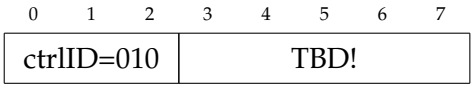
This message is sent by a UE through the Control Channel OR the Data Channel. It is used to request a data transmission in the Uplink.



PacketQueueState is the size of the data packet queue in bytes.

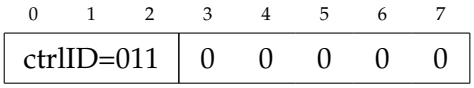
B.2.3 channel_quality

This message is sent by the UE through the Control Channel or the Data Channel. It is used to give information on the downlink channel quality. This will be used by the BS to select the proper Modulation scheme. To be done later!



B.2.4 keepalive

This message is sent by the UE through the Control Channel or the Data Channel. It is used if the user was granted an uplink control or data channel, but has no other packets in the queue. It serves as a keepalive acknowlegdement.



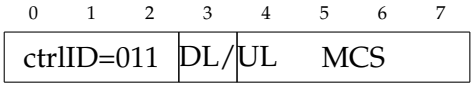
B.2.5 control_ack

This message is sent by the UE through the Control Channel or the Data Channel. It is sent if the UE received some control message in the downlink which requires an acknowledgement (mcs_info, timing_advance).

ackedCtrlID the control ID of the message that is acknowledged.

B.2.6 mcs_change_req

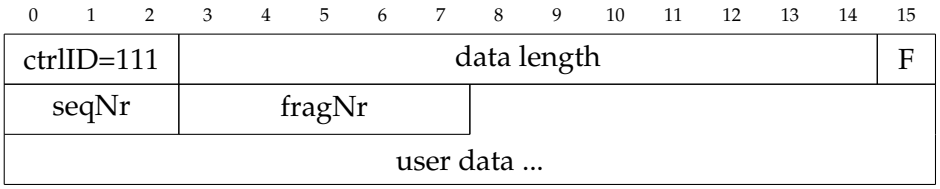
This message is sent by the UE through the Data Channel. It is used to actively request a change of the MCS from the BS. Upon reception the BS shall send a ul/dl_mcs_info packet to initiate the MCS change procedure.



DL/UL 0=request change for Downlink MCS. 1=request change for Uplink MCS.
MCS is the Modulation Coding Scheme index that is requested.

B.2.7 ul_data

This message is sent by the UE through the Data Channel. It is used to transmit data from higher layers.



data length data size in bytes.
seqNr sequence number of the packet.
fragNr fragment number within a packet.
F Final flag. is set if this is the last fragment of a packet.
user data carries the data of variable size.

C

Network Configuration

The figure below gives a example network configuration of a HAMNET backbone site that provides user access using the developed protocol.
The backbone site in the figure consists of at least one access point that provides a point-to-point link to another HAMNET site. A BGP router is used to route the traffic between the links. An Adalm Pluto operating as a basestation is also connected to the BGP router to provide the 70 cm access. The Adalm Pluto hosts a DHCP server and routes all traffic to the BGP router by default.
We show two approaches to connect to this basestation as a client. In the first approach "Routing+NAT", an Adalm Pluto acts like a home-router. The Plutos TAP device fetches a DHCP IP from the Adalm Pluto basestation, the ethernet device uses a static IP from the LAN at home. The Pluto hosts a network address translation service to forward traffic from the LAN to the HAMNET. All clients in the 192.68.178.1/24 network can access the HAMNET as the Home-Router knows that the 44.0.0.1/9 network is reachable via the Pluto.
In the second approach, the *tap* and *ethernet* device of the Adalm Pluto client are bridged. Thus, the device that is connected to the Pluto via USB can directly fetch a HAMNET IP-address via DHCP and access the HAMNET.

