AN LTE SIGNAL ANALYSER

Masterarbeit
der Philosophisch-naturwissenschaftlichen Fakultät
der Universität Bern

vorgelegt von

Marcel Stolz
2015

Leiter der Arbeit:
Professor Dr. Torsten Braun
Institut für Informatik

Contents

List of Figures

List of Tables

Acknowledgements

First of all, I would like to thank Prof. Dr. Torsten Braun, head of the *Communication and Distributed Systems* group at the Institute of Computer Science, University of Bern, for giving me the opportunity to realise the *LTE Signal Analyser* project and write this thesis. His interest in my work and encouragement were a great source of motivation.
Furthermore, I would like to express my gratitude to Mr. Zan Li, who supervised my work. He invested immense resources in providing most useful advice and gave me an inspirational counsel every once in a while. He also obtained various hardware and software to support a neat and frictionless development. Last but not least, he spent a great amount of time to proofread and comment on my work, suggest improvements and bring up additional research ideas.
Moreover, I want to give my thanks to Mr. Heinz Wyss, janitor of the institute building and its surroundings, who allowed and helped in setting up measurements in restricted areas. I also want to thank Mrs. Daniela Schroth, secretary of our research group, as she provided tremendously rapid support in bureaucratic and other matters at all times.
Additional thanks go to my friends Arian Uruqi, Gianluca Greco, Christian Rippstein, Vojtěch Vasko, Patrick Wyler, Nina and Joscha Frey, Vera and Lukas Böhler, Tobias and David Weibel, Stéphane Vuilleumier, Bruno Rüedi, Florian Weibel and all people working in Czech cinematography, who provided relief and distraction in periods of hard and long nightly work.
Finally, I want to thank my parents and my sister—Michael, Zuzana and Bernadette Stolz—for their endurance, patience and daily support. The endorsement I felt over all stages of my work—the easy and more demanding ones—was the one only a family can give.

It is my wish to dedicate this thesis to the memory of my grandfather, Jaromír Jan Hladký, who passed away during its preparatory work.

Marcel Stolz, July 2015

Abstract

LTE, the mobile cell phone technology of the fourth generation, has recently been deployed by operators in the majority of western countries. An increasing number of devices using this technology is also encountered in Switzerland. Furthermore, research and development all over the world is turning towards *LTE*. However, only a small number of projects analysing the reception of traffic from *passive* stakeholders has yet been implemented. Additionally, powerful *Software Defined Radio* devices have become available at affordable costs, enabling the implementation of radio devices by means of software.

Therefore, we propose a software based *LTE Signal Analyser*, which attempts to draw a first overview of possibilities in signal analysis by means of *Software Defined Radio*. For this purpose, two scenarios were defined that motivate a specific use-case of analysed data, directed at localisation and device distinction. With this focus, the *LTE* specifications have been browsed for messages providing suitable data. Special interest was hereby laid on the extraction of identifier data. As a preliminary for an implementation, a workflow had been defined on the basis of the findings from the study of the *LTE* specifications. This workflow was used to implement a software defined capturing solution for downlink control messages.

Moreover, a specific environment of hardware and software components has been defined and implemented for real-world measurements. The defined workflow was run and the measured data have been analysed and visualised, focussing on the received identifiers transmitted in downlink control signalling.

As a further step the measured and analysed data have been evaluated and discussed. Adjustments to the measurement environments have been proposed and the structure of the visualisations interpreted as a result of the operators' load balance decisions. Also, the further steps necessary in order to achieve localisation have been outlined.

This work contributes to the field of both *Software Defined Radio* applications and the understanding of *LTE* transmissions and operator's choices. As results, we provide a *Software Defined Radio* implementation aimed at real-time, passive signal capturing. The findings that have been achieved from this implementation provide insights into the load in the cells measured and inspire further measurement scenarios and environments. They encourage and propose a variety of future work.

Chapter 1

Introduction

Mobile cell phone technology has become a fundament of western life-style during the past decades [13]. While it emerged into everyday life during the first ten years of the current century, it was pushed towards becoming an omnipresent resource for quotidian gadgets during the second ten years.
At the end of the past century, the mobile cell phone technology's second generation (e.g., *GSM*) was already widely spread. It promoted the factor of mobile voice communication, bringing the classical landline capabilities to all areas with cell phone reception. However, with the increasing importance of the Internet, the additional transmission of non-voice data became a new focus of technological development. The third generation of mobile technologies met this requirement by introducing standards such as *UMTS* at the beginning of the 21st century. These new technologies allowed packet switching for data transmissions instead of the voice centred circuit switching mechanism. 3G technologies considerably changed the use of mobile phones towards small Internet terminals. However, this also triggered an increase in cell traffic [13] and a new technology had to be developed in order to meet the high demands: The fourth generation of mobile technology was defined, introducing standards such as *WiMAX* [14] or *LTE* and *LTE-Advanced* [15, 16]. As already mentioned, these technologies are being deployed during the current decade (2010—2019). They are at the focus of modern device development and communication technology research. One of the reasons why 4G technologies are especially interesting is the continually increasing amount of non-phone devices, which make use of a 4G network directly or indirectly (via a smart phone's connection). Furthermore, the current trends for 5G technology development support this finding [17], as 5G is mainly aimed at supporting the *Internet of Things* (IoT) [18].
Hence, the analysis of 4G standards, such as *LTE* and *LTE-Advanced*, is an important factor when aiming to exploit the capabilities—but also know the drawbacks—of a technology that progressively forms our quotidian life. Moreover, the evolution of radio technologies has made it easier and affordable to do research in the field of wireless technologies. Devices such as the *Universal Software Radio Peripheral* (*USRP*) [19] allow software-driven development and adjustment of radio signal processing for both reception and transmission.
These two factors—the deployment of an "omnipresent" 4G technology and the possibility of programmable universal radio devices—are the basis for this thesis. In the upcoming sections a broader discussion of the motivation and scope of this thesis is given. First, Section 1.1 gives

an overview of the motivation for research in 4G technologies, their possibilities and potential use cases. Thereafter, Section 1.2 introduces the problems that are faced. Finally, Section 1.3 outlines the contents of the following chapters.

1.1 Motivation

As mentioned earlier, 4G technologies are at the focus of current hardware design and deployment. In this section, a broader motivation for study and development in a specific 4G technology is given: First, Subsection 1.1.1 provides a brief overview of this technology, called *LTE*, and its enhanced version, called *LTE-Advanced*. Subsequently, in Subsection 1.1.2 the current state of deployment concerning *LTE* networks in Switzerland is covered. Finally, Subsections 1.1.3 and 1.1.4 introduce two possible usage scenarios for technologies exploiting the analysis of LTE signals, to which this thesis could be a contribution.

1.1.1 The 4th Generation: *LTE* and *LTE-Advanced*

When looking into the field of mobile telecommunication standards of the fourth generation, two standards have been implemented in the past years: *LTE* (including *LTE-Advanced*) and *WiMAX*. As the aim of this thesis is to analyse 4G signal transmission, both of these technologies would need to be analysed. However, recent deployments of 4G cell systems have been made mainly with *LTE* [20]. Even though *WiMAX* (standardised by IEEE 802.16 [21]) was the first of the two standards being used—at that time in a *pre*-4G implementation—the advantages of *LTE* have induced the distribution of the latter [22]. Furthermore, the evolution of the core network in *LTE*, called the *Evolved Packet Core* (*EPC*), enables providers to move the background network to an *LTE* standard while maintaining, for example, *WiMAX* for radio access [23]. This leads to a further distribution of *LTE* technologies. Hence, this thesis focuses on *LTE*.
The *Long Term Evolution*—abbreviated as *LTE*—is a mobile communications technology standardised by the *3GPP* (*3rd Generation Partnership Project*). Its first major release was version 8, which the term *LTE* commonly refers to. However, version 8 does not fully meet the specifications of the fourth generation of mobile technology. Nevertheless, *LTE*, i.e. version 8, is often marketed as "4G" technology.
The actual implementation of the 4G specifications was met with *LTE* version 10, which is often called *LTE-Advanced*. Even so, it is important to consider that what is called *LTE-Advanced* is actually just a further development of *LTE* [5]:

> *Later releases of* LTE *are sometimes known as* LTE-Advanced*, but it is important to point out that* LTE *and* LTE-Advanced *are the same technology. The label "Advanced" was primarily added to highlight the relationship between* LTE *release 10 (*LTE-Advanced*) and* ITU/IMT-Advanced [...]. *This does not make LTE-Advanced a different system than LTE* [...]. *Another important aspect is that the developmental work on LTE and LTE-Advanced is performed as a continuing task within 3GPP* [...].

In particular, *LTE-Advanced* is backward compatible with *LTE*, i.e. findings that are made for *LTE* version 8 can also be applied to *LTE* version 10 (*LTE-Advanced*). Moreover, the innovations

in *LTE-Advanced* mainly include *Carrier Aggregation*, enhancements in multi-antenna (*MIMO*) techniques and *Relay Nodes* [24, 25]. These are advanced technologies that require specialised hardware. Additionally, even though *LTE* version 10 defines these advanced standards, it does not enforce them and allows the same technologies as with *LTE* version 8. Therefore, this thesis refers to *LTE* version 8 in the majority of the cases.
As already introduced, *LTE* (including *LTE-Advanced*) meets an increased interest in the last years: First, this new technology reaches a large number of devices, provides a complete transformation to a packet-switched environment, meets the demands of larger numbers of devices, increased data rates, higher capacities and mobility support. The detailed requirements of LTE can be looked up in Section 1.4 (page 11 ff.) of [23].
Second, *LTE* has been established as the dominating standard for 4G mobile telecommunications. We have seen that even though other technologies, such as *WiMAX*, have been introduced by cell phone providers, a transfer to *LTE* is taking place even in areas, where *WiMAX* has already been deployed.
Third, *LTE* is continuously developed and enhanced, as the introduction of *LTE* version 10 and its successors have shown. This promotes *LTE* as a long-term standard and it can be assumed that developments for *LTE* are going to be usable over longer time, making *LTE* especially interesting for research.
Hence, the decision to make an analysis of mobile telecommunication signals of the fourth generation by means of the *LTE* standard is an obvious consequence.

1.1.2 *LTE* in Switzerland

It has been stated, why *LTE* is of a general interest in communication technology research. Yet, a cell phone telecommunication standard has to be deployed by a mobile operator, in order to be used and enable an analysis of ongoing signalling. The only alternative to using a network deployed by a commercial operator would be to deploy a cell phone research network. However, the deployment of such a network would already be far beyond the scope of this thesis. Therefore, the deployment situation of *LTE* in Switzerland is of importance for doing 4G research.
The three major cell phone operators in Switzerland are *Swisscom*, *Sunrise* and *Salt* (formerly *Orange*). Meanwhile, all of these providers have deployed *LTE* in their networks. Traditionally, *Swisscom* is the telecommunication company that uses the most advanced technology and pushes its development and deployment. At the end of the year 2012 it launched its *LTE* network [26]. Since that time it continued enhancing its network and—along with *Sunrise* and *Salt*—deployed *LTE-Advanced* in certain areas [27]. Currently, *Sunrise* claims that over 50% of its customers can use *LTE*, while *Salt* states that 92% of the Swiss population lives in areas covered by its *LTE* network and *Swisscom* asserts a coverage of 97% [28, 2, 3]. Corresponding overview maps are shown in Figures 1.1–1.3. Even though this data may not be directly comparable, it shows that an analysis on the *Swisscom* network could be the most productive, as it uses the more advanced technology and provides a higher network coverage.
Nevertheless, *LTE* signal analysis is possible on all three networks and the technical quality and deployment status of *LTE* and *LTE-Advanced* seem to be on an exceptionally high level in Switzerland. This fact makes the analysis of LTE signals particularly interesting within Switzerland, be it for academic or economical purposes.

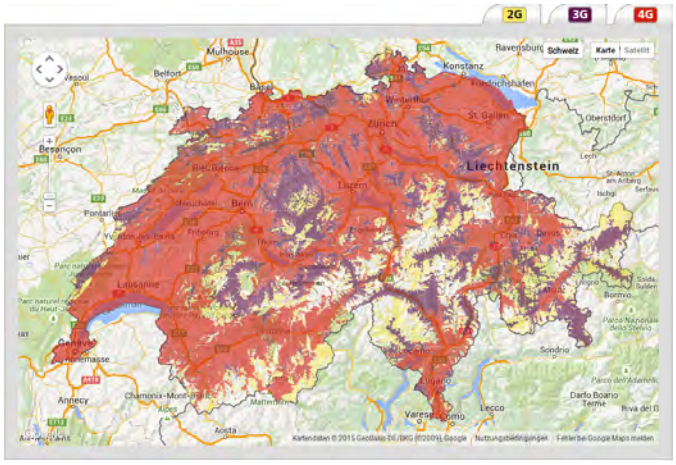

Figure 1.1: Coverage of the *Sunrise* network [1].

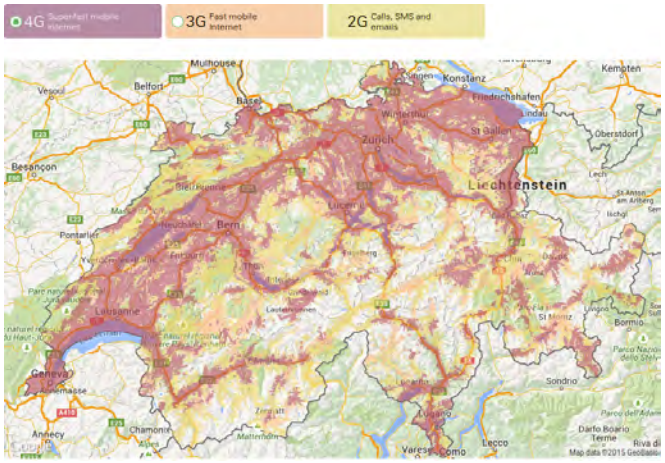

Figure 1.2: Coverage of the *Salt* 4G and 2G network (3G is neglected, as it cannot be displayed simultaneously with 4G and mostly overlaps with the 4G coverage) [2].

Figure 1.3: Coverage of the *Swisscom LTE* network [3].

1.1.3 A Long-term Development Goal: The *Shopping Mall Application Scenario*

In the previous subsections, the *research* interest for *LTE* in general and for Switzerland specifically was introduced. Yet, there are scenarios in other domains that further reinforce the motivation for *LTE* signal analysis. We introduce the *Shopping Mall Application Scenario*, which we take as a long-term goal for mobile telecommunications signal analysis.
It is a common practice nowadays that vendors create statistics on their customers' buying behaviour. This data is used, for example, to increase the sales volume by promoting further products the customers may be interested in [29]. One system that people probably encounter often in their everyday life is that of bonus cards or bonus programmes: The products sold in a shopping mall are mapped to a specific customer, which makes the promotion of other products, depending on the previous purchases, possible.
However, this system has certain limitations. The shop owner can determine a set of bought products. Yet, in a shopping mall, it may be relevant to keep track of the order in which a customer chooses to buy products. This may allow enhancements of the product locations inside a shop. For example, a large number of customers may prefer certain combinations of products. They are, therefore, located near to each other inside a shop. However, the customers end up walking past a product several times without perceiving it, or they intuitively want to buy the products in a specific order, which results in moving forth and back several times inside the shop. This can be annoying during crowded times for both the shop owner and the customer, especially in large superstores. Moreover, a marketing responsible may want to promote a new product to people. He finds out that this product is especially attractive for people that buy a set of other products. In this case, he may want to track whether the positioning of a promotion reaches its target audience when moving around the shop. Furthermore, he may want to direct the path of customers past specific locations with lucrative products.
These scenarios would need the possibility of tracking people as they move through the shop. It is less relevant, whether a specific customer actually buys a product. Rather, it is of interest how customers in general move through the shop—from which product type to which other type etc. This means that anonymised tracking data would be sufficient. Such movement tracking, however, presumes that the customers wear a device, which could be tracked. Obviously, most customers carry their smart phones with them, so these devices could be used.
Accordingly, we define the *Shopping Mall Application Scenario*, where a shop owner wants to track his customers' movements by means of mobile communication devices. He does, however, not have the possibility to work together with the operator due to data protection laws or financial considerations. Rather, he tries to extract information from messages that are sent between the user's smart phone and the operator's base station. A possible application of such a scenario is shown in Figure 1.4: The graphic on the left hand side shows a situation, where tracking information was not available, whereas the right hand side graphic depicts the situation after exploiting customer tracking. In particular, the latter shows the implementation of improvements based on customer movement analysis: Some customers may rather be interested in specific lunch goods. Their products are placed in one shelf, as these customers tend to visit the shop during "rush hours". The other goods have been repositioned so that the customer also passes the shelves with discounts or special offers.

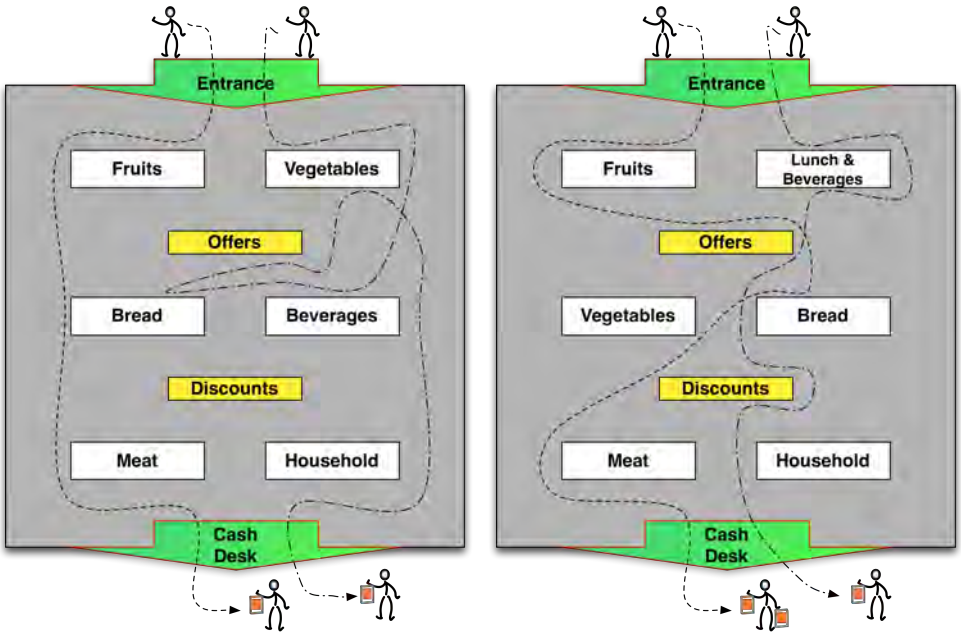

Figure 1.4: The *Shopping Mall Application Scenario*: Customers moving through a shop.

Certainly, the *Shopping Mall Application Scenario* is not the only possible use case for the analysis of radio telecommunication data. But it nicely illustrates, what could be an economical motivation for the analysis of *LTE* signals.

1.1.4 The *Event Management and Security Scenario*

We have seen the *Shopping Mall Application Scenario* in the previous subsection as a long-term goal for *LTE* signal analysis. Its ultimate motivation is to achieve localisation capabilities by means of signal capturing in *LTE*. An important prerequisite for localisation is device *distinction*. It is needed in order to know how many devices are available for localisation and by means of what information these devices can be identified. As device distinction forms an intermediate goal that should also yield a possible real-world application, we introduce the *Event Management and Security Scenario*.

This scenario is motivated by the following considerations: A variety of events, such as open air concerts, festivals or fairs trigger massive gatherings of visitors. The large amount of people does not only influence the management of the event area. It may also have an impact on surrounding areas, as visitors move towards or away from an event area. In both cases, it is a concern of the event management and security to know the approximate number of people within a specific area. Large numbers of people could cause security risks. An illustration of such a situation is shown in Figure 1.5.