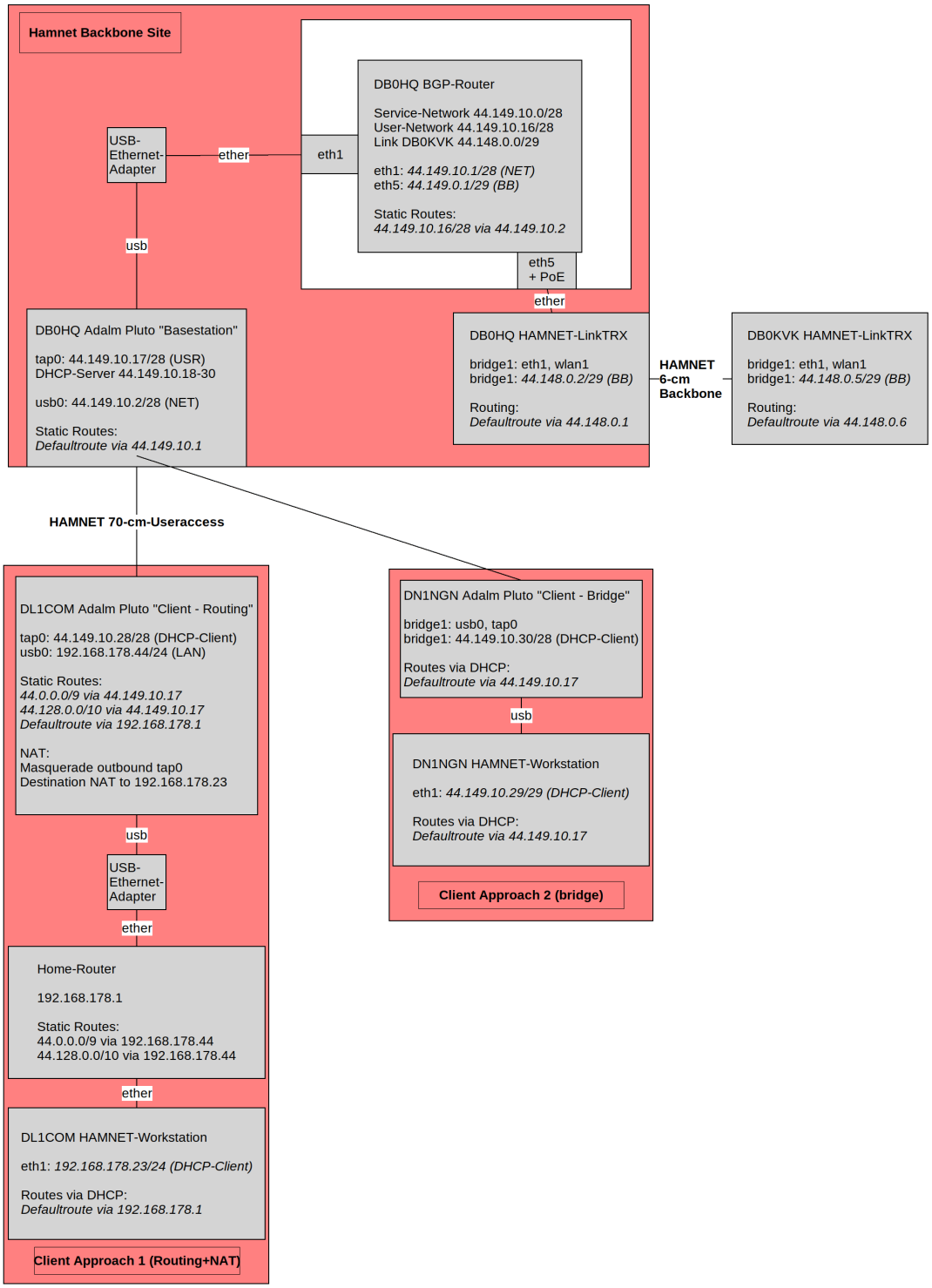


FIGURE C.1: Example configuration of a HAMNET site that provides access using our developed system.

D

ABBREVIATIONS

ADC Analog to Digital Converter

ARP Address Resolution Protocol

ARQ Automatic Repeat Request

BER Bit Error Rate

CDMA Code Division Multiple Access

CRC Cyclic Redundancy Check

CSMA Carrier Sense Multiple Access

FDMA Frequency Division Multiple Access

FFT Fast Fourier Transform

FPGA Field Programmable Gate Array

GFSK Gaussian Phase Shift Keying

HAMNET Highspeed Amaterurradio Multimedia NETwork

IP Internet Protocol

ITU International Telecommunication Union

LO Local Oscillator

LOS Line-of-Sight

LTE Long Term Evolution

MAC Medium Access Control

NCO Numerically Controlled Oscillator

OFDM Orthogonal Frequency Division Multiplex

PN Pseudonoise

QAM Quadrature Amplitude Modulation

SDR Software-defined Radio

SDU Service Data Unit

SIMD Single Instruction Multiple Data

SNR Signal to Noise Ratio

TCP Transmission Control Protocol

TCXO Temperature Controlled Crystal Oscillator

TDMA Time Division Multiple Access

UDP User Datagram Protocol

VoIP Voice over Internet Protocol

BIBLIOGRAPHY

[1] "Hamnetdb website," https://hamnetdb.net, [Accessed: 14.04.2020].

[2] R. Wilke, H. A. Munz, and D. Heberling, "Radio Coverage Prediction for a Wireless IP-based Network in Central Europe," 2014.

[3] J. G. Proakis and M. Salehi, *Digital communications*, 4th ed. McGraw-Hill New York, 2001.

[4] J. S. Seybold, *Introduction to RF Propagation*. John Wiley & Sons, Ltd, 2005.

[5] T. S. Rappaport *et al.*, *Wireless communications: principles and practice*, 2nd ed. Prentice Hall, 2002.

[6] H. Asplund, A. A. Glazunov, A. F. Molisch, K. I. Pedersen, and M. Steinbauer, "The COST 259 directional channel model part I: Overview and Methodology," *IEEE Transactions on Wireless Communications*, vol. 5, no. 12, p. 3421, 2006.

[7] ETSI, "Digital cellular telecommunications system (Phase 2+); Radio Transmission and Reception (ETSI TS 100 910 V8.20.0)," European Telecommunications Standards Institute, 2005.

[8] ETSI, "LTE; Evolved Universal Terrestrial Radio Access (E-UTRA); User Equipment (UE) radio transmission and reception (3GPP TS 36.101 Version 14.3.0 Release 14)," European Telecommunications Standards Institute, 2017.

[9] A. Algans, K. I. Pedersen, and P. E. Mogensen, "Experimental analysis of the joint statistical properties of azimuth spread, delay spread, and shadow fading," *IEEE Journal on selected areas in communications*, vol. 20, no. 3, pp. 523–531, 2002.

[10] L. J. Greenstein, V. Erceg, Y. S. Yeh, and M. V. Clark, "A new path-gain/delay-spread propagation model for digital cellular channels," *IEEE Transactions on Vehicular Technology*, vol. 46, no. 2, pp. 477–485, 1997.

[11] H. Asplund, A. A. Glazunov, A. F. Molisch, K. I. Pedersen, and M. Steinbauer, "The COST 259 directional channel model-part II: macrocells," *IEEE Transactions on Wireless Communications*, vol. 5, no. 12, p. 3434, 2006.

[12] W. C. Jakes and D. C. Cox, *Microwave mobile communications*. Wiley-IEEE Press, 1994.

[13] A. C. V. Gummalla and J. O. Limb, "Wireless medium access control protocols," *IEEE Communications Surveys & Tutorials*, vol. 3, no. 2, pp. 2–15, 2000.

[14] A. S. Tannenbaum, *Computer networks*, 4th ed. Prentice Hall, 2003.

[15] S. Parkvall, A. Furuskar, and E. Dahlman, "Evolution of LTE toward IMT-advanced," *IEEE Communications Magazine*, vol. 49, no. 2, pp. 84–91, 2011.

[16] J. G. Proakis and M. Salehi, *Fundamentals of communication systems*, 2nd ed. Prentice Hall, 2014.

[17] M. S. El-Tanany, Y. Wu, and L. Házy, "OFDM uplink for interactive broadband wireless: analysis and simulation in the presence of carrier, clock and timing errors," *IEEE Transactions on Broadcasting*, vol. 47, no. 1, pp. 3–19, 2001.

[18] T. M. Schmidl and D. C. Cox, "Robust frequency and timing synchronization for OFDM," *IEEE transactions on communications*, vol. 45, no. 12, pp. 1613–1621, 1997.

[19] J.-J. Van De Beek, O. Edfors, M. Sandell, S. K. Wilson, and P. O. Borjesson, "On channel estimation in OFDM systems," in *1995 IEEE 45th Vehicular Technology Conference. Countdown to the Wireless Twenty-First Century*, vol. 2. IEEE, 1995, pp. 815–819.