As already stated in the previous subsection, most people nowadays carry a cell phone with them. However, the usage of cell information from the operator does not form an adequate instrument

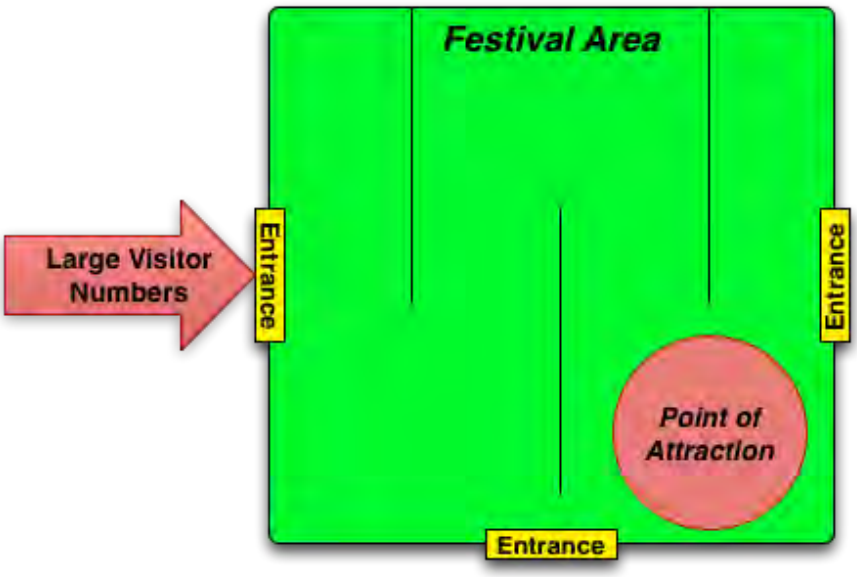

Figure 1.5: The *Event Management and Security Scenario*: Points of massive gatherings of people.

due to data protection laws or financial considerations. Hence, the number of people could be determined by analysing cell phone traffic. This presumes the distinction of cell phones, which enables the calculation of the number of people using a cell phone in a specific area.

1.2 Problem Formulation

In the previous section, the interest for research in the domain of *LTE* networks has been described. First, Subsection 1.1.1 introduced the evolution and dominance of *LTE* as a 4G standard. Thereafter, Subsection 1.1.2 reported how *LTE* network deployment is promoted in Switzerland and showed that a general interest and possibility for research is present in this country. Finally, Subsections 1.1.3 and 1.1.4 proposed two scenarios as possible use cases.
As the primary motivation for doing an analysis of *LTE* signalling is laid out, the challenges in doing so should be found and highlighted. This is done in the following subsection.

1.2.1 The Analysis of *LTE* Signals

We have found a clear motivation for analysing *LTE* signals. This task can be split up into the following phases:

1. The theoretical definition of procedures where an external stakeholder can extract any kind of signalling from the user's device or the operator's base station.

2. The design of an implementation architecture in which signals can be received and decoded.

3. The application of the implementation architecture in a specific location and environment in order to do a *proof of concept* and analysis.

4. The extraction of insights by means of an evaluation and the definition of a resulting conclusion with recommendations for future work (i.e. future iterations).

During these phases, several difficulties may be met. The first step includes a very broad examination of the lower *LTE* layers, such as the physical layer, which define the means of radio communication. This includes a vast variety of technologies that have their origin in the domain of physics and electrical engineering, rather than in computer science.
The difficulty of the second phase is that the findings from step one need to be implemented by means of devices compatible with both the high technical requirements of *LTE* and an interface that can be regulated by means of software. Moreover, this includes finding an appropriate operator network and working within it. However, network operators primarily do not publish technical information concerning their network set-ups.
Finally, in the third and fourth step, the data found needs to be analysed in an appropriate way and the correct conclusions need to be drawn, as far as possible. It is especially important to also highlight the probability of wrong interpretations or findings from the results and to determine, how likely the findings are to represent the reality, as they cannot be compared to operator data.

1.3 Outline

The prior sections outlined the motivation for analysing *LTE* signals and the corresponding challenges. In Section 1.1 a motivation for research in the domain of *LTE* was given. Furthermore, the specific situation of *LTE* deployment in Switzerland was outlined and two scenarios defined as a trace for the economical use of *LTE* signal analysis. Subsequently, Section 1.2 defined the challenges of this thesis by outlining the steps necessary for implementing signal analysis. The aim of the current section, eventually, is to draw on outline of the following chapters an their contents.
As several attempts on *LTE* signal analysis have already been made and also partial software-driven implementations of *LTE* exist, these are summarised in Chapter 2. Furthermore, this chapter contains a short overview of possible devices that allow software-driven development in radio signal transmission and reception.
After this overview of related work, Chapter 3 outlines the basic concepts in *LTE* with a focus on the physical layer and radio signal transmission.
Thereafter, theoretical concepts for exploiting transmissions of radio signals for signal analysis are defined in Chapter 4. This includes the application of the theoretical fundaments found in the foregoing chapter to specific transmissions of interest for signal analysis as well as general procedures needed in order to decode transmitted information.
Subsequently, Chapter 5 introduces the environment set-up for doing measurements and, eventually, signal analysis. The measurements made in the defined environments are then presented and discussed in Chapter 6.
Chapter 7, finally, draws a conclusion of the findings from the other chapters. Moreover, proposals for further iterations of our implementation for signal analysis are made.

Chapter 2

Related Work

A variety of projects are available in the domain of *LTE* research, with a majority focussing on the *LTE* downlink. A selection of these projects has been found to be helpful for both the detection and analysis of *LTE* signals. The projects are introduced in the upcoming sections. However, in order to do research and development for *LTE* in the domain of computer science, an important preliminary is the availability of software-driven platforms for radio transmission. They build the foundation on top of which further software can be developed and are introduced first: Section 2.1 introduces some *Software Defined Radio* devices and Section 2.2 a popular development framework for *Software Defined Radio* called *GNURadio*.
In a second step, a selection of the most relevant projects concerning *LTE* that intersect with signal analysis aspects are presented:
First, the *Open Air Interface* (*OAI*) [30] project of *EURECOM* represents a very broad implementation of the *LTE* release 8.6 [16] and a subset of LTE release 10 (also known as *LTE Advanced*) [16]. Its advantage is its very extensive implementation. However, this is also its drawback: the structure of its implementation is complex and searching for specific implementation parts is rather cumbersome. Furthermore, it does not provide a satisfactory documentation of the source code structure. The *Open Air Interface* is presented in Section 2.3.
Second, the *LTEye* project [31] provides a proposal for a system allowing *LTE* signal propagation analytics. However, the code provided by the *LTEye* project [32] is only able to do offline processing in a *MATLAB* [33] environment. The *LTEye* project is described broader in Section 2.4.
Finally, the *OpenLTE* project [34] provides a code base for various *Software Defined Radio* (*SDR*) devices. It is able to search for base stations or even emulate parts of a base station (*eNobeB*). The *OpenLTE* project is introduced in Section 2.5.

2.1 *Software Defined Radio* Devices

Software Defined Radio—commonly abbreviated as *SDR*—is a term embracing technologies, which provide programmable radio interfaces for virtually ubiquitous purposes in both reception and transmission. The big advantage of *SDR* is its flexibility. Traditionally, radio hardware was designed for a specific purpose and technology. The functions were “hard wired”, i.e. could not be changed after designing the hardware. *SDR* brings a new flexibility to radio hardware, as it

is no longer necessary to design all functions in advance. This means that it has also become simpler to do research or analysis in radio reception, transmission and design. The resulting benefits for purposes such as *LTE* signal analysis are immense. For example, it is possible to programme an *FM* broadcast radio transmitter or receiver on *SDR* hardware.
A series of devices are available on the market enabling *SDR* in various qualities and for various purposes: *rtlSDR* [35], *OsmoSDR* [36], *HackRF* [37] or *USRP* [38] are only some examples of available *SDR* devices. These devices differ in their technical specifications, e.g. the reception and transmission quality, the processable bandwidth or the available sampling rates. Higher technical abilities usually cause higher costs but are necessary in order to fulfil the high demands of technologies such as *LTE*.
The *USRP* (*Universal Software Radio Peripheral*) is a series of *low-cost, high-quality* Software Defined Radio *(*SDR*) systems* [38] developed by *Ettus Research* [38], aimed but not limited to academic and research purposes. Because of their high quality and wide application range, several *USRP* device types are interesting for developing *LTE* applications with *SDR*.
USRP devices and *SDR* hardware in general are usually accessed by a common programming framework. It is called *GNURadio* and is briefly introduced in the next section.

2.2 The *GNURadio* Framework

The *GNURadio* project is a framework that aims to provide a programming environment enhancing the development of applications for *SDR* devices. This is done on the one hand by providing easier and consistent interfaces for the communication with such devices and, on the other hand, by providing a variety of predefined code blocks, which are often used for signal processing. Furthermore, *GNURadio* provides a convenient way to adapt or add individual code blocks and interconnect them in a straight-forward manner.
The *GNURadio* project is open source and supports various *SDR* hardware, including the devices listed in Section 2.1. It generally supports *C++*, including several variants of the *C* programming language. This is especially important when processing performance critical code of resource-demanding implementation. Additionally, *Python* or *C++* can be used to programmatically combine signal processing blocks into a flowgraph. *GNURadio* also provides the possibility of creating flowgraphs and combining signal processing blocks by means of a graphical user interface called *GNURadio Companion* (*GRC*).
As *GNURadio* supports a wide range of devices, supports performant applications, is open source and can easily be extended, it is popular for research and development in the domain of *SDR* applications and provides a good basis for the implementation of *LTE* based applications.

2.3 *OpenAir 4G*

The *Open Air Interface* [30] is a software and hardware platform, which aims to support research in wireless mobile communication protocols of the fourth and fifth generation. As it is the aim of this thesis to work with the 4G technology *LTE*, *OpenAir 4G* is of interest. Apart from hardware

components, which provide an interface similar to general *Software Defined Radio* components, *OpenAir 4G* primarily includes software components in all layers of the radio interface of *LTE*. This includes simulators for parts of the radio interface. Furthermore, *OpenAir 4G* claims to provide a complete implementation of the *Evolved Packet Core* (*EPC*), which is the back-end *IP* based network that includes components such as the *Mobile Management Entity* (see Chapter 3). Most of the information concerning *OpenAir 4G* is found on the *twiki*, which describes the installation from source [39]. Even though *OpenAir 4G* claims it is open source [30, 39, 40], some parts of the code are not publicly accessible and, therefore, parts of the software are not included in the public release. This explicitly includes the *EPC* implementation and does not violate the source licence, since it is possible to compile the other parts without the *EPC* code and to run the software. However, it demonstrates one of the problems when working with *OpenAir 4G*: Despite the source code being generally open source, many parts of information or documentation are only accessible for people that belong to *EURECOM* or benefit from some kind of affiliation. This is also one of the major drawbacks when using *OpenAir 4G*. Then again, one of the advantages of *OpenAir 4G* is that it is said to run not only on *EURECOM*'s devices, but also on the widely spread *USRP* device family [40].
The structure of the public code is as follows [39]: The implementation files of the different *LTE* layers are grouped into dedicated folders. The low level physical layers as well as radio transmission simulators are grouped into the directory labelled *openair1*. Additionally, the *RLC*, *MAC*, *PDCP* and *RRC* layers are condensed in the *openair2* folder. Eventually, higher layer protocol implementations are found in *openair3*, whereas parts of the core network are found in *openair-cn*.
Although this coarse structure is quite explicit, the code arrangement within this structure is neither very intuitive nor well commented. Besides, general documentation material on the *OpenAir 4G* code is very scarce and does not encompass more than rather general presentation material (see point *Additional Documentation* in [39]). For all of these reasons, *OpenAir 4G* has not been thoroughly used during the development for *LTE* signal analysis. Rather, its code has been used as a reference, where the corresponding parts could be identified and used for any part of the implementation.

2.4 The *LTEye* Project

The *LTEye* project aims to be the *first open platform to monitor and analyse* LTE *radio performance at a fine temporal and spatial granularity* [31]. It claims that it is even able to localise mobile users within a certain area. However, the project's code that was released on one of its developer's website [32] only allows offline analysis of recorded data. The provided code is written for *MATLAB* [33] and cannot directly process data from a radio device. Furthermore, it has been found that the code did not work for recorded data from Switzerland. The analysis of the implementation showed that it was an implementation for a very specific environment and could not be used in Switzerland. An adaptation of the code to work in Switzerland would have brought rather high overhead for the mere result of an offline analysis. Hence, an adjustment of the code to the Swiss environment was not made.
Nevertheless, the paper that describes the goal of *LTEye* [31] has been found to be of interest.

Apart from describing the general aims of the project, it gives a summary of possible approaches for the passive extraction of information in the down- and uplink of *LTE*. Notably, the project proposes to use temporary network identifiers (*RNTI*s) in the channels of *LTE*. Even though these findings appear very interesting, it is not clear whether all of them are actually possible to implement. However, the findings of the paper have served as an inspiration for the possibilities of *LTE* signal analysis.
Eventually, it has to be stated that even though the project code was accessible over some time on the developer's website, it is currently no longer available on any resource on the Internet.

2.5 The *OpenLTE* Project

The *OpenLTE* project is an *open source implementation of the 3GPP LTE specifications* [34]. It provides a partial implementation of *LTE*, including *E-UTRAN* capabilities needed for cell scanning and basic *eNodeB* functions (see Chapter 3). This makes *OpenLTE* a suitable code base for further *SDR* based implementations of *LTE*. Furthermore, it allows both simulations in *GNU Octave* [41] as well as signal processing by means of compatible *SDR* devices (see also Section 2.1). The project is completely open source and written voluntarily by developer Ben Wojtowicz. Nonetheless, its code structure is intuitive and abound in comments. Some minor documentation is also found in the project's wiki [34].
Due to these findings, the *OpenLTE* project is a very convenient project that can be used as a code base: By providing readable and well structured code and a partial implementation of the *LTE* standard, an ideal starting position for further development and analysis is given.
The most important parts of the code are structured in the following directories, each containing a folder for the header and corresponding implementation files:

- ***liblte*** contains a common code base for all parts of the project. It includes partial implementations of the different layers (e.g. *PHY*, *MAC*, *RLC*, *PDCP*), which are split up into individual files. Moreover, the implemented functions include exact references to the *LTE* specification documents. These two facts are the major reasons for the good readability of the *OpenLTE* code.

- ***LTE_fdd_dl_file_gen*** provides the source code for generating a file containing a downlink signal. As all the other "front ends", it relies on the *GNURadio* framework and the files from *liblte*. This applies to all of the following directories.

- ***LTE_fdd_dl_file_scan*** includes the code and flowgraph for scanning a recorded file for *LTE* signals. This means that the compiled programme can take any recorded file with downlink signals as input and detect these signals including the contained information that is broadcast by an *LTE* base station.

- ***LTE_fdd_dl_scan*** contains the same functionality as *LTE_fdd_dl_file_scan* without the limit of only scanning files. More precisely, it is a front end that requires an *SDR* device, by means of which it scans the received signals in real-time.

- ***LTE_fdd_enodeb*** is a partial implementation of an *LTE* base station (*eNodeB*). Therefore, it transmits control information to cell phones. Devices containing a previously registered *SIM* card can connect to this base station.
- ***LTE_fdd_file_recorder*** provides a front-end for recording signals that are received by an *SDR* device to a file.

As mentioned, the *OpenLTE* project is based on the *GNURadio* framework and provides support for multiple *SDR* devices. These facts and the possibility for real-time signal processing make *OpenLTE* a primary choice for further developments in the *LTE* domain.

Chapter 3

LTE Concepts and Procedures

In the previous chapters, the motivation for analysing *LTE* signals was outlined and related projects were introduced. In order to analyse *LTE* signals, it is inevitable to study the technical specifications of this technology before considering any implementation. Hence, the goal of the current chapter is to provide a succinct and understandable summary of *LTE* concepts and the corresponding technical specifications. Focus is put on contents necessary for the implementation of an *LTE* signal analyser.
The upcoming sections introduce the relevant technical specifications step by step. All of the specifications for *LTE* are published as *Specification Series 36* by the *3rd Generation Partnership Project* (*3GPP*) and are accessible online [42]. However, the provided resources are, on the one hand, written in a very marginal style and, on the other hand, the information is distributed over a large number of different files (approximately 200). This makes the provided specifications difficult to read and understand. Therefore, the aim of the following sections is to provide an *easy-to-understand*, yet as profound as necessary documentation of the relevant technological aspects. For this purpose, the contents of this chapter have been written based on both the mentioned technical specifications of the *3GPP* as well as selected literature describing the *LTE* standard more specifically [23, 5, 43, 4]. It should be pointed out that especially *Ghosh et al.* [4] and *Dahlmann et al.* [5] have been found very helpful. The contents of the upcoming sections strongly relies on the mentioned sources.
The content of this chapter is organised as follows: An overview of *LTE* network components and their purpose is first given in Section 3.1. Also, general terms and techniques often used in *LTE* are described. Subsequently, more detailed explanations on the communication between a mobile device and an operator's radio base station are given: Section 3.2 describes the communication structure, layers and channels of the so-called *E-UTRAN* part of *LTE*. Thereafter, Section 3.3 outlines the specific physical technologies used for radio access. The details of uplink transmission are not covered in this chapter, as they do not form a major part of this thesis.

3.1 An Overview of the *LTE* System Architecture

Before describing the technical details of *LTE* technologies, it is reasonable to provide a broad overview of the concepts and components of *LTE*. The current section focusses on yielding such

an overview, based on [43, 23, 5].
Mobile telecommunication cell networks usually consist of two prominently visible components: A terminal device, which accesses the network by means of radio signals, and a radio base station enabling access to an operator's network. In *LTE*, the terminal device is called *User Equipment* while the base station's name is *Evolved Node B*. The abbreviations *UE* and *eNodeB* are commonly used, respectively. Examples for *UEs* include (but are not limited to) smart phones, *LTE* capable tablets or laptops. The communication between a *UE* and an *eNodeB* is transmitted as a radio signal (i.e., wirelessly) and includes two distinct directions called *uplink* and *downlink*. The term downlink defines communication from the *eNodeB* to the *UE* and the term uplink expresses communication in the opposite direction. An illustration of downlink and uplink communication between a *UE* and an *eNodeB* is depicted in Figures 3.1 and 3.2, respectively.
An *LTE* network, however, consists of additional components, which are less obvious. They are grouped into different parts of the network, depending on their tasks. The functions that are implemented in an *eNodeB* are referred to as *Evolved Universal Terrestrial Radio Access Network* (*E-UTRAN*). These functions comprise both user- and control-plane protocols necessary for the communication between *eNodeB* and *UE*, where the terms user- and control-plane refer to the type of data that is being handled: The first describes data the user actually wants to receive and transmit, while the second stands for communication needed in order to handle user data transmissions. The *E-UTRAN* does explicitly not include the *UE*, as the radio communication is fully controlled by the *eNodeB*. Direct communication between *eNodeB*s is possible, for example to exchange data concerning a handover procedure from one *eNodeB* to another. It is defined by the so-called *X2* interface.
A further group of *LTE* network components is called *Evolved Packet Core* (*EPC*) and is sometimes also referred to as *Software Architecture Evolution* (*SAE)*. It consists of components ensuring data transmissions between the *eNodeB* and outside networks, such as the Internet or *VoIP* networks. The individual components forming the *EPC* are presented in the following.
First, the *Mobile Management Entity* (*MME*) is responsible for managing and controlling most communication aspects with a *UE*. It is directly connected to the *eNodeB*, assigns radio resources to a *UE*, is responsible for the establishment and release of radio connections and handles a device's mobility. Furthermore, the *MME* is responsible for ensuring both device authentication and security management. It allocates temporary identities to a *UE*.
Second, every *eNodeB* is also connected to the *Serving Gateway* (*S-GW*), which functions as a central point of user-plane data traffic communication for the *UE*. Notably, the *UE* communicates with the same *S-GW* independently of the serving *eNodeB*. Hence, it is often described as the *UE*'s "mobility anchor".
Both of these components interact with the *eNodeB* directly via an interface called *S1*. It is further subdivided into the *S1-MME* and the *S1-U* interface, depending on whether it it is used for the communication with an *MME* or an *S-GW*, respectively.
The *Packet Data Network Gateway* (*P-GW*) forms the third component of the *EPC*. It is responsible for the communication between *LTE* components and an external packet based network. This usually includes the Internet as well as *IP Multimedia Subsystem* networks, such as *VoIP* networks. Consequently, the *P-GW* is also responsible for assigning an *IP* address to the *UE*. As

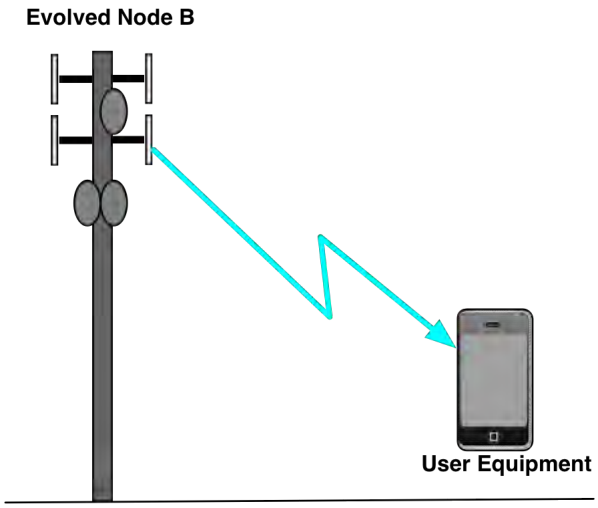

Figure 3.1: An illustration of downlink communication.

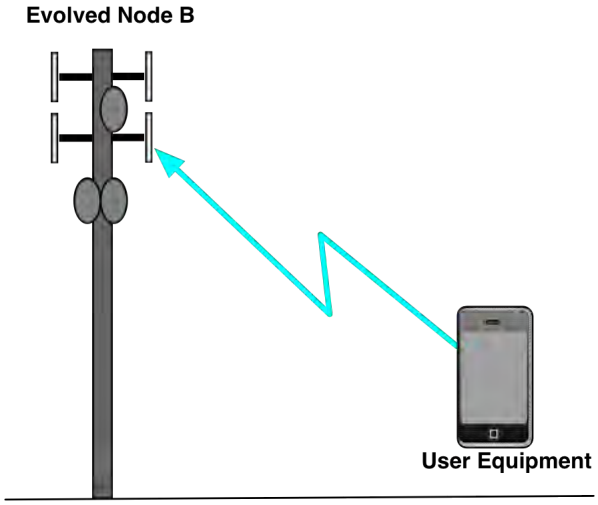

Figure 3.2: An illustration of uplink communication.

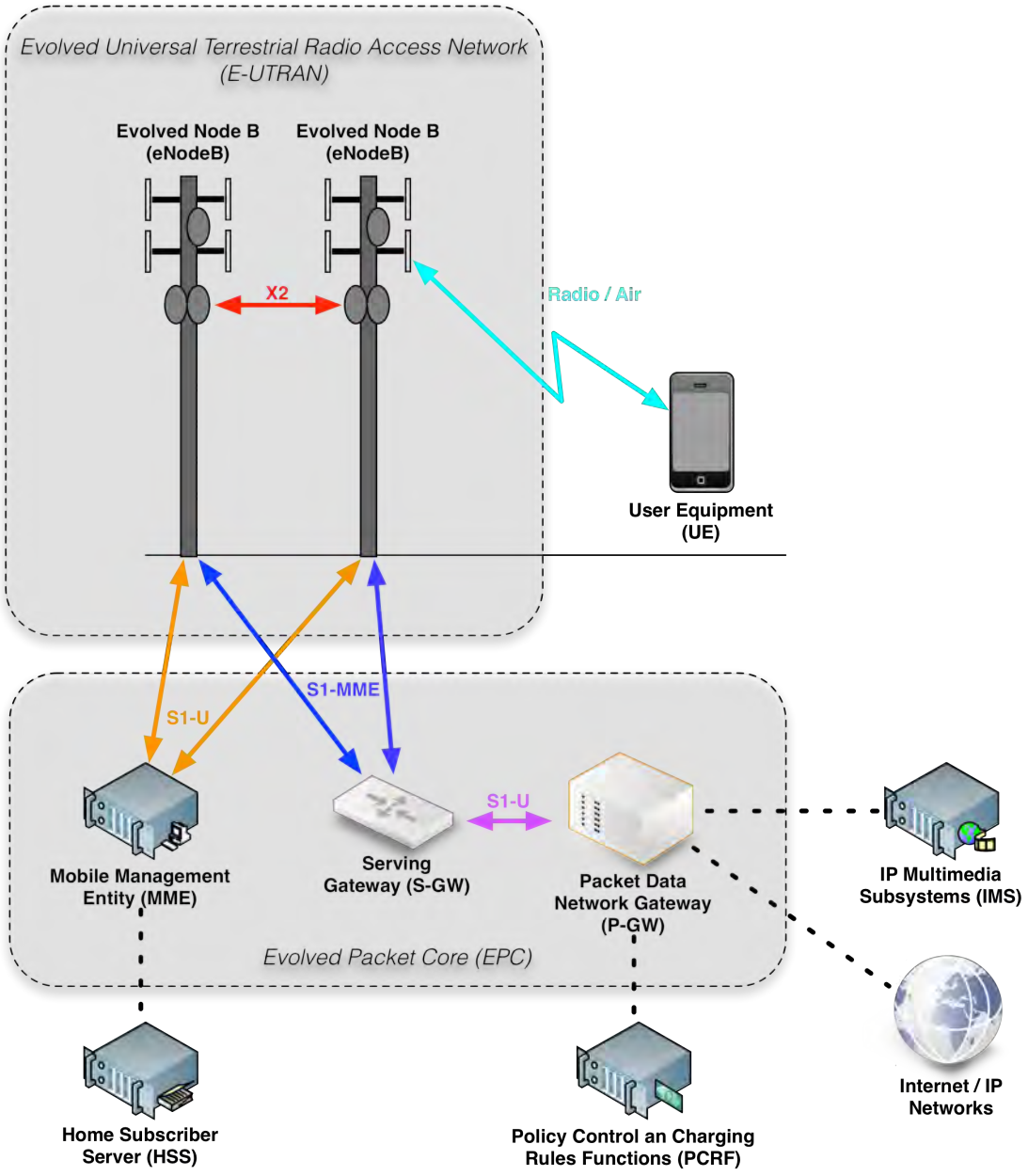

Figure 3.3: Components of an *LTE* network. Structure based on [4].

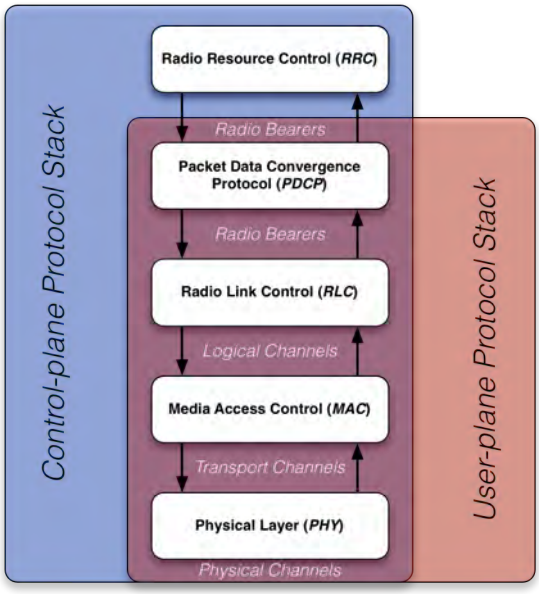

Figure 3.4: The user- and control-plane protocol stack of the *E-UTRAN*, based on [5]

the entire *LTE* network architecture makes use of a packet switched topology for both data and voice services, the *P-GW*'s address allocation is a crucial component of *LTE*.
Additionally, two other components are often apportioned to the *EPC*: The *Home Subscriber Server* (*HSS*) and the *Policy Control and Charging Rules Functions* (*PCRF*). The former is connected to the *MME* and stores user specific information, while the latter supports the *P-GW* in ensuring *UE* specific policies and charging, such as *QoS* agreements or volume- and time-based charging.
The introduced components are displayed in Figure 3.3. They represent the *LTE* network architecture, commonly referred to as *LTE* or sometimes called the *Evolved Packet System* (*EPS*). As our aim is to implement signal analysis capabilities, the *E-UTRAN* technologies form the most relevant part of the *LTE* network architecture. The upcoming sections focus on their explanation and the treatment of related radio communication aspects.

3.2 Structure of the *E-UTRAN*

Previously, a general overview of the *LTE* network structure was given. Specifically, it was laid out that an *LTE* network consists of two major subnetworks: On the one hand, the *EPC* forms the background network of *LTE*, interconnecting *eNodeB*s and providing a gateway to external networks, such as the Internet. On the other hand, the *E-UTRAN* is responsible for the communication between the *UE* and an *eNodeB*. As the latter includes radio transmissions enabling passive decoding, it is of special interest. The following subsections describe the general structure of the *E-UTRAN* needed for *LTE* signal analysis, based on [4] and [5].
First, Subsection 3.2.1 introduces the hierarchy and responsibilities of the different network lay-

ers defined for *E-UTRAN* communication. Subsequently, the most important communication channels of the bottom layers of the protocol stack are introduced in Subsection 3.2.2. They function as a medium for dedicated data streams. Finally, network identifiers, which form an important means for *E-UTRAN* radio transmissions, are described in Subsection 3.2.3.

3.2.1 The *E-UTRAN* Protocol Stack

As discussed, the *LTE* network relies on IP communication. When an IP packet is transmitted by means of the radio medium, it is first processed by the *Packet Data Convergence Protocol* (*PDCP*). It is further processed by the *Radio-Link Control* (*RLC*), the *Media Access Control* (*MAC*) and the *PHY* layer, as shown in Figure 3.4. In the following, the responsibilities of each layer are summarised from the transmitter's point of view, i.e. in a top-down approach. On the receiver's side, this process is performed correspondingly in the opposite order.
The *PDCP* layer processes an *IP* packet and compresses its header in order to reduce the packet's transmission size. It is also the task of this layer to perform security functions (e.g., ciphering and integrity protection) and to forward the data units to the *RLC* layer. The *PDCP* handles data input and output on the basis of radio bearers, which are assigned to a user, i.e. each separate radio bearer is handled by one *PDCP* entity.
The aim of the *RLC* layer is to segment (and reassemble) data packets. Moreover, it handles error correction by means of *Automatic Repeat Request* (*ARQ*, see [44]) and transmits the protocol data units by means of the corresponding logical channels (see Subsection 3.2.2). Similarly to *PDCP*, *RLC* handles data from the overlying layer in a per radio bearer way.
In the *MAC* layer, the data from the preceding layer (i.e. logical channel data) is mapped to the corresponding transport channels for the *PHY* layer. Furthermore, *MAC* is responsible for *Hybrid Automatic Repeat Request* (*HARQ*, see [45]) retransmissions and, in the *eNodeB*, for both up- and downlink scheduling.
The responsibilities of the *PHY* layer consist of *Cyclic Redundancy Check* (*CRC*) insertion, channel coding. scrambling, modulation and transmission over the radio medium. In particular, *PHY* maps data units from the transport channels to corresponding physical channels. An additional task of this layer is the application of multiple antenna techniques for transmission, i.e. the implementation of *Multiple Input Multiple Output* (*MIMO*). As we do not need these techniques for our implementation, their discussion is omitted.
In Figure 3.4, the *E-UTRAN* protocol stack is divided into a user- and control-plane part. The procedures in the layers that are common to both planes are in most instances the same or similar and their differences are not discussed here. However, the control-plane protocol stack contains an additional layer called *Radio Resource Control* (*RRC*). It is responsible for the broadcasting of system information, connection management, *UE* mobility (e.g., cell re-selection), the measurement of radio signal quality and the retrieval of the *LTE* hardware capabilities from the *UE*. Furthermore, it handles the connection state between a *UE* and an *eNodeB*. Two connection states are defined for a *UE* called *RRC_CONNECTED* and *RRC_IDLE*. When a *UE* is in the *RRC_CONNECTED* state, both the *UE* and the *eNodeB* can communicate to each other. In particular, the *UE* has assigned a temporary network identity (see also Subsection 3.2.3). In the *RRC_IDLE* state, this is not the case. Furthermore, the *UE* is not hosted by a specific cell and switches off its radio interface at most times, in order to save battery power. Only during specific

Table 3.1: Downlink logical channels (white), transport channels (grey) and physical channels (light cyan).

Acronym	Name
PCCH	Paging Control Channel
BCCH	Broadcast Control Channel
CCCH	Common Control Channel
DTCH	Dedicated Traffic Channel
DCCH	Dedicated Control Channel
MCCH	Multicast Control Channel
MTCH	Multicast Traffic Channel
PBCH	Physical Broadcast Channel
PDSCH	Physical Downlink Shared Channel
PMCH	Physical Multicast Channel

Acronym	Name
PCH	Paging Channel
BCH	Broadcast Channel
DL-SCH	Downlink Shared Channel
MCH	Multicast Channel
PHICH	Physical Hybrid-ARQ Indicator Channel
PDCCH	Physical Downlink Control Channel
PCFICH	Physical Control Format Indicator Channel

short periods the *UE* "wakes up" in order to listen for paging messages. These may be triggered, for example, by incoming phone calls or text messages. An additional aspect of the *RRC_IDLE* state is that the *UE* has no uplink resources assigned. Hence, it can only access the network by means of unscheduled *random access procedures*.

3.2.2 Communication Channels

The previous subsection presented the responsibilities of the protocol layers in the *E-UTRAN*. It has been stated, that these layers communicate either over the structures of radio bearers or, at the lower layers, by means of explicitly defined communication channels. First, the *logical* channels are responsible for the communication between the *RLC* and the *MAC* layer. Second, the *MAC* and *PHY* layer communicate by means of *transport* channels. The physical layer, finally, transmits data over the *physical* channels. In the following, the responsibilities of these *logical*, *transport* and *physical* channels are discussed.
An overview of all available communication channels is shown in Figure 3.5 for the uplink and in Figure 3.6 for the downlink. Subsequently, both channel types are discussed in a top-down approach, beginning with the downlink channels. The corresponding channel names and acronyms are displayed in Tables 3.1 (downlink) and 3.2 (uplink).
In the previous subsection, a difference between user-plane and control-plane protocols was made (see also Figure 3.4). Similarly, the channels that carry information between layers can generally be divided between user data and control data channels. Of the seven logical channels available in the downlink, only the *DTCH* and *MTCH* are dedicated to user data. The remaining channels carry different kinds of control data. Among the logical channels, two are not designed for regular traffic but for the control and transmission of *Multimedia Broadcast Multicast Service* data, e.g. mobile *TV* transmissions. These are the *MTCH* and *MCCH*, which are not needed during the implementation and neglected in this discussion.

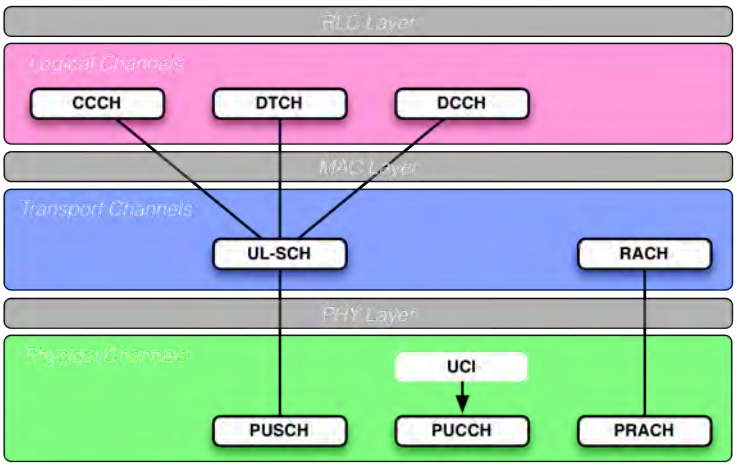

Figure 3.5: Overview of the uplink logical, transport and physical channels, adapted from [5].

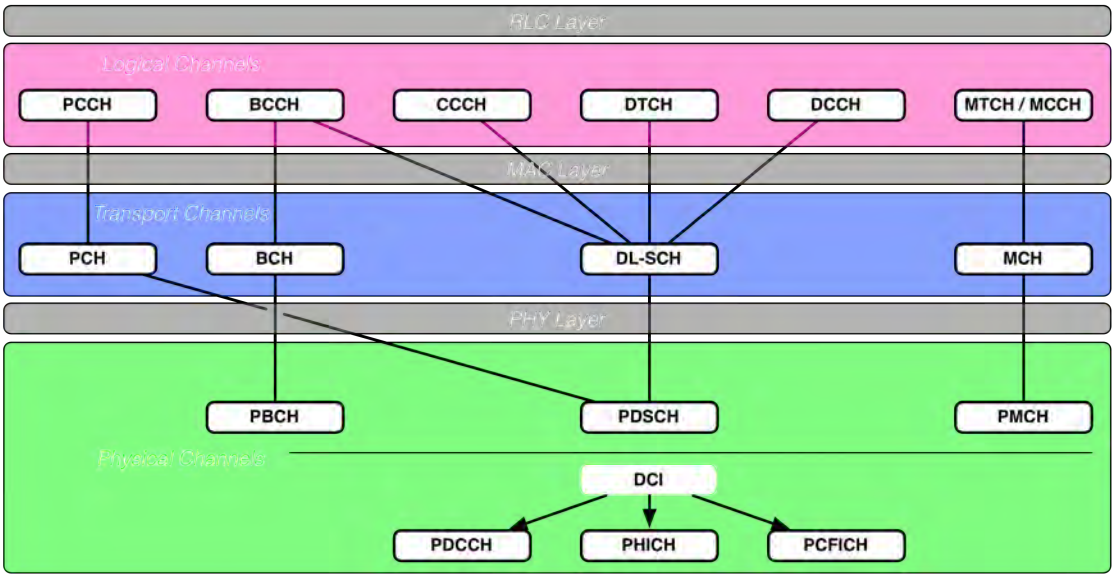

Figure 3.6: Overview of the downlink logical, transport and physical channels, adapted from [5].

The *PCCH* is directed to paging attempts from an *eNodeB* to a *UE* in the *RRC* layer, as discussed at the end of the preceding subsection. The responsibility of the *BCCH* is to broadcast system information that is relevant for all devices hosted by an *eNodeB*. This includes information needed in order to initiate a connection to an *LTE* radio network. The *CCCH* is dedicated to control information directed at specific *UE*s as a response to so-called *random access requests*. They are triggered by a *UE* in the uplink when it is not assigned any or enough uplink radio resources. The final two channels, *DTCH* and *DCCH*, are devoted to tasks concerning a specific *UE*. While *DTCH* transmits the actual user data to a *UE*, *DCCH* transmits *UE* specific control information.

As discussed, it is the responsibility of the *MAC* layer to map logical channels to transport channels. This mapping is done as shown in Figure 3.6. Most of the transport channels have similar tasks as the overlying logical channels from which they carry data: The *PCH* is responsible for paging messages and adds mechanisms that allow idle *UE*s to receive a paging message over several predefined wake-up intervals. In the *BCH*, parts of the *BCCH* are transmitted to all *UE*s: The so-called *Master Information Block* (*MIB*) is transmitted over this channel, carrying important information for *UE*s that newly connect to an *eNodeB*. The responsibility of the *DL-SCH* is mainly the transmission of *UE* dedicated downlink data. It adds classical *MAC* capabilities, such as *HARQ*. Furthermore, it supports discontinuous reception by the *UE*s (in order to save battery power) and transmits the contents of the *BCCH* not transmitted by the *BCH*, such as *System Information Blocks* (*SIB*s). We refrain from discussing the *MCH* transport channel, as we do not implement any multicasting techniques.