[20] Bundesnetzagentur, "Frequenznutzungsplan Oktober 2019," www.bundesnetzagentur.de/DE/Sachgebiete/Telekommunikation/Unternehmen_Institutionen/Frequenzen/Grundlagen/Frequenzplan/frequenzplan-node.html, 2019, [Accessed: 23.03.2020].

[21] "Gesetz über den Amateurfunk (Amateurfunkgesetz - AFuG 1997)," BGBl. I S. 1494.

[22] Deutscher Amateur Radio Club e.V., "Bandplan 70cm - Stand 2017," www.darc.de/der-club/referate/vus/bandplaene/, [Accessed: 23.03.2020].

[23] Bundesministerium für Wirtschaft und Arbeit, "Verordnung zum Gesetz über den Amateurfunk (Amateurfunkverordnung - AFuV)," 2005, (BGBl. I S. 242).

[24] ITU, "Rec. ITU-R SM.1541-6: Unwanted emissions in the out-of-band domain," International Telecommunication Union, 2015.

[25] B. Goode, "Voice over Internet protocol (VoIP)," *Proceedings of the IEEE*, vol. 90, no. 9, pp. 1495–1517, Sep. 2002.

[26] Cisco, "VoIP - Per Call Bandwidth Consumption," www.cisco.com/c/en/us/support/docs/voice/voice-quality/7934-bwidth-consume.html, [Accessed: 18.11.2019].

[27] "NPR New Packet Radio project page," https://hackaday.io/project/164092-npr-new-packet-radio, [Accessed: 18.10.2019].

[28] F4HDK, "Ham Radio Does Distant Data Networking," https://spectrum.ieee.org/geek-life/hands-on/build-a-longdistance-data-network-using-ham-radio, Nov 2019.

[29] F. Capozzi, G. Piro, L. A. Grieco, G. Boggia, and P. Camarda, "Downlink packet scheduling in LTE cellular networks: Key design issues and a survey," *IEEE communications surveys & tutorials*, vol. 15, no. 2, pp. 678–700, 2012.

[30] Y. . E. Wang, X. Lin, A. Adhikary, A. Grovlen, Y. Sui, Y. Blankenship, J. Bergman, and H. S. Razaghi, "A Primer on 3GPP Narrowband Internet of Things," *IEEE Communications Magazine*, vol. 55, no. 3, pp. 117–123, March 2017.

[31] R. Ratasuk, B. Vejlgaard, N. Mangalvedhe, and A. Ghosh, "NB-IoT system for M2M communication," in *2016 IEEE wireless communications and networking conference (IEEE WCNC)*. IEEE, 2016, pp. 1–5.

[32] "Programm der Amateurfunktagung München 2018," www.darc.de/der-club/distrikte/c/amateurfunktagung-muenchen/, 2018, [Accessed: 23.03.2020].

[33] "liquid-sdr project page," https://liquidsdr.org/, [Accessed: 05.04.2020].

[34] "Charon project page," https://github.com/tvelliott/charon, [Accessed: 05.04.2020].

[35] Analog Devices, "Adalm-Pluto wiki," www.wiki.analog.com/university/tools/pluto/users/receiver_sensitivity, [Accessed: 24.10.2019].

[36] ETSI, "Fixed Radio Systems: Parameters affecting the Signal-to-Noise Ratio (SNR) and the Receiver Signal Level (RSL) threshold in point-to-point receivers (ETSI TR 03 053 V1.1.1)," European Telecommunications Standards Institute, 2014.

[37] C. Berrou and A. Glavieux, "Near optimum error correcting coding and decoding: Turbo-codes," *IEEE Transactions on communications*, vol. 44, no. 10, pp. 1261–1271, 1996.

[38] "Beschreibung der Kodierung des Rufzeichens in einer MAC-Adresse," db0fhn.efi.fh-nuernberg.de/doku.php?id=projects:wlan:proposal, 2011, [Accessed: 15.05.2020].

[39] "Amateur Radio Numeric Callsign Encoding," github.com/darconeous/ham-addr/blob/master/n6drc-arnce.md, 2019, [Accessed: 15.05.2020].

[40] Analog Devices, "AD9361, AD9364 and AD9363 overview," wiki.analog.com/resources/eval/user-guides/ad-fmcomms2-ebz/ad9361, 2019, [Accessed: 15.05.2020].

[41] ITU, "Rec. ITU-R SM.329-12: Unwanted emissions in the spurious domain," International Telecommunication Union, 2014.