The transport channels are mapped to physical channels by the *PHY* layer. They correspond to specific resources in the time-frequency grid (see also Section 3.3.3). The *PBCH* carries the data provided by *BCH* and is located in the time-frequency grid such that all *UE*s are capable of receiving it, without knowing anything about the cell configuration. General downlink data from *DL-SCH*, consisting of traffic dedicated to a specific *UE*, and paging messages are carried by the *PDSCH*. Again, *PMCH* is not discussed, as it is only used for multicast services. Apart from physical channels that are mapped from overlying transport channels, a series of physical channels exist, which are directly supplied by the *PHY* layer. They provide *Downlink Control Information* (*DCI*) containing details necessary for receiving and decoding downlink data. First, the *PDCCH* provides control information used for the downlink. In particular, scheduling information for decoding the *PDSCH* is transmitted. As the *PDCCH* can have different formats, the *PCFICH* is needed, which is always transmitted in the same format and location, in order to decode the *PDCCH*. Finally, the *PHICH* carries information on *PHY* layer *HARQ* processing.

All channels of the downlink have been summarised and we continue with the presentation of the uplink channels. The names for the uplink channel acronyms are shown in Table 3.2. As can be seen in Figure 3.5, the uplink channel structure is similar to that of the downlink.

In the case of the logical channels, the names all correspond to names used for downlink logical channels. As the tasks of these uplink channels are also very similar to those of the downlink, we refrain from discussing them again.

Among the the uplink transport channels, the *UL-SCH* caries data of similar type as the *DL-SCH*: It contains uplink user and control data from the overlying layers that is transmitted from the *UE* to the *eNodeB*. The *RACH*, on the other hand, implements a function uniquely available

Table 3.2: Uplink logical channels (white), transport channels (grey) and physical channels (light cyan).

Acronym	Name
CCCH	Common Control Channel
DTCH	Dedicated Traffic Channel
DCCH	Dedicated Control Channel
UL-SCH	Uplink Shared Channel
RACH	Random Access Channel
PUSCH	Physical Uplink Shared Channel

Acronym	Name
PUCCH	Physical Uplink Control Channel
PRACH	Physical Random Access Channel
PRACH	Physical Random Access Channel

Table 3.3: *Radio Network Temporary Identifier* types and ranges.

Acronym	Value (decimal)	Value (Hex)	Name
RA-RNTI	1—60	0001—003C	Random Access RNTI
C-RNTI	61—65'523	003D—FFF3	Cell RNTI
M-RNTI	65'533	FFFD	Multicast RNTI
P-RNTI	65'534	FFFE	Paging RNTI
SI-RNTI	65'535	FFFF	System Information RNTI

in the uplink: It provides the possibility for *UE*s to communicate with the *eNodeB* even if they do not have any or enough uplink resources assigned. This can be either the case when more regular uplink resources are needed or when initial access to a cell is requested.
The uplink physical channels are, once more, similarly organised as in the downlink. First, the *PUSCH* is an uplink alteration of the *PDSCH*. Second, the *PUCCH* is directly supplied with *PHY* information, similarly to the *PDCCH*. It transmits *Uplink Control Information* (*UCI*), containing information on the radio transmission quality on the *UE* side. Furthermore, it also transmits *PHY* layer *HARQ* responses to the *eNodeB*.

3.2.3 *E-UTRAN* Network Identifiers

The most important parts of *LTE* and the communication channels of the radio access component *E-UTRAN* have been discussed. However, the topic of network identifiers has not yet been treated. It is evident from the preceding subsections that, as *LTE* is a packet switched network and provides a gateway to the Internet, *LTE* relies on *IP*. Thus, every *UE* uses *IP* addresses in order to communicate with the Internet. However, an *IP* layer is not present in the *E-UTRAN* protocol stack and, hence, these addresses are not used for direct transmissions between a *UE* and an *eNodeB*. Rather, they are encapsulated in the user data that is transmitted to and from the *EPC* / *eNodeB*. Nevertheless, an identifier distinguishing downlink and uplink communication packets to and from specific *UE*s is inevitable.
Obviously, a fixed user address, such as the *IMSI* should only be used ciphered for radio transmissions due to security and privacy risks. Hence, a different kind of identifier is needed

for transmissions in the *E-UTRAN*. *LTE* uses so-called *Radio Network Temporary Identifiers* (*RNTI*s) for this purpose. As their name states, they are assigned on a temporary basis.
The *LTE* specifications define different types of *RNTI*s, depending on what kind of content is transmitted [46]. In Table 3.3, the most important *RNTI* types and their acronyms are listed. The probably most significant type is the *Cell RNTI* (*C-RNTI*). It is assigned uniquely to one *UE* per cell. Hence, it is the device identifier used for *E-UTRAN* communication. As discussed later on in Chapter 4, *RNTI*s are used to mark packets depending on the type of information they carry. In particular, packets dedicated to a specific *UE* are marked by the device-specific *C-RNTI* in the up- and downlink. As the *C-RNTI* is temporary, it can also be changed over time, even if the *UE* stays within the same cell. The specifications of *LTE* do not give a concrete time interval or algorithm, on the basis of which the *C-RNTI* is reassigned. However, the *LTEye* project claims the following in one of its released papers [31]:

> *A user continues to have the same C-RNTI as long as she is in the same cell and is not idle for more than the pre-configured tail timer value. The timer value is typically a few seconds to tens of seconds (12 sec in the measurement result* [...]*). Hence, the C-RNTI assigned to a user may change quite often if it transmits sporadically.*

As stated, other *RNTI* types are used to mark certain types of transmissions. For example, the *Paging RNTI* (*P-RNTI*) identifies messages that are used for paging. Similarly, a *UE* listening for radio transmissions in the downlink can identify system information that is being transmitted in the *PD-SCH* by means of the *System Information RNTI* (*SI-RNTI*). The *Multicast RNTI* (*M-RNTI*) is used to identify multicast messages, which are not treated in this thesis.

3.3 *LTE* Radio Transmissions

The general structure of the *E-UTRAN* has been described in Section 3.2. However, the specifications needed for the actual physical transmission of radio signals between an *eNodeB* and a *UE* has not yet been described. This mostly includes procedures used in the *PHY* layer discussed in Subsection 3.2.1, including the corresponding physical channels.
In the following, the general specifications used for radio transmissions in the *E-UTRAN* are first described in Subsection 3.3.1. Thereafter, modulation schemes, which transform a number of bits into a radio signal and are used in *LTE* are described in Subsection 3.3.2. Finally, an entity of the *LTE* time-frequency grid called *radio frame* is summarised in Subsection 3.3.3.
The technical content of the following subsections is based on [4] and [5].

3.3.1 *LTE* Radio Bands, Bandwidths and Duplexing

We have seen the components of an *LTE* network. Before introducing the mechanisms of *LTE* radio signal transmissions in the next subsections, it is necessary to provide information on the environment in which signalling is possible. For this purpose, three aspects shall be treated in this subsection: The available radio bands for *LTE*, possible bandwidths and the options for duplex communication.

Table 3.4: Frequency spectrum defined by the *Federal Office of Communication* (*BAKOM*) for *LTE* in Switzerland [10].

BAKOM Frequency band	**Bandwidth**	**LTE band numbers**	**Duplex mode**
800 MHz	2x30 MHz	20	FDD
900 MHz	2x35 MHz	8	FDD
1800 MHz	2x75 MHz	3, 9	FDD
2100 MHz	1x20 MHz 2x60 MHz 1x15 MHz	1, 33, 34	TDD FDD TDD
2600 MHz	2x70 MHz 1x50 MHz	7, 38	FDD TDD
3400—3800 MHz	400 MHz	42, 43	FDD and TDD

LTE Radio Bands

The radio bands defined for *LTE* communication span from approximately 0.7 GHz to 3.8 GHz and lie in the licensed frequency spectrum. The bands are defined by the specifications of the *3GPP* [42] and can be looked up, for example, in [4]. The frequency bands are either paired or unpaired, depending on the duplexing mode used (see Paragraph *Duplexing Mode*). The band numbering currently ranges from 1 through 25 and 33 through 43. In Switzerland, the *Federal Office of Communication* administers the allocation of the licensed frequency spectrum [10]. It states that the available frequencies lie in the ranges as shown in Table 3.4. If we compare this to the definition of *LTE* frequency bands, we see that several of these bands are available in Switzerland. However, currently only bands 3, 7 and 20 are being used by Swiss operators.

Available Bandwidths

In the previous paragraph, the frequency bands used for *LTE* in general and for Swiss *LTE* networks in particular have been presented. However, depending on the allocation of the frequency spectrum to the operators, the physical environment and other parameters, different bandwidths are available for the transmission. These bandwidths can be chosen by the operator for a specific *eNodeB* and do generally not change. The available bandwidths are 1.4 MHz, 3 MHz, 5 MHz, 10 MHz, 15MHz and 20 MHz. However, depending on the bandwidth used only specific bandwidths are possible. For band 3 all of the mentioned bandwidths can be used by an operator. Bands 7 and 20 only allow the usage of 5 MHz, 10 MHz, 15MHz and 20MHz. Any of these defined bandwidths can be chosen for any *eNodeB*, i.e. the operator is not bound to consequently use the same bandwidth for all *eNodeB*s deployed in an *LTE* frequency band.
In the downlink the whole available bandwidth is divided into a corresponding number of subcarriers. In the uplink, on the other hand, the bandwidth utilised by a *UE* is usually smaller due to the single-carrier property of the uplink signal generation (see Subsection 3.3.2).
The choice of the bandwidth in the downlink has a direct influence on the number of resources

that are available for transmission. The details of these relations are presented in Subsection 3.3.3.

Duplexing Modes

We have already seen that different frequency bands are available in *LTE*. As stated, the choice of a band depends, among others, on the choice of the *duplexing mode*. *LTE* communication in the radio domain should allow both up- and downlink communication, as this is needed for voice communication. The simultaneous transmission of up- and downlink data is called *full duplex*, whereas its successive transmission is called *half-duplex*.
LTE uses two duplexing modes: One is called *Time Division Duplex* (*TDD*), the other is called *Frequency Division Duplex* (*FDD*). As its name implies, *TDD* accomplishes duplex transmission by assigning the same frequency spectrum to either the up- or downlink during specific time intervals. As simultaneous transmissions of up- and downlink data is obviously impossible in *TDD*, it represents a half-duplex technique. On the other hand, *FDD* provides both half- and full duplex capabilities. It defines a frequency spectrum dedicated to uplink transmissions and another frequency spectrum for downlink transmissions. The choice of half- or full duplex *FDD* depends on the capabilities of the *UE*, as not all devices support simultaneous reception and transmission of radio signals. Half-duplex is, if applicable, only supported by the *UE*, while the *eNodeB* always uses full duplex, as it can schedule transmissions for other *UE*s in the downlink while one *UE* accomplishes transmissions on the uplink. Regardless of using half- or full duplex, the user experiences services such as voice communication as "full duplex". This results from very short switching times between up- and downlink communication when using half-duplex mode.
Even though we have seen previously that both *FDD* and *TDD* bands can be licensed in Switzerland, only *FDD* is deployed by the operators. The reason may be that two of the three bands capable of *TDD* (42 and 43) have been introduced in *LTE* as of version 10. The operators, therefore, had already licensed other frequency spectra which imply *FDD*. In general, it can be said that *FDD* seems to be the more spread duplexing mode in other countries as well. As *LTE* in Switzerland currently only uses *FDD*, we refrain from discussing *TDD* specific varieties of the *LTE* standard in the following subsections.

3.3.2 Modulation and Multiplexing Schemes

E-UTRAN communication is divided into a four layer user-plane and five layer control-plane protocol stack, as discussed in Subsection 3.2.1 and illustrated in Figure 3.4. The current subsection describes parts of the bottommost *PHY* layer. It is responsible for processing and transmitting data it receives from overlying layers over the radio interface. One important aspect of the *PHY* layer is the transformation of digital information, i.e. bits, to analogue radio waves, in order to transmit data between the *UE* and the *eNodeB*.
The transmission of bits by means of radio waves is enabled by *modulation schemes*. The modulation schemes used in *LTE* are *QPSK*, *16QAM*, *64QAM* and, infrequently, *BPSK*. The former three are treated in detail by [5] in Chapter 2 and the latter can be found in Chapter 4 of [47]. We refrain from discussing the technical details of these modulation techniques. However, we

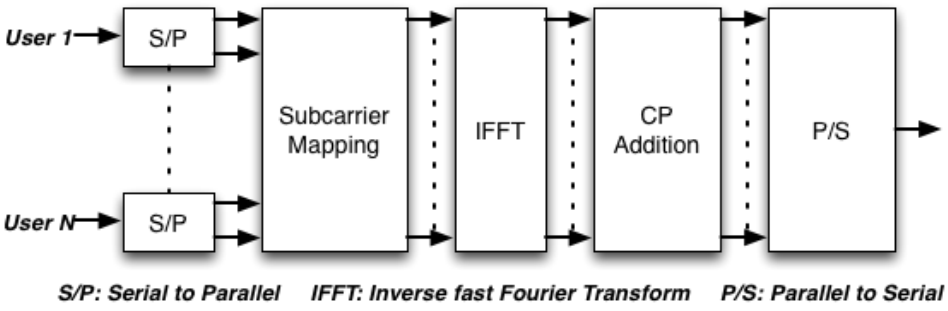

Figure 3.7: Block diagram for *OFDMA*, adapted from [4].

note that each of them transforms a specific number of bits (e.g. 1 bit for *BPSK* or 6 bits in the case of *64QAM*) into a *symbol* that can be transmitted as a radio signal. They form the means of communication for *LTE* radio transmission in the up- and downlink.
After a bit stream has been transformed into analogue information in the form of a radio signal at a specific carrier frequency, this signal can be generated and transmitted. However, *LTE* is a multiple access technology: Multiple *UE*s are served by one *eNodeB* and, hence, multiple signals are transmitted simultaneously. Allowing communication with several devices is a technique called *multiplexing*. It can be achieved by dividing the available resources among the devices. This can be done by means of time division, frequency division or increased complexity techniques. *LTE* chooses frequency multiplexing by means of *Orthogonal Frequency Division Multiple Access* (*OFDMA*) and *Single-carrier Frequency Division Multiple Access* (*SC-FDMA*) for the down- and uplink, respectively. The reason for two different multiple access techniques roots in the limited power resources of conventional *UE* devices. While *OFDMA* introduces a lot of enhancements for the downlink transmissions due to its orthogonality, *SC-FDMA* includes a subset of these enhancements while taking into account the limited power resources available for the signal generation in the uplink. The following two paragraphs provide a brief summary of the most important aspects of both technologies.

OFDMA

The *Orthogonal Frequency Division Multiple Access* technology enables communication to multiple *UE*s by dividing the available frequency bandwidth into orthogonal subcarriers. All users are assigned a (possibly different) number of subcarriers, which may be chosen due to their devices' transmission characteristics. This enables *LTE* to choose the optimal subcarriers for a user with the lowest probability for negative influences, such as multipath fading.
The *OFDM* modulator takes a series of arbitrary symbols from one of the previously presented modulation schemes as an input for a user. These symbols are first converted from a serial stream to a parallel stream, depending on the number of subcarriers assigned to the user. The parallel streams are then mapped to the corresponding subcarriers. Subsequently, the frequency-domain signal is transformed into time-domain information by means of an *Inverse Fast Fourier Transform*. Thereafter, a *Cyclic Prefix* (*CP*) is inserted before each *OFDMA* symbol. The reason for

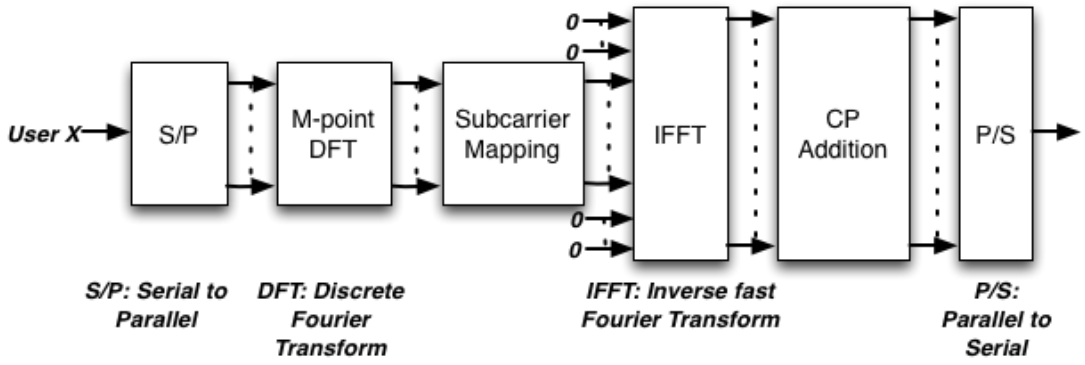

Figure 3.8: Block diagram for *SC-FDMA*, adapted from [4].

this *CP* insertion is a measure to prevent inter-symbol interference and is described in greater detail in Subsection 3.3.3. Finally, the parallel signals are transmitted by means of one radio signal from a wireless interface. An illustration of the *OFDMA* modulation process is shown in Figure 3.7. The demodulation of an *OFDMA* symbol is done by inverting the order of the modulation procedure.

SC-FDMA

Single-carrier Frequency Division Multiple Access is a technology similar to *OFDMA*. It accepts the same possible input symbols on the *UE*'s side as *OFDMA* does on the *eNodeB*'s side. However, while *OFDMA* uses all of the available system bandwidth, divided into equally sized subcarriers that may be arbitrarily assigned to a user, *SC-FDMA* only uses neighbouring subcarriers for a user and a user only generates signals for the bandwidth he actually utilises. This illustrates in an intuitive way, why *SC-FDMA* is less power consuming than *OFDMA*. Figure 3.8 depicts the steps necessary for modulating an *SC-FDMA* signal. First, the serial symbol stream is parallelised. Subsequently, a per-used-subcarrier *Discrete Fourier Transform* is used, in order to map the symbols to consecutive subcarriers. So as to eventually achieve the single-carrier property, the remaining unused subcarriers of the system bandwidth are filled with zeroes. Thereafter, as in *OFDMA*, a *CP* is added and the resulting signal is transmitted via a wireless interface. The demodulation of the signal is likewise performed in the opposite direction.

3.3.3 *LTE* Radio Frame Architecture

As already brought up in Subsection 3.3.2, radio transmission in *LTE* is achieved by means of two dimensions: On the one hand, the specific frequency bandwidth is used. It is divided into distinct subcarriers, each carrying a radio *symbol*, which represent certain combinations of bits. On the other hand, the information in the subcarriers changes over time, and so does the information transmitted by the subcarriers, i.e. the radio waves change in order to represent different combinations of bits. This results in a bit stream, which is decoded on the receiver's side or encoded for transmission on the transmitter's side. For defining the transmission characters

Table 3.5: Relation between the used bandwidth and further parameters for normal cyclic prefix [11, 12].

Channel Bandwidth [MHz]	1.4	3	5	10	15	20
Number of Resource Blocks in the Downlink	6	15	25	50	75	100
Number of Occupied Subcarriers	72	180	300	600	900	1200
Sample Rate [MHz]	1.92	3.84	7.68	15.36	23.04	30.72
Number of Samples per Slot	960	1920	3840	7680	11520	15360
FFT size	128	256	512	1024	1536	2048

in these two dimensions, we talk of the *time-frequency* grid.
In the frequency domain, the subcarriers form the smallest unit. The number of subcarriers depends on the chosen downlink bandwidth, as shown in Table 3.5. Based on the bandwidth, the receiving device also has to choose further parameters, which are displayed in Table 3.5 but not further discussed.
In the time domain, different units are defined for common use. First, the *radio frame* stands for a period of $10ms$, independently of the number of subcarriers used. It is further divided into 10 *subframes* of each $1ms$ length. A subframe simultaneously defines the *Transmission Time Interval* (*TTI*). The *TTI* forms the smallest unit of time, during which a *UE* can be assigned a resource. Finally, a subframe is further divided into two *slots* with a duration of $0.5ms$. This time domain structure is depicted in Figure 3.9.
Figure 3.9 represents an important structure, which combines the frequency and time domain. It is called *Physical Resource Block* and always consists of twelve subcarriers in the frequency domain. In the time domain, it corresponds to 1 slot, i.e. $0.5ms$, and carries either six or seven *OFDM* modulation symbols, depending on whether a normal or extended *Cyclic Prefix* (*CP*) is used. The *CP* is an element that is added to the beginning of each *OFDM* symbol in order to improve the reception. It consists of a copy of the last part of the same symbol and forms a so-called *guard-interval*, which helps to reduce the *inter-symbol interference* with the preceding symbol. The information of the *CP* can either be used to improve the received information or be neglected. It should be noted that the *CP* of the first symbol of a *PRB* is slightly longer ($\approx 5.1\mu s$) than the remaining *CP*s ($\approx 4.7\mu s$).
The smallest components of the time-frequency grid—the *OFDM* symbol including a *CP* on the one hand and a subcarrier on the other hand—form a *resource element*. Hence, a *PRB* is made up of 7 x 12 resource elements. We ignore the extended *CP*, which would result in a *PRB* structure of 6 x 12 resource elements, as it is not used in the cells we detected in Switzerland. Furthermore, the defined structure only counts for *FDD* and we refine from discussing the *TDD* structure.

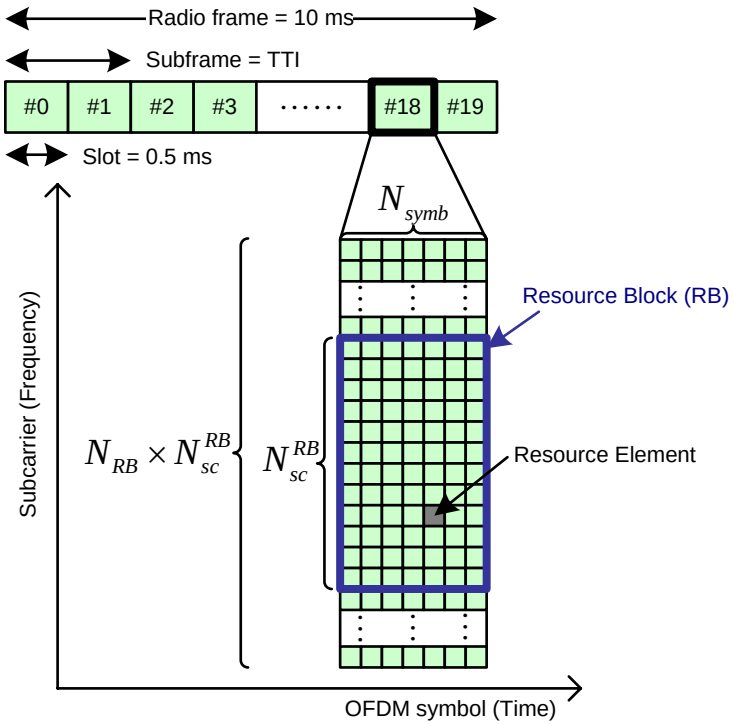

Figure 3.9: Overview of the time and frequency domain structures used in *LTE* downlink radio transmissions (graphic from [6]).

Chapter 4

Theoretical Downlink Decoding Workflow

The previous chapter introduced the basic specifications of *LTE* used in this thesis for signal analysis. This knowledge needs to be browsed in order to define adequate procedures enabling signal analysis by means of passive signal reception and decoding. The aim of the current chapter is to present the findings from this process, i.e. to describe the procedures used in the implementation described in Chapter 5.
In order to provide a possibility for signal analysis in *LTE*, a focus on specific signals was laid. The structure of *LTE* communication implies that the downlink needs to be decoded before any decoding of the uplink would be possible. Due to the high complexity of this process and the even higher complexity faced in uplink signal reception, a concentration on the downlink signal transmission was decided. The downlink provides the possibility to receive control messages and extract the identity of their receivers.
Even though further possibilities for analysis in the downlink may be found, the extraction of control signals and their identities form a very promising means of signal analysis. This is in particular the case when considering the motivation of the *Shopping Mall Application Scenario* and the *Event Management and Security Scenario*, which require the localisation—and prior to this, the distinction—of users moving within a specific area. As a basis for implementing these scenarios the acquisition of downlink decoding and subsequent identity extraction was defined as a goal for the implementation of this thesis. This process and its individual steps are shown in Figure 4.1. The depicted four steps are discussed in the following two sections as follows: The first two steps of *Cell Synchronisation* and *System Information Extraction* form a prerequisite for decoding any further data on the downlink. Therefore, they are discussed together in Section 4.1 and are referred to as *downlink capturing procedure*. The following two steps of Figure 4.1 subsequently decode the information included in the *PRB*s received by identifying specific downlink channels. As a consequence, this enables the extraction of the identifiers included in the control messages. As this is the main goal of the decoding of downlink information, we name the combination of these two steps *RNTI Extraction* and discuss it in Section 4.2. Finally, the found procedures and steps are put together in order to define a workflow for decoding downlink data and extracting identifiers. The definition and illustration of this workflow is done in Section 4.3.

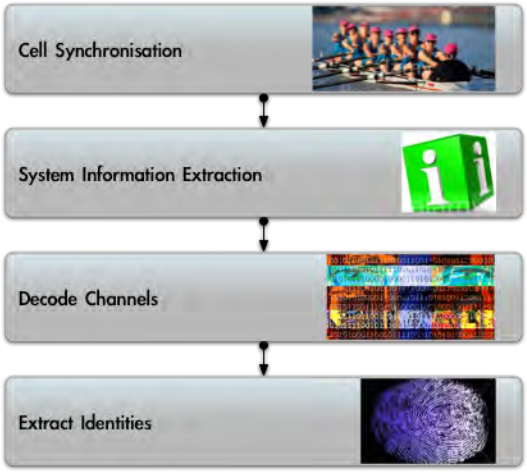

Figure 4.1: The procedure implemented by the *LTE* signal analyser for achieving device distinction.

4.1 Downlink Capturing Procedure

The aim of the current section is to define the prerequisites for extracting any user identifiers from control messages. Hence, it includes the steps that have to be achieved *before* any control messages can be decoded or identifiers extracted. An important process in order to decode downlink information is, first of all, the synchronisation of a *UE* to a cell. This enables the correct decoding of any further information transmitted by the *eNodeB*, as it gives the *UE* information on the notion of time the *eNodeB* has. The procedure for acquiring cell synchronisation is described in Subsection 4.1.1.
As soon as the decoding of downlink information is possible, the *UE* must extract information related to the cell's transmission characteristics. This includes, for example, information on the used bandwidth. The process needed for the acquisition of cell information is described in Subsection 4.1.2.
The technical details on which the procedures from both subsections are based have been adapted from [4] and [5].

4.1.1 Cell Synchronisation

In order to communicate with a cell, a *UE* first has to synchronise its notion of time with the *eNodeB*. This is necessary, as down- and uplink communication is largely achieved on the basis of scheduled signalling. Synchronisation to a cell is similarly needed, when an external device wants to receive and analyse the communication that is done within a cell.
Cell synchronisation is achieved by periodical transmissions of cell synchronisation signals. Information on the overall cell search and acquisition procedure as well as the reception of the two different synchronisation signals is provided in the following paragraphs.

Cell Search and Acquisition

A *UE* needs not only to search for cells when it initially connects to a network. It continuously has to perform cell search in order to keep track of neighbouring cells, so that it can switch to one of them in case it offers higher communication quality.
To identify a cell, a *UE* searches on specific centre subcarriers for synchronisation signals. Two types of such synchronisation signals are usually needed for cell synchronisation: The *Primary Synchronisation Signal* (*PSS*) and the *Secondary Synchronisation Signal* (*SSS*). They are utilised twofold: On the one hand, to achieve the same notion of signal transmission in the time domain and, on the other hand, to determine the physical identity of a cell. This cell identity N_{ID}^{cell} is defined as follows [5]:

$$N_{ID}^{cell} = 3N_{ID}^{(1)} + N_{ID}^{(2)}$$

The expression $N_{ID}^{(1)}$ stands for the *cell identity group* and consists of one of the values $(0, ..., 167)$, whereas the expression $N_{ID}^{(2)}$ represents the *cell identity within a group* and is located in the interval $(0, ..., 2)$. The cell identity group is included in the *SSS*, while the cell identity within a group is contained in both the *PSS* and the *SSS*. The calculation of the physical cell identity is needed, as several parameters for reception of signals transmitted in the cell rely on this value. For example, the *Cell-specific Reference Signal* depends on the physical cell identity and is needed in order to report the signal quality received from an *eNodeB*.
Once the *UE* has determined the physical cell identity, it can listen for cell specific information by decoding the information of the *PBCH*.

Primary Synchronisation Signal

We have seen the broader context of what the *PSS* is used for in the preceding paragraph. The details of the *PSS* generation and detection are laid out subsequently.
As presented in the previous paragraph, the *PSS* contains the *cell identity within a group* ($N_{ID}^{(2)}$), i.e. one of the three values $(0, 1, 2)$. More precisely, the *PSS* itself can have three distinct forms, depending on the $N_{ID}^{(2)}$ value it represents. Three *Zadoff-Chu* sequences (see Chapter 5 in [48]) are used to represent the *PSS*. These sequences have the ability that, when the initial signal is compared to the transmitted signal, the correlation function shows an extraordinary peak in exactly the place in time where the signal is presented. At all other times and for all other signals, the correlation function provides a significantly lower value that is relatively near to a zero value. This enables a precise detection of a *PSS* signal without the risk of any false positives and also means that distinct *PSS* signals from different cells do not interfere with each other. An illustration of a correlation function for *Zadoff-Chu* sequences in *LTE* is shown in Figure 4.2.
The *PSS* is always found in the same location within a radio frame: It is contained in *the last symbol of the first slot of subframes 0 and 5* [5]. An illustration of the *PSS*' location is shown in Figure 4.3. It is contained on the 73 centre subcarriers of the downlink. However, the centre subcarrier (called *DC* subcarrier) is not used for any transmissions, hence the signal is in fact only present on the 72 centre subcarriers. This size corresponds to six *Physical Resource Blocks* in the frequency domain. However, only the 62 subcarriers surrounding the *DC* subcarrier actually carry a signal, as the *Zadoff-Chu* sequences have a length of 62 *OFDM* symbols. The remaining

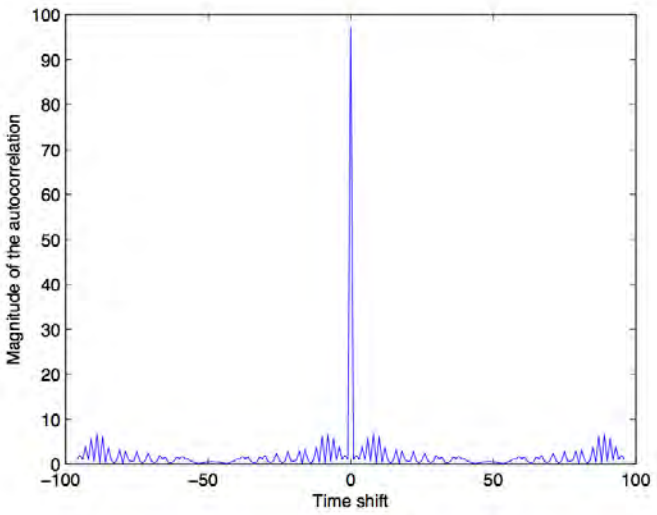

Figure 4.2: An illustration of the correlation function of a *Zadoff-Chu* sequence used in *LTE*.

ten subcarriers at the edges are filled with zeroes and, hence, do not transmit any values.
By means of the *PSS* the *UE* achieves a synchronisation in the time domain, that allows a distinction of signals up to the level of resource elements (i.e., individual *OFDM* symbols). However, the *UE* does not yet have an understanding of the notion of subframe numbers or frame dimensions, as the received *PSS* signals have two possible locations within a radio frame and are the same regardless of their containing subframe number.

Secondary Synchronisation Signal

We have seen that the *PSS* can be used to get a time synchronisation up to the level of an *OFDM* symbol. However, the received *PSS* signals have to be put into context within their radio frame, as two *PSS* signals per frame are transmitted. The *SSS* is responsible for the synchronisation of a *UE* on the level of radio frames. Again, as for the *PSS*, the *SSS* is included twice within a radio frame. It is located in the very same subframes (zero and five) and slots (first) as the *PSS*. Its location within the slot is the second last symbol. Hence, it is contained in the *OFDM* symbol preceding the *PSS*, as shown in Figure 4.3. In contrast to the *PSS*, however, the *SSS* has two different structures depending on its location within the radio frame. Similarly to the *PSS*, both types of the *SSS* occupy the 62 subcarriers surrounding the *DC* subcarrier. The signals transmitted in the *SSS* are made up of two 31 bit *m*-sequences, which differ depending on the 168 possible values of the cell identity group. The individual bits of these sequences are put in sequence alternatively. The two types of *SSS*, now, differ in the order of the mapping of the bits onto the subcarriers. In one case, the bit for a subcarrier is first taken from the first sequence, in the other case, it is first taken from the second sequence. An illustration of the mapping of the *m*-sequences to the corresponding subcarriers is shown in Figure 4.4.
As the *UE* can distinguish the location of the *SSS* within a radio frame, it is now also aware of

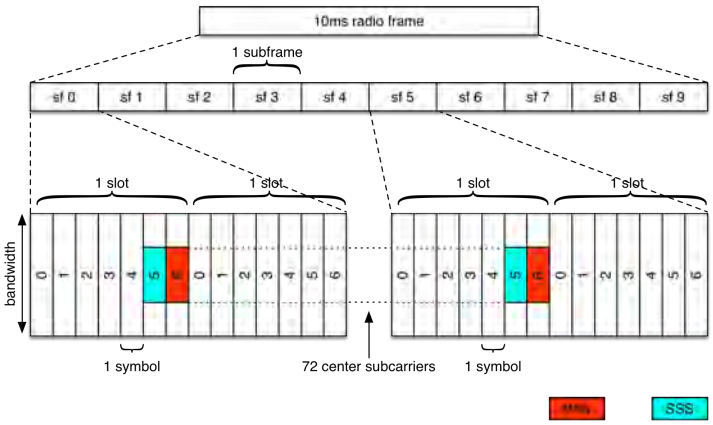

Figure 4.3: The location of the *Primary Synchronisation Signal* and *Secondary Synchronisation Signal* within a radio frame (graphic based on [7]).

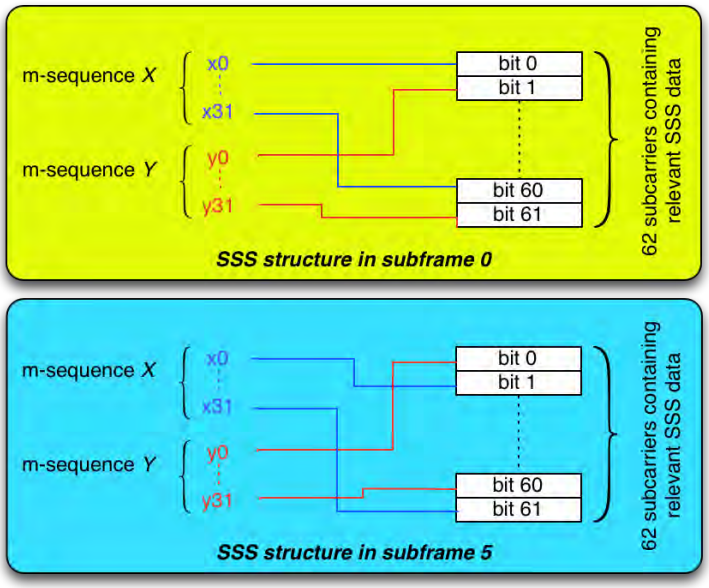

Figure 4.4: The mapping of the two 31-bit *m*-sequences to the 62 subcarriers surrounding the *DC* subcarrier (graphic based on [5]).

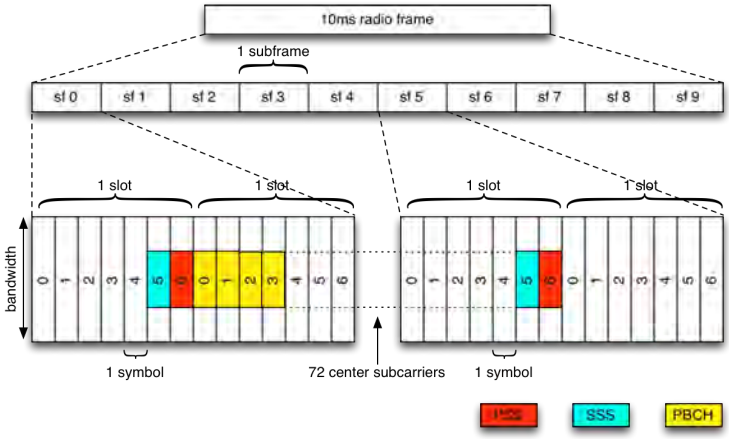

Figure 4.5: The location of the *PBCH* containing the *MIB* within a radio frame.

the context in which a resource element (or *OFDM* symbol) is transmitted. Hence, it is capable of decoding further information in the downlink.

4.1.2 Acquisition of System Parameters

In order to acquire the basic parameters of a cell, the *UE* uses two means of information. First, the *Master Information Block* (*MIB*) holds the most important cell parameters and is located in the *BCH* / *PBCH*. Second, a series of *System Information Blocks* (*SIB*s) is used to transmit further information and can be extracted from the *DL-SCH* / *PDSCH*. Both *MIB* and *SIB* messages are mapped to the *BCCH* logical channel.
As the *SIB* messages are transmitted over the *PDSCH*, it is not possible to extract *SIB* information until the information on their location has been decoded from the corresponding control channels. This process is described in Section 4.2, as it also relies on the extraction of *RNTI* values from *PDCCH*. Nevertheless, the information on the content of the different *SIB* types is already given in the upcoming paragraphs. Prior to that, however, the information contained in the *MIB* and its extraction is discussed.

Master Information Block

The *MIB* contains the information needed by a *UE* in order to decode any other downlink channels. This includes the cell's downlink bandwidth (occupying four bits, needed for determining the number of subcarriers used), the *PHICH* configuration (occupying 3 bits, necessary for decoding *PDCCH*) and the *System Frame Number* (*SFN*) (ten bits, needed to identify which number between 0 and 1023 the current radio frame has). Furthermore, the *MIB* contains 10 spare bits which are reserved for later use.
The *MIB* represents the only content of the *BCH* / *PBCH*. One *BCH* transport block is mapped repeatedly to four consecutive radio frames, whereafter the next *BCH* transport block is mapped repeatedly to the following four radio frames. Before the transport block is mapped to a ra-

Table 4.1: *System Information Block* (*SIB*) types, based on [5].

SIB Type	Contents
SIB1	Operator information, connection restrictions (e.g., whether certain users cannot access a cell), scheduling of further *SIB*s with indices > 1.
SIB2	Parameters necessary for uplink access to a cell.
SIB3	Cell reselection information.
SIB4—8	Information on neighbouring cells, including non-*LTE* cells.
SIB9	Name of the home *eNodeB*.
SIB10—12	Public warning messages, e.g. in case of a natural disaster.
SIB13	Multicasting information.

dio frame, a 16-bit *Cyclic Redundancy Check* (*CRC*) suffix is appended to the 24 bits of actual *MIB* data. This suffix is used as a checksum, in order to determine faulty *MIB*s. The resulting block of 40 bits is then coded, rate matched and modulated with *QPSK*, as defined in the *LTE* specifications [42]. Finally, the resulting *OFDM* symbols are mapped to corresponding *PBCH* locations in the time-frequency grid. As for the *PSS* and *SSS*, not the whole downlink bandwidth is used. Again, the *PBCH* symbols are only transmitted on the 72 subcarriers surrounding the *DC* frequency. This enables the reception of the *BCH* / *PBCH* information, before a *UE* has determined the bandwidth of a cell.
Within these 72 centre subcarriers, the *PBCH* is mapped to resources next to the synchronisation signals discussed in Subsection 4.1.1: It is situated in subframe zero (*not*, however, in subframe five), within the first slot, where it occupies the first four symbols. Hence, it is mapped directly after the *SSS* and the *PSS* inside subframe zero, as depicted in Figure 4.5.

System Information Blocks

The *SIB* information messages are not transmitted within the *PBCH*. Rather, they are transmitted in the *PDSCH*. Their exact position is defined by the transmission of a *PDCCH* message marked with a *SI-RNTI*. Hence, the regular procedure for decoding a message from the *PDSCH* is needed in order to retrieve *SIB* data. This is the same procedure as the one needed for the decoding of regular, *UE* dedicated *PDSCH* transmissions and is discussed in Section 4.2. Nevertheless, the different types of *SIB* messages are already discussed here. It should be noted that not all of these *SIB* messages are necessary in order to use a cell and, in general, the decoding of downlink data is possible without the knowledge of any *SIB* data. Yet, it provides interesting information when trying to analyse the capabilities and properties of a specific cell.
The different types of *SIB* messages differ in the information they carry. A compilation of the most common *SIB* messages is shown in Table 4.1. Not necessarily all *SIB* types are present in a cell, if they contain information which is not relevant in the given cell. The information that can be exploited for signal analysis purposes is located in *SIB1* and *SIB2*. While *SIB1* contains general information needed for downlink access to a cell, *SIB2* is needed in case of uplink signal analysis.

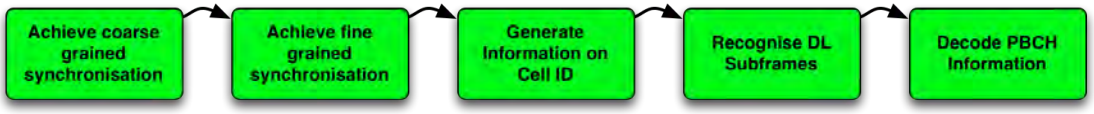

Figure 4.6: The steps necessary in order to enable downlink capturing, represented as a workflow.

Before *SIB*s are transmitted, they are mapped to so-called *System Information* messages (*SI*s), which represent the transport blocks that are used for transmission. However, this mapping depends on the choice of the operator for most *SIB*s and we refrain from discussing the details here. The only mapping, which is consistent, is the mapping of the *SIB1* to *SI-1*, as it contains the information on the scheduling of the remaining *SIB*s. The *SI-1* is always transmitted within subframe five of a physical resource block. However, its bandwidth and the occupied resource blocks are flexible and need to be determined from the corresponding *PDCCH* message.

4.1.3 Workflow for Downlink Capturing

In the previous sections, the steps necessary for achieving cell synchronisation and retrieving cell information were laid out. In order to implement these steps, it is necessary to define their order and create a workflow. We have seen that the detection of two synchronisation signals is inevitable to initialise any form of communication with the cell. Hence, we define that we achieve a coarse grained synchronisation in a first step by means of the *PSS*, whereafter a fine grained synchronisation by means of the *SSS* should be accomplished. After these two signals have been identified, the information on the physical cell identity should be calculated, in order to enable cell signal reception. Finally, the recognition of individual downlink subframes is possible, which enables to decode the information from the *PBCH*. These steps are depicted as a workflow in Figure 4.6.
After following these steps, the structure of the downlink time-frequency grid is known to the device receiving these signals. Hence, it is now possible to continue with the reception of specific downlink channels and messages. In order to stay synchronised with the *eNodeB*, the steps defined in the workflow should be executed continuously.

4.2 RNTI Extraction

The execution of the workflow defined in the previous section yields the prerequisite for decoding the actual control and user information, which is transmitted in the downlink time-frequency grid. In this section, the mechanisms for decoding downlink messages and control information are treated.
First, a general overview of the mechanisms for downlink control messaging is given in Subsection 4.2.1. In particular, the structure and responsibilities of the *PCFICH* and *PDCCH* are discussed. Furthermore, the mapping of *PDSCH* messages in relation to downlink control messaging is highlighted.
Subsequently, Subsection 4.2.2 describes the process of *CRC* insertion in *DCI* messages and the possibility it provides for extracting *RNTI* values from *PDCCH*.

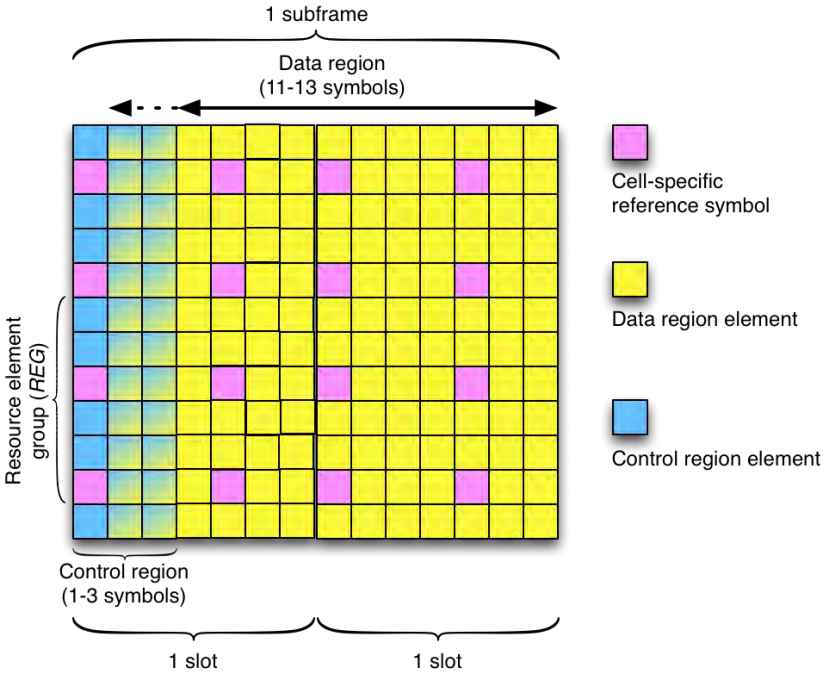

Figure 4.7: The mapping of downlink control and user information in a regular subframe.

Finally, Section 4.2.3 presents the resulting workflow that we define for the extraction of *RNTI* values and *SIB* messages.
The technical specifications used in this section are based on [4] and [5].

4.2.1 Downlink Control Signalling

Downlink control signalling is needed for several purposes: First, the most obvious purpose is the scheduling of down- and uplink resources for a specific *UE*. Second, control signalling is needed in order to determine the quality of the communication with a *UE*. For this purpose, the reception characteristics of special signals transmitted from an *eNodeB* to a *UE* can be measured and reported. Moreover, handover decisions can be performed. Finally, control signalling is used for reporting the success or failure of transmissions to and from the *UE*.
For the measurement and determination of the signal quality in a cell, mechanisms such as the *Cell-specific Reference Signal CRS* can be used. The success or failure of transmissions and triggering of retransmissions is accomplished by means of *Hybrid Automatic Repeat Request* (*HARQ*, see [45]) in the *PHICH*. We refrain from discussing both of these mechanisms, as they are not relevant for our implementation.
All control channel data is always transmitted in the first part of each subframe, in order to allow its usage for the following user data region of a subframe as fast as possible. The size of this control data part of a subframe can vary between one, two or three *OFDM* symbols, as depicted in Figure 4.7. The number of *OFDM* symbols used can vary in each subframe and aggregate carrier. The contents of the control region is made up of *PCFICH*, *PDCCH* and

PHICH data. The former two are needed in our implementation and their decoding is discussed in the subsequent paragraphs. As known from Subsection 3.2.2, these channels are defined directly in the *PHY* layer and, thus, are not mapped to any transport or logical channels.

Physical Control Format Indicator Channel

The purpose of the *PCFICH* is to indicate how the control region of a subframe is structured. The *PCFICH* is always located in the first symbol of a control region, as no information of the size of the whole control region is yet known and the first symbol is always part of the control region. It can define one of the three sizes of the control region of a subframe of either one, two or three *OFDM* symbols. Hence, it is made up of a 2-bit value. These two bits are then coded into a 32 bit word, which is scrambled (by means of a cell- and subframe dependant procedure) and, finally, modulated by means of *QPSK*. As *QPSK* can represent 2 bits in an *OFDM* symbol, 16 symbols are mapped to the control region.
However, not the whole control region is occupied by the *PCFICH*. Apart from the *cell-specific reference signal* (*CRS)*, which occupies every third subcarrier, some parts of the remaining subcarrier pairs are used for other control information, such as the *PDCCH*. In order to define which subcarriers are used by the *PCFICH*, we introduce the term of *resource element groups* (*REG*s). A *REG* consists of two neighbouring subcarrier pairs separated by a *CRS*, as shown in Figure 4.7. Only every fourth *REG* of the first *OFDM* symbol of a subframe contains *PCFICH* data, where the starting point is defined by the physical cell identity. A *REG* contains four *OFDM* symbols with *PCFICH* data and, thus, four consequent *REG*s assigned to the *PCFICH* are needed for the transmission of the 16 symbols that define the length of the control region.

Physical Downlink Control Channel and *Physical Downlink Shared Channel*

As soon as the size of the control region of a subframe is determined by means of the *PCFICH*, the remaining part of the control region can be decoded. This includes the *PDCCH*, which carries *Downlink Control Information* (*DCI*) messages. These messages are used for various aims. For example, down- and uplink scheduling assignments are granted or power control directions are sent to a *UE*.
As several *DCI* messages are available, a selection of the *DCI* messages relevant for our purposes first needs to be done. Our environment does not support *MIMO* or spatial multiplexing techniques, as they would demand hardware capabilities beyond the financial means of this thesis. Hence, the main interest lies in *DCI* messages that do not rely on any such techniques. The resulting *DCI* message formats for our purposes are, therefore, formats *1*, *1A* and *1C* as specified in [49] (see also [5]). While formats *1* and *1A* are used for downlink resource assignment purposes, format *1C* carries special information, such as responses to random access requests or system information. Hence, it is format *1C* that is needed for receiving the *SIB*s mentioned in the previous Subsection.
By means of decoding the mentioned *DCI* formats, we can determine the allocation of the corresponding *PDSCH* data and also retrieve the system information, which gives us relevant data on the configuration of a cell. This provides, on the one hand, the possibility to analyse how often a user is assigned user data space. However, the transmission of *PDSCH* data is not restricted

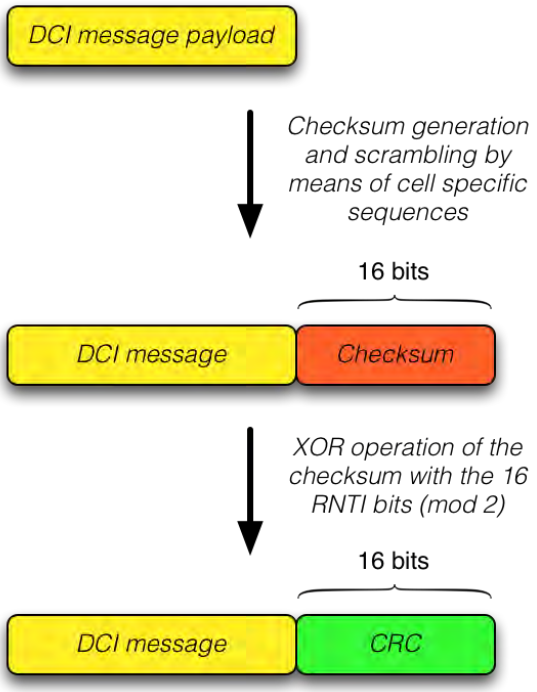

Figure 4.8: The *CRC* attachment procedure for *DCI* messages.

to user data: It could also be used to change the *C-RNTI* of a specific user or inform of other parameter adjustments. We can also not decode the actual contents of user dedicated *PDSCH* messages, as they are encrypted by higher layers. On the other hand, we get the possibility to decode system information that is transmitted to all *UE*s over the *PDSCH*.

4.2.2 *CRC* Generation for *DCI* Messages

We have seen how control and user data is assigned to the time-frequency grid in the previous subsection. However, one important aspect during the creation of *DCI* messages has not yet been discussed: The process how a *DCI* message is "marked" so that the receiving *UE* knows it has to decode it. This is done by means of the addition of a reversible algorithm, which generates a *Cyclic Redundancy Check* (*CRC*) suffix that is appended to each *DCI* message. The *CRC* can be used twofold: On the one hand, it enables an integrity check by means of the checksum. On the other hand, the appended *CRC* can be used to determine which user or users a *DCI* message is directed to.
The generation of the *CRC* suffix is illustrated in Figure 4.8. First, a 16-bit checksum of the *DCI* message is generated. The length of the checksum is fixed to 16 bits, as this corresponds to the length of a binary representation of any *RNTI* value (an *RNTI* may take values between 1 and 65'555). Thereafter, the resulting checksum is *XOR*ed with the corresponding *RNTI*: If the message is dedicated for a distinct *UE*, the corresponding *C-RNTI* is used, for messages announcing system information, paging or random access data, the *SI-RNTI*, *P-RNTI* or *RA-RNTI* are used,

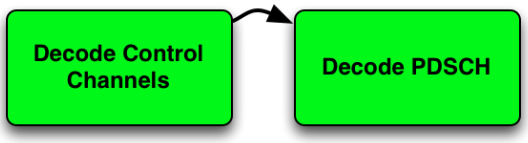

Figure 4.9: The steps necessary for decoding downlink messages, represented as a workflow.

respectively. This results in a *DCI* message with a *CRC* attachment, which is then processed by the *PDCCH*. As the generated checksum is *XOR*ed with the *RNTI*, the *RNTI* can be reverse engineered: First, the payload of a *DCI* message is used to generate the checksum, then the received *CRC* suffix can be used to find the corresponding *RNTI* from the generated checksum. As the *CRC* is used to determine a *DCI* message's integrity, the process of reverse engineering the *RNTI* could generate false *RNTI*s due to bit errors. This could increase the number of detected *RNTI*s in a measurement of *DCI* message. Nevertheless, the number of falsely received *CRC*s is small can be neglected as *measurement error* for our implementation.
After the *CRC* attachment, the messages are regularly processed by means of coding, rate matching, scrambling, *QPSK* modulation and mapping to resource elements. The details of these procedures can be looked up in the corresponding literature [4, 5] and specifications [50]. The sequences used to generate the checksums of *DCI* messages are described in [49].

4.2.3 Workflow for *RNTI* and *SIB* Extraction

We have seen how messages can be extracted from the time-frequency grid by means of decoding specific channels. The aim of this Subsection is to define a workflow for this process using the findings from the previous subsections.
We have seen that, in order to be able to extract any messages, we first need to get an understanding of which areas of a subframe are used for control data. By retrieving this information, we also know which areas can be used for user data, i.e. the *PDSCH*. Thereafter, the contents of more complex control information can be decoded: The *PDCCH* and, more specifically, the *DCI* messages interesting for our purpose can be decoded. Furthermore, the structure of the *PDSCH* messages can be extracted, which gives us the opportunity of extracting *SIB* messages. Thus, we define two steps that need to be executed continuously: First, the decoding of control information in the corresponding channels and second, the decoding of payload data in the *PDSCH*. An illustration hereof is shown in Figure 4.9.

4.3 Resulting Workflow

In the previous sections, we described the steps necessary in order to achieve an analysis of specific information transmitted in a cell. First, we defined how synchronisation to a cell is achieved and basic parameters of a cell can be determined. This enabled the recognition of all the downlink traffic transmitted in the cell. Afterwards, the procedures needed in order to decode specific parts of the communication and, ultimately, extract cell parameters of interest and *RNTI*s used were outlined.

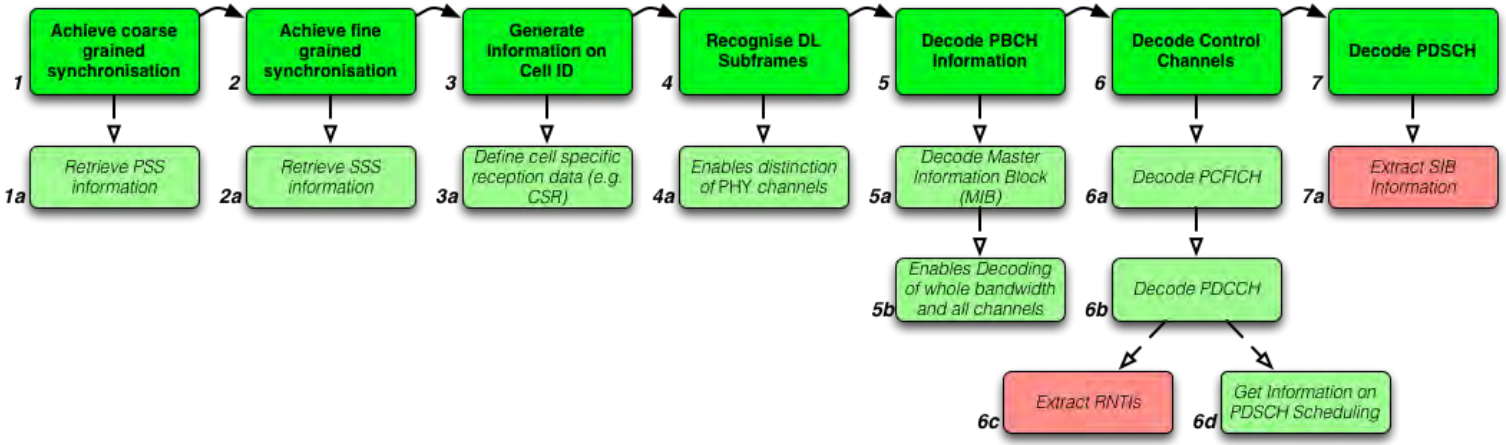

Figure 4.10: The steps necessary in order to enable downlink capturing, represented as a workflow.

Additionally, every section concluded with the definition of the steps necessary for determining the information needed. As a consequence, we want to combine the two workflows defined in the previous sections in order two define the overall execution workflow for our implementation. The final execution workflow is depicted in Figure 4.10 along with the most important results or actions of every step. Moreover, the two main goals of the workflow—the extraction of system information from a cell and the extraction of *RNTI* values used in the *PDCCH*—are marked by red boxes. This workflow functions as a theoretical basis for the implementation presented in the next chapter. The meaning of the individual steps and results of Figure 4.10 are shortly summarised subsequently.

Boxes 1 and 2 represent the detection of the two synchronisation signals needed in order to achieve time synchronisation with a cell. As this process also includes the retrieval of information necessary for the generation of the physical cell identity, these results are shown in Boxes 1a and 2a. Box 3 represents the generation of the physical cell identity, which enables the determination of the *CSR* (see Box 3a). These first three steps are needed for the determination of individual transmission symbols from the time-frequency grid, as represented by Box 4 and, subsequently, for decoding channel information from these symbols (see Box 4a). As we first need to determine the transmission bandwidth in order to decode any further channels, this information is decoded from the *PBCH* (see Boxes 5, 5a and 5b). It is now possible to decode further channels in the downlink. As we are interested in the decoding of control channels, this step is shown in Box 6. First, the *PCFICH* needs to be decoded (see Box 6a) in order to further decode the *PDCCH* (see Box 6b). From the decoded *PDCCH* we can extract the *RNTI*s used and, furthermore, retrieve information necessary for decoding the *PDSCH*. Ultimately, Box 7 shows the decoding of this channel, which yields the possibility to access desired *SIB* messages. In our case, we are especially interested in the messages of type *SIB1*, as they provide information on the operator.

Chapter 5

Implementation

We have found a general understanding of the *LTE* system architecture and, furthermore, outlined some specific procedures in greater detail in the previous chapters. As we now have a theoretical basis for designing *LTE* components, we pass on to the summary of how a signal analyser for *LTE* has been implemented. For this purpose, the individual components needed in order to accomplish a software defined solution for the analysis of *LTE* messaging are first described in Section 5.1. Thereupon, The specific software-side implementation is shown in Section 5.2.

5.1 Implementation Components

Several components are necessary in order to enable *LTE* signal analysis by means of software. Some of these components have already been introduced in Chapter 2. The *Universal Software Defined Radio Peripheral* (*USRP*) device family has been found to be a good platform for *SDR* purposes and the *OpenLTE* project builds a good basis for the software side development. However, further components which do not fall into the categories of software and hardware had to be used during the measurements.
These three parts are discussed in the upcoming subsections. First, Subsection 5.1.1 presents the hardware used and the decisions for the selection of certain hardware components. Subsequently, Subsection 5.1.2 describes the software set-up. Finally, Subsection 5.1.3 describes further components that were used and do not fall into the category of hard- or software.

5.1.1 Hardware

As already mentioned during the introduction of this section, the *USRP* device family, described in Section 2.1 has been used during the implementation. The drivers for this choice are twofold: First, these devices are directed to academic and research use cases. They are powerful and their specifications are suitable for applications in *LTE*. Second, in contrast to industrial devices with similar capabilities, the *USRP* devices are available at a reasonable cost.
For the implementation, two *USRP* devices were chosen: The *USRP N210* [8] can be connected to a machine by means of a *GBit Ethernet* connection. Even though it is a very powerful device that supports a processing bandwidth up to approximately 20 MHz, it offers lower flexibility.

Figure 5.1: A *USRP N210* device for *Software Defined Radio* reception [8].

Figure 5.2: A *USRP B210* device for *Software Defined Radio* reception [9].

Table 5.1: Overview of the machines used for *SDR* processing.

Component	Machine Alpha	Machine Bravo
CPU	Intel®Core™ i5-2400, four cores @ 3.10GHz	Intel®Core™ i5-3470, four cores @ 3.20GHz
Memory	16 GB	4 GB
External Memory	512 GB S-ATA SSD, up to 6 Gb/s bus speed	1 TB S-ATA HD, up to 3 Gb/s bus speed

One major drawback concerning flexibility is that the sampling rate can only be tuned to fixed rates. Even though this issue does not actually restrict the reception at arbitrary sampling rates, as it can be corrected on the software side of signal processing, it increases the processing overhead. For this reason, the *USRP B210* [9] has been chosen for later measurements. It offers flexible tuning to arbitrary sampling rates and also a higher maximum system bandwidth of approximately 56 MHz. The *USRP B210* is, however, not operated over a *GBit* network interface. Rather, it uses a *USB 3* (*"superspeed"*) connection. According to *Ettus Research*, this also represents its major drawback [51]: The *USB 3* bus performance can vary highly and the *USRP N210* is recommended for applications requiring high performance. As a consequence, it has been chosen for the initial measurements. However, experiments showed that the *USRP B210* actually performs better for the purpose of this thesis. It is assumed that the better results found with the *USRP B210*, which contradict the vendor's recommendation, are experienced due to the sampling rate flexibility and the resulting reduction of processing overhead as well as the use of high quality *USB 3* hardware. Detailed technical specifications and instruction manuals for both *USRP* devices are available online [19].

The second important piece of hardware when performing signal analysis by means of software defined radio is the computer that performs the actual radio processing. It represents a major component, as it has a massive impact on the performance of *SDR* applications.

In our case, two different machines have been used. The reason for the choice of two different machines has less to do with performance than with flexibility: On the one hand, measurements were performed in mainly two different locations. Using two distinct machines made logistics a lot easier when performing these measurements. On the other hand, using two machines permits simultaneous measurements, making comparisons between the measurements possible.

An overview of the two machines used is given in Table 5.1, including the most important components, which have an influence on processing. For the purpose of distinction, we call one machine *Alpha* and the other *Bravo*. As evident from the table, the two machines have similar *CPU* speeds but differ in terms of memory capacity and external memory speed, due to limited implementation resources. Machine *Alpha* is clearly faster. However, this was not found to have a general influence on our measurement results.

Table 5.2: Overview of the software used for *SDR* processing.

Operating System	Ubuntu 12.04.5 LTS (GNU Linux)
GNURadio version	3.7.6.1
OpenLTE version	00.18.04 and 00.18.03

5.1.2 Software

The software used has already been introduced in Chapter 2. Its detailed specifications are presented in this section. The discussion of the implementation added to the mentioned projects is outlined later in Section 5.2.
Our implementation relies on the *OpenLTE* framework developed by Ben Wojtowicz [34]. The major reason for this decision is that the project comes with a set-up that provides good preliminaries for *LTE* signal analysis. *OpenLTE* provides a feature that enables the scanning of bands in order to find cells. This means that a cell search implementation is present, which continuously scans signals on the available frequencies of an *LTE* radio band. This is, on the one hand, convenient, as it allows to search for cells with a good reception. Furthermore, this feature can be adapted to continuously monitor the activities of one single cell, as described in Section 5.2.
Apart from *OpenLTE*, further software components were used that may have an influence on the measurements. First, as *OpenLTE* relies on the *GNURadio* framework, this framework was installed with the version specified in Table 5.2. Second, the *GNURadio* community [52] recommends to use the widely spread and popular Linux distribution of *Ubuntu* [53] as an operating system for running *GNURadio*. Hence, a so-called *Long-term Support* (*LTS*) version was installed on the machines used for the measurements. An overview of the software installed on both machines *Alpha* and *Bravo* is shown in Table 5.2.

5.1.3 Additional Components

It may seem that the software and hardware components define everything necessary for doing measurements. However, in order to provide a series of optimal measurements, further constructions were needed. To provide an ideal positioning of the *USRP* in a place with low interference due to building construction properties, the *USRP* was positioned on the roof of a building for a series of measurements. These installations needed additional material, in order to provide the necessary fixation and outdoor durability of the *USRP*.
Hence, a metal case for the *USRP B210* was obtained, as it is shipped as a “board only” device (see Figure 5.2). Furthermore, in order to fix the *USRP* on the roof of a building, a possibility had to be found how this could be achieved. On the roof, a handrail is located to which a wooden bar was attached by means of cable wires. Hence, the *USRP*, enclosed in a metal case, was positioned on the top of this bar by means of a wooden plate, nails, and tape. Finally, as the distance between the *USRP* and the processing machine was larger than regular *USB 3* cables allow, a special power-supplied *USB 3* cable was used to bridge the distance.
The wooden bar used for this purpose is shown in Figure 5.3, including a person for size reference as a means for better conceivability of the construction. All additional components used

Figure 5.3: Wooden bar needed for fixation of a *USRP B210* on the roof of a building.

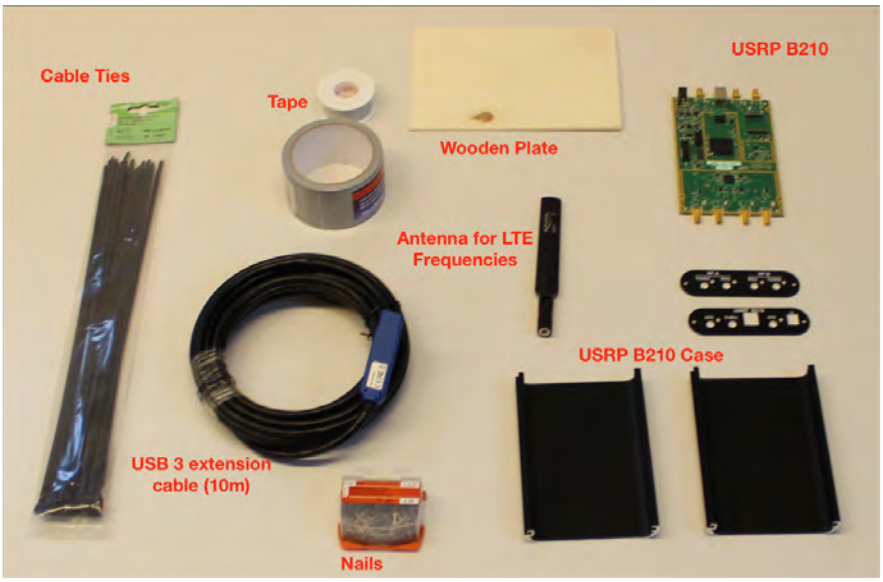

Figure 5.4: Additional components needed for the installation of a *USRP B210* on the roof of a building.

for the installation are shown in Figure 5.4. An illustration of the fixed installation on a roof is shown when describing the corresponding measurement set-up in Section 6.1.
A final component that was used during the measurements was an *antenna splitter*. It is needed in order to ensure the analysis of the same received signal by two distinct *USRP* devices and signal processing machines, as outlined in Chapter 6. This was done in order to ensure that the measurements of signals would not depend significantly on the abilities of the two machines or overflows experienced during signal processing. The antenna splitter used is shown in Figure 5.5, together with two *USRP*s of type *B210*.

5.2 Core Implementation

As the hardware and software components used for the implementation have now been introduced, we can proceed to the discussion of the implementation of the *LTE* signal analyser. In Chapter 3, the needed technological basis of *LTE* was described. Furthermore, Chapter 4 defined the steps necessary for the extraction of specific messages of interest from the downlink. It concluded with the theoretical definition of a workflow in Figure 4.10.
The aim of the current section is to illustrate the software side implementation of this workflow. The implementation of the workflow on the basis of the *OpenLTE* framework is discussed in Subsection 5.2.1. Subsequently, more specific and autonomous additions, which have been made in the *OpenLTE* framework, are discussed in Subsection 5.2.2. This subsection also defines the format of the generated measurement data files. Finally, Subsection 5.2.3 describes the additional post-processing of the measurement data files in order to plot graphs of the data in a reasonable manner. The installation of the implemented *LTE Signal Analyser* is not discussed here, as it is not of central interest for the implementation. It is found in Appendix A.

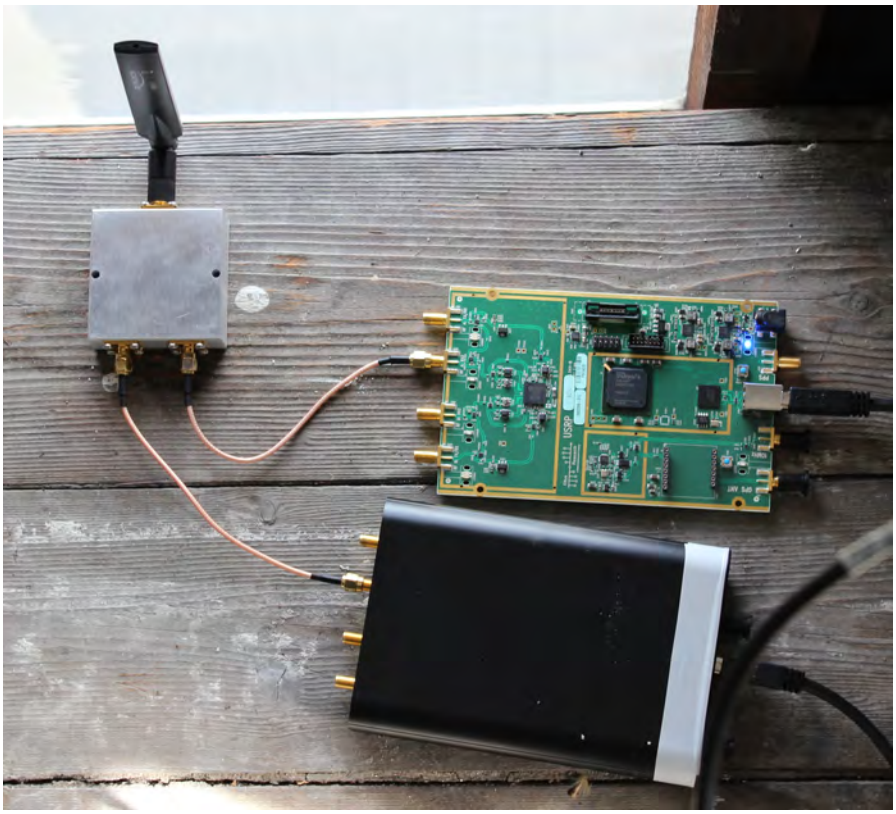

Figure 5.5: An antenna splitter attached to two *USRP B210* devices (one of them in a metal case).

5.2.1 Implementation in *OpenLTE*

This subsection describes the implementation of the workflow presented at the end of Chapter 4. We use *OpenLTE* as a basis for our implementation, as some parts are already present in this framework. Hence, this subsection discusses the present implementation and changes made to files from *OpenLTE*. As the implementation is in the *C* and *C++* programming language, the source files are split into header and implementation files. The implementation files share the same name except for the corresponding extension. For matters of readability, we refrain from making a difference between header and implementation files in these subsections.
The most important part of the files from the *OpenLTE* project for our aim is located in two directories, namely *liblte*, containing the core part of the *OpenLTE* implementation, and *LTE_fdd_dl_scan*, holding the implementation of the downlink cell search procedures. In the following, the purpose of the most relevant files contained in these folders is discussed.

Implementation File Overview

Four files form the most important part of the already present implementation of *OpenLTE*, namely *LTE_fdd_dl_scan_flowgraph*, *LTE_fdd_dl_scan_interface*, *LTE_fdd_dl_scan_state_machine* and *liblte_phy*. They are illustrated in Figure 5.6.
The class on the left hand side represents the implementation of a *GNURadio* flowgraph. The purpose of such a flowgraph is to continually process incoming signals by means of connecting *GNURadio* blocks. The most important *GNURadio* block used in the flowgraph is implemented in the *LTE_fdd_dl_scan_state_machine* class and is discussed later on. The flowgraph of *OpenLTE*, however, is not simply run as a regular flowgraph. It is controlled by commands that are invoked over an interface provided via *telnet*. This interface is implemented in the *LTE_fdd_dl_scan_interface* class. It does not only allow to start or stop the flowgraph, but also provides the possibility to change specific parameters, such as the radio band used or the centre frequencies that should be used within a radio band.
As already mentioned, the *LTE_fdd_dl_scan_state_machine* class implements a *GNURadio* block. In fact, it implements the most important radio block of the flowgraph, as it is responsible for running the tasks of the workflow defined in Chapter 4. In order to accomplish this task, it processes the signals it gets by means of specific functions from the *liblte_phy C* file. The purpose of the latter is to hold functions implementing tasks of the *LTE* specifications. The detailed outline of the functions called in the *LTE_fdd_dl_scan_state_machine* class is provided in the following paragraph.

Implementation of the Workflow

We have seen that the actual task of processing incoming signals is done within the *LTE_fdd_dl_scan_state_machine* class. By the consecutive invocation of specific functions in order to process incoming data it represents the actual implementation of the workflow defined in Chapter 4. It should be noted that the initial functionality already present in *OpenLTE* was restricted to the search for and synchronisation to cells.
An illustration of the order in which the functions from *liblte_phy* are called in

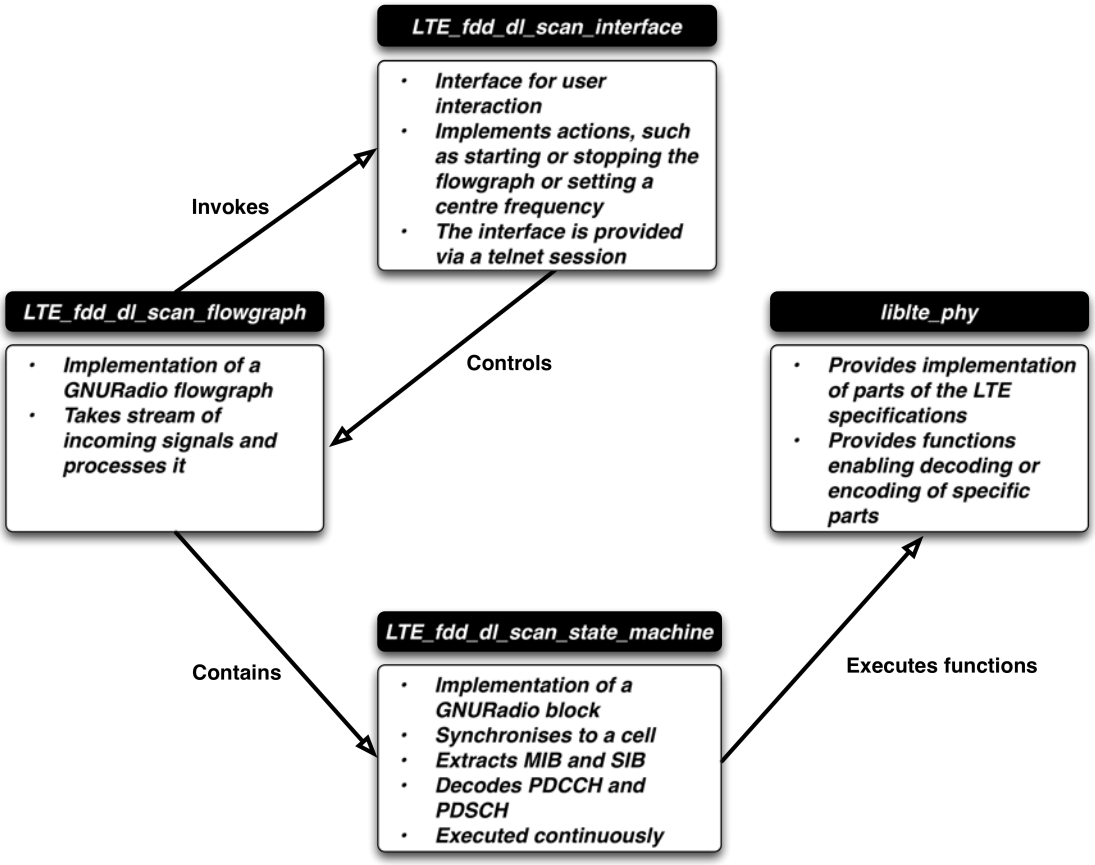

Figure 5.6: Overview of the responsibilities and relations of the most important files of the *OpenLTE* implementation.

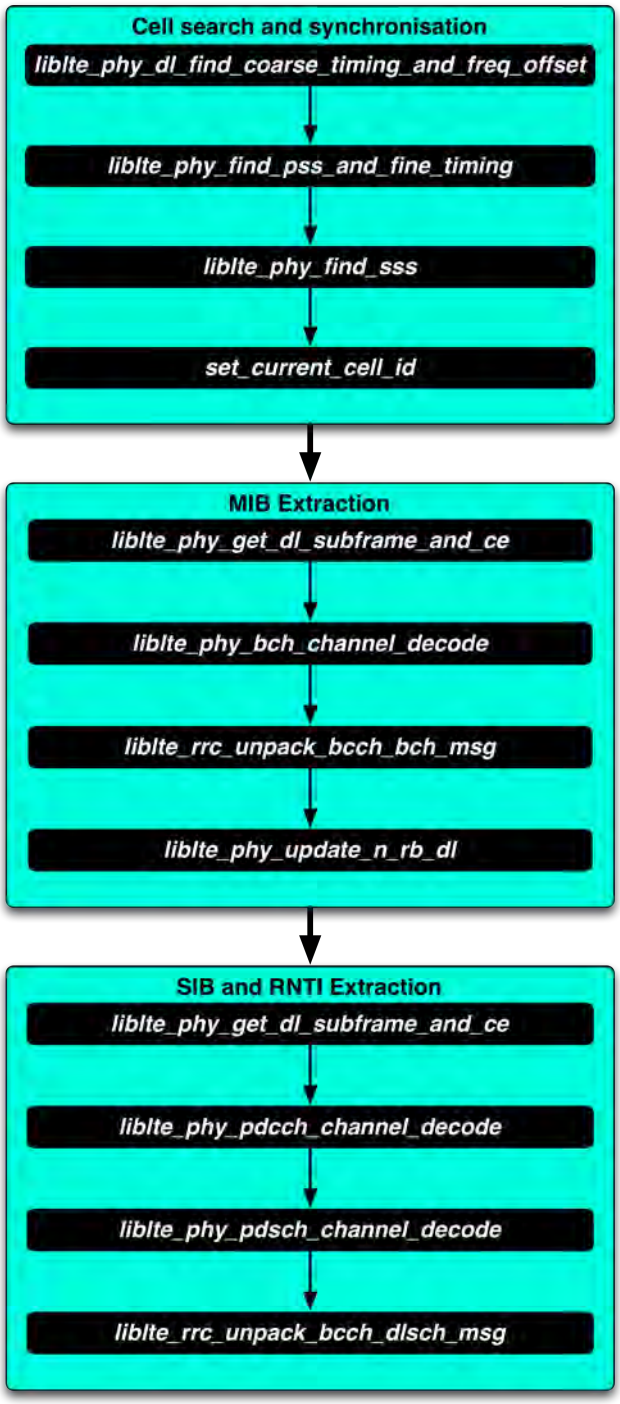

Figure 5.7: Overview of the function calls in *LTE_fdd_dl_scan_state_machine*, implementing the workflow defined in Chapter 4.

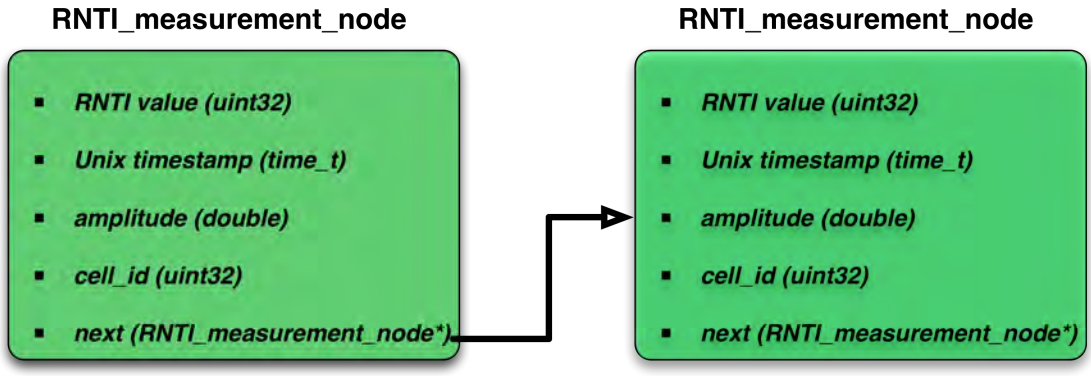

Figure 5.8: Illustration of the data structure used to store extracted *RNTI* values.

LTE_ fdd_ dl_ scan_ state_ machine is depicted in Figure 5.7. The function calls are grouped into specific greater tasks, to which they contribute. It has to be highlighted that all of these function calls appear in this order and are executed repeatedly, as data from received signals is forwarded for processing to this *GNURadio* block. The block always waits, until a certain amount of data is in its buffer and then processes it. Upon successful execution of the cell synchronisation functions, the rest of the input data is decoded. First, the *PBCH* is decoded in order to find the *MIB*, then the other physical downlink channels are decoded.
As the *GNURadio* block was created for the continuous scanning of a specific *LTE* band for cell signals, the frequency is switched after each cycle of the *work* function. This behaviour was adjusted in order to allow continuous processing of received signals with the same centre frequency. Nevertheless, short *overflows* (i.e. interruptions in processing incoming signals) are detected, as the complexity of the *GNURadio* block is high and the buffer is sometimes still full while new data for processing is forwarded. This results in the loss of a certain amount of received signals due to data processing overhead. The overflows experienced last for approximately one second, and occur on an irregular basis in the range of approximately six seconds. In order to ensure that the overflows have no significant impact on the measurement, an antenna splitter, connected to two distinct *USRP* devices and two processing machines, was used and the results compared, as discussed in Chapter 6.

5.2.2 Additionally Implemented Functions

We have seen how the overall implementation of the workflow inside the *GNURadio* block (defined in *LTE_ fdd_ dl_ scan_ state_ machine*) has been implemented. Yet, additional functions for the generation of a file containing measured and decoded data of interest have to be implemented. Even though we accomplish the extraction of the *RNTI* values from *DCI* messages in the *liblte_ phy_ pdcch_ channel_ decode* function of *liblte_ phy*, as described in Chapter 4, this detection needs yet to be stored in an appropriate format.
First of all, the storage of extracted *RNTI* values needs to be accomplished by means of an appropriate data structure. The data structure chosen for implementation is shown in Figure 5.8. It is based on the concept of a *linked list*. As evident from Figure 5.8, it does not only contain

the extracted *RNTI* value, but also stores a number of further values related to a distinct extraction of an *RNTI*. The *Unix* timestamp is included in order to make an analysis of the number of received *RNTI*s over time possible. The received signal's amplitude is calculated directly from the received samples and enables the analysis of the received signal strength encountered at the moment of reception. The physical cell identity is added, in order to distinguish signals from different cells transmitting on the same frequency. Finally, the pointer to the next node in the list is implied by the *linked list* data structure.
Naturally, the *setter*, *getter* and *add* functions for the previously defined and additionally stored data were implemented. They enable the definition ("setter") and retrieval ("getter") of the information in memory, which is needed for the creation ("add") of nodes and corresponding data for extracted *RNTI*s. We refrain from discussing the details of these functions here.
Eventually, the data structure of measurement nodes present in memory needs to be stored to a data file as soon as a measurement is finished. For this purpose, first a folder is created if it does not yet exist, in which all measurement data files are stored. This folder is called *measurement_ stats* and is located in a user's home folder. A measurement data file stored to this location carries the data and time of the beginning of the measurement in its name and a *.csv* suffix. The contents of the file are created by calling the *print_ rnti_ stats* function in *liblte_ phy*. An example output is listed in Appendix B. Its format is based on the *comma separated value* (*csv*) specification, i.e. one extracted *RNTI* and its corresponding parameters are printed on one line, separated by commas. The first line of the file contains the header of each column. At the end of the file, one line per *RNTI* type (e.g., *SI-RNTI*) is printed, containing the overall sum of the measured *RNTI*s of a specific type.

5.2.3 Plotting Script

We have seen that the measurement data file contains one line per extracted *RNTI*. As measurements may last over several days or weeks, this may result in an exorbitantly high number of lines per file, considering that a measurement lasting a minute may already yield approximately 10'000 received *RNTI*s. In order to make a reasonable plot, a *bash* script has been created, which condenses the number of *C-RNTI*s found within periods of 15 minutes. This makes it possible to generate plots of the number of *C-RNTI*s transmitted within a certain time window, while preserving the measured data for ulterior use.
The plotting script takes every line, appends it to a specific time window and processes the information contained in the line adequately. For example, each *C-RNTI* occurrence increases the counter for a specific time window. Other parameters stored within a measurement file can be combined to an average value. The details on how this *measurement plotter* works are summarised in Appendix B. It should be noted that the processing of the *csv* files by the measurement plotter can last considerably long, as the file sizes of the *csv* files can be up to several gigabytes. It has been experienced that for certain measurements the processing by the measurement plotter took nearly as long as the measurement itself.

Chapter 6

Evaluation

We have seen how signal analysis of *LTE* is enabled by the theoretical concepts presented in Chapter 3. Thereafter, a theoretical workflow has been defined in Chapter 4 and a practical implementation of this workflow has been described in Chapter 5. This chapter presents the results of various measurements and evaluates them.
However, before any measurements could be recorded or their results analysed, specific measurement locations and environments had to be defined. The environments are labelled *1a*, *1b*, *2*, *3a* and *3b*. They are presented together with the chosen locations in Section 6.1.
The measurement results of these environments are discussed in the subsequent sections: The findings from an initial cell search on all available *LTE* bands used in Switzerland (3, 7 and 20) are presented in Section 6.2. Second, the measurement results of environment 1b for measurements lasting 60 seconds each are discussed in Section 6.3. These two sections can be looked at as a discussion of preliminary results, as the results from the actual implementation, as presented in Chapter 5, follow in the succeeding Sections.
The continuous measurements, mapping a number of measurements to a specific time interval per day, are discussed subsequently. These measurements from the previously defined environments 3a, 3b and 2 are discussed in Sections 6.4 and 6.5.

6.1 Measurement Set-up and Evolution

We have described the hard- and software set-up chosen for our analysis of *LTE* signals in the preceding chapter. The practical set-up of the measurements made is outlined in this section. It includes, first of all, the different locations chosen for the placement of the antenna interfaces. They are described in Subsection 6.1.1. Furthermore, even if the same locations are used for measurements, their specific set-up may change. For example, the machine used or the focus of the analysis process may be different. We call the specific set-up of a measurement *environment*. Additionally, findings from first measurements may yield a change in the environment of following measurements. An overview of this "evolution" of measurement set-ups is given together with the definition of the corresponding environments in Subsection 6.1.2.

Figure 6.1: The general situation around the institute building, including the three defined locations.

6.1.1 Measurement Locations

Three main measurement locations resulted from the general process of research and development. All locations are situated at the *Institute of Computer Science* (*INF*) of the *University of Bern* at Neubrückstrasse 10 in Bern (Switzerland). The location of this building and its surrounding are depicted in Figure 6.1.
The choice of the first location was a pragmatic choice in favour of the developer's office in the third floor of the *INF* building. The other locations evolved from the idea that a measurement in a near-optimal environment should be done and that the office is not such a location. A major drawback in the office is the presence of a high number of (experimental) wireless devices that could influence the reception of an *LTE* signal, even if not actually using the corresponding frequencies. Furthermore, the rather old and massive skeletal structure of the building could damp signals and let the *USRP* receive signals of a lower or even useless quality. This led to the choice of locations two and three in a surrounding, which would not have a high influence on the received signal's quality. Additionally, these locations had to be near to network and power grid connections needed for the hardware. Such locations were found on the roof (location two) and in the institute's tower (location three). First, the location on the roof was chosen as a quite optimal location. The *USRP* and its antenna interface could be installed outside, yet the processing machine could be installed inside the neighbouring tower of the institute. However, the drawback of this location was its resistance to weathering for the *USRP*. Obviously, the *USRP* could not be located outside at all times, even though certain precautions were made. Thus, a third good location for the *USRP* and its antenna interface was found to be inside the tower, as its building structure is less massive than that of location one and no electromagnetic parts are included in it.
The three locations used for the measurement are illustrated on Figure 6.2. As we want to be able to refer to these three locations later on, we will define them as locations *one*, *two* and *three*

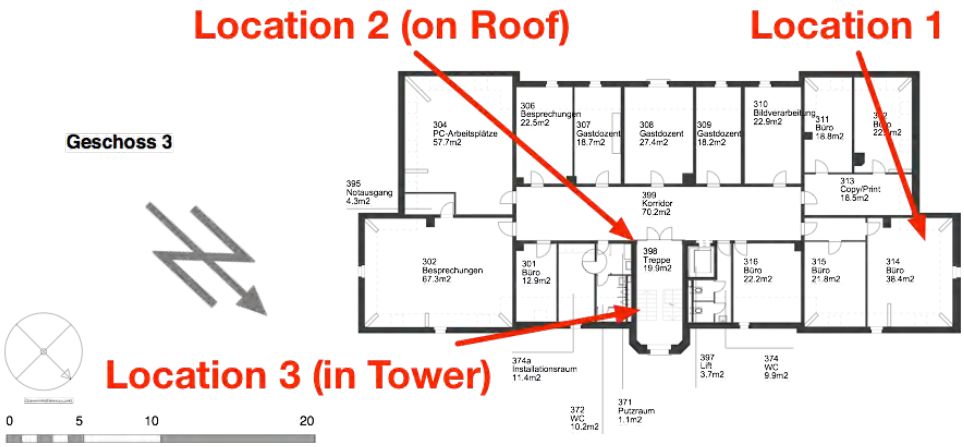

Figure 6.2: The three experiment locations illustrated on a ground plan of the third floor.

as indicated in this Figure. As evident, these locations are very near to each other. On the one hand, this had the advantage of flexibility: While the implementation and its immediate testing always took place in location one, locations two and three provided the advantages of better reception quality. On the other hand, the motivation for choosing different locations was not to measure different cells but to increase the number of analysis data for the same cell.
Some general facts should be noted concerning the location of the institute: It is situated in an urban area, near to the local main train station and several office buildings. Hence, the density of *UE*s should be quite high during regular working times. However, it should decrease as the major part of the people moves from their offices to their homes. A further factor, which could influence the number of the *UE*s, could be the large number of party locations in the near surrounding of the institute building, which could become noticeable during nightly hours, especially on weekends.

6.1.2 Measurement Environments and their Evolution

We have defined the locations in which measurements are made. The next task is to define how these measurements should be done, i.e. what devices are used and how the data is processed.
A first type of measurements was made to determine what kind of cells are available. This measurement was made by means of the initial implementation of *OpenLTE* as provided by the project's website [34]. As these were the first measurements conducted, they were done by means of a *USRP N210* in location 1. We shall refer to this environment, processed by machine *Alpha*, as *environment 1a*.
Further measurements were done with an adapted version of *OpenLTE*, which did a scan of *RNTI*s over a specific period of time but did not provide timestamped measurements. These measurements were limited to a time of 60 seconds each. They measured, how many times a distinct *C-RNTI* was encountered during that time window. However, these measurements did not enable to see when exactly which *C-RNTI* was measured. They were also done by means of

Figure 6.3: A photograph of environment 3b.

a *USRP N210* and define *environment 1b* in location 1, processed on machine *Alpha*.
Subsequently, a series of measurements was made on the roof and in the tower of the *IAM* building, i.e. locations 2 and 3. Initially, location 3 was not intended. However, location 2 caused problems because of the utilisation of a special *USB 3* extension cable, which was the reason why location 3 was defined and used. At this point, the presented implementation of Subsection 5.2.1 was used, as an interest on the number of measured *C-RNTI*s over time developed. Moreover, for these measurements and all subsequent measurements, only *USRP B210* devices were used, which operate over *USB 3* and do not require a stand-alone power connection.
We define the series of measurements in location 3 by means of a *USRP B210* and on machine *Bravo* as *environment 3a*. Additionally, a measurement by means of two *USRP B210* devices connected to an antenna splitter was performed. Obviously, both machines *Alpha* and *Bravo* were needed for this measurement, defined as *environment 3b* (see Figure 6.3).
Later on, measurement series located at the initially intended position on the roof (location 2) of the institute could be undertaken. They are referred to as *environment 2* and were processed on machine *Bravo*. An illustration of what the fixed *USRP B210* installation on the roof of the building looked like is available in Figures 6.4 and 6.5.
An overview of all measurement environments is shown in Table 6.1.

Figure 6.4: A photograph of environment 2, looking into north-east direction.

Figure 6.5: A photograph of environment 2, looking into north-west direction.

Table 6.1: Overview of the defined environments.

Environment	Location	Machine	Summary
1a	Office	Alpha, *USRP N210*	Cell scan directed at finding cells for later measurements.
1b	Office	Alpha, *USRP N210*	Measurement lasting for 60 seconds, determining the number of *RNTI*s transmitted during this period.
2	Roof top	Bravo, *USRP B210*	Measurement in an near-ideal location by means of a special installation on top of a wooden bar outside.
3a	Inside the institute's tower	Bravo, *USRP B210*	Measurement in a weather-safe location with good reception.
3b	Inside the institute's tower	Alpha and Bravo, *USRP B210*	Measurement of the same signal on two machines. An antenna splitter was installed, connected to two different *USRP* devices and processing machines.

6.2 Cell Search in Environment 1a

The set-up of the measurement for cell search has been defined by *environment 1a*. The procedure of cell search is done by means of running a basic function of the *OpenLTE* project, which searches a specific band for cells. As discussed, the *LTE* bands used in Switzerland are bands 3, 7 and 20. Hence, these bands have been scanned by means of *OpenLTE* and the results are shown in Tables 6.2, 6.3 and 6.4.
The tables contain the corresponding centre carrier frequency of the detected cells, along with the *cell identity within a group* (extracted from the *PSS*), the *cell identity group* (extracted from the *SSS*), the resulting *physical cell identity* and the bandwidth, which is extracted from the *MIB*. Detected cells of which the bandwidth could not be extracted from the *MIB* have been neglected from the listing in the tables.
Even though a large number of cells has been detected, only a small number of them has been found to be suitable for further measurements. Some cells in band 20 provide the best reception quality, as several of their *SIB* messages could be decoded. These are the cells found at 806 MHz (operated by Swisscom) and 796 MHz (operated by Sunrise). As the Swisscom cell at 806 MHz (physical cell ID 138) provided an especially large number of decoded *SIB* messages, a selection of them is displayed in Table 6.5. As known from Chapter 4, *SIB1* provides general information of a cell and indicates the parameters needed to decode the other *SIB*s, e.g. their *window length*. *SIB2* is used for the transmission of information needed for uplink access. The examples include random access channel parameters and the length of the *Cyclic Prefix* for the

Table 6.2: Cells found when scanning *LTE* band 3.

***DC* frequency [MHz]**	**Cell identity within a group (from *PSS*)**	**Cell identity group (from *SSS*)**	**Cell ID**	**Bandwidth**
1809.7	2	23	71	15
1810.1	2	53	161	1.4
1811.3	1	33	100	10
1815.1	2	129	389	20
1815.7	1	12	37	10
1826.4	1	7	22	1.4
1827.2	0	149	447	1.4
1850.1	2	33	101	20
1870	2	17	53	20

Table 6.3: Cells found when scanning *LTE* band 7.

***DC* frequency [MHz]**	**Cell identity within a group (from *PSS*)**	**Cell identity group (from *SSS*)**	**Cell ID**	**Bandwidth**
2636.8	0	34	102	15
2639	2	110	332	1.4
2660.1	2	162	488	15
2663.3	2	129	389	10

Table 6.4: Cells found when scanning *LTE* band 20.

***DC* frequency [MHz]**	**Cell identity within a group (from *PSS*)**	**Cell identity group (from *SSS*)**	**Cell ID**	**Bandwidth**
806	0	46	138	10
793.7	0	51	153	10
796	1	60	181	10
801.9	0	10	30	15
800.9	1	16	49	10
796	0	150	420	10

Table 6.5: Selection of *SIB* messages decoded from a *Swisscom* cell on 806 MHz with physical cell id 138.

SIB type	Name	Value
1	network	Swisscom
1	si_win_len	10
2	num_rach_preambles	64
2	ra_response_window_size	10
2	ul_cp_length	normal
3	cell_resel_priority	5

SIB type	Name	Value
3	neigh_cell_cnfg	2
3	t_resel_eutra	2
5	neigh_cell_cnfg[1]	2
6	thresh_x_high[0]	4
6	thresh_x_low[0]	8

uplink. *SIB3* informs about cell reselection properties, while the *SIB* messages of types 5 and 6 provide information concerning the neighbouring cells.
Because these initial measurements for the Swisscom cell (ID 138 at 806 MHz) have been found to be the best, all of the subsequent measurements have been made on this cell.

6.3 *C-RNTI* Density in Environment 1b

In environment 1b we faced nearly the same set-up as in environment 1a. However, we did not rely solely on cell scan. Rather, a continuous scan on one centre frequency was performed. As mentioned in Section 6.2, the Swisscom cell with a centre frequency of 806 MHz was used for this purpose. It was determined that this cell has a physical cell identity of 138.
For these measurements, apart from only tuning to one frequency, also an extraction of all received *RNTI*s transmitted in *PUCCH* was accomplished. However, the *RNTI*s are not yet timestamped. Rather, the measurements lasted only for a short period of 60 seconds. This allows the analysis of how often specific *C-RNTI*s are used. Ten such measurements were performed, all within the same hour, on February 18th 2015. The results showing a plot over the *C-RNTI* values on the one axis and the number of occurrences on the other axis is shown in Figures 6.6 and 6.7. For all measurements we can state the following: Only a small number of *C-RNTI*s seems to be used more than 10 times a minute. Even though the specific *C-RNTI*s that appear more often than others are not the same, this structure of a small part of the *C-RNTI*s appearing far more often than the others is striking. An assumption that can be made is that only the *UE*s that actually are in use and generate a larger number of traffic trigger a higher occurrence of their *C-RNTI*, as higher traffic would also imply more variations in control signalling on the *PUCCH* when resources on the *PUSCH* are redistributed.
In order to further support this assumption, we use a different kind of plot. It shows a scale of 100% on the y-axis and on the x-axis the number of occurrences is shown. This plot makes it possible to see how many per cent of our measured *C-RNTI*s has an occurrence of a certain number. Representatively, we look at a plot of the second measurement in Figure 6.8. We see that only a very small percentage of the measured *C-RNTI*s has a higher occurrence than 20, as the plot already nearly reaches the 100% mark after approximately five occurrences. If we scale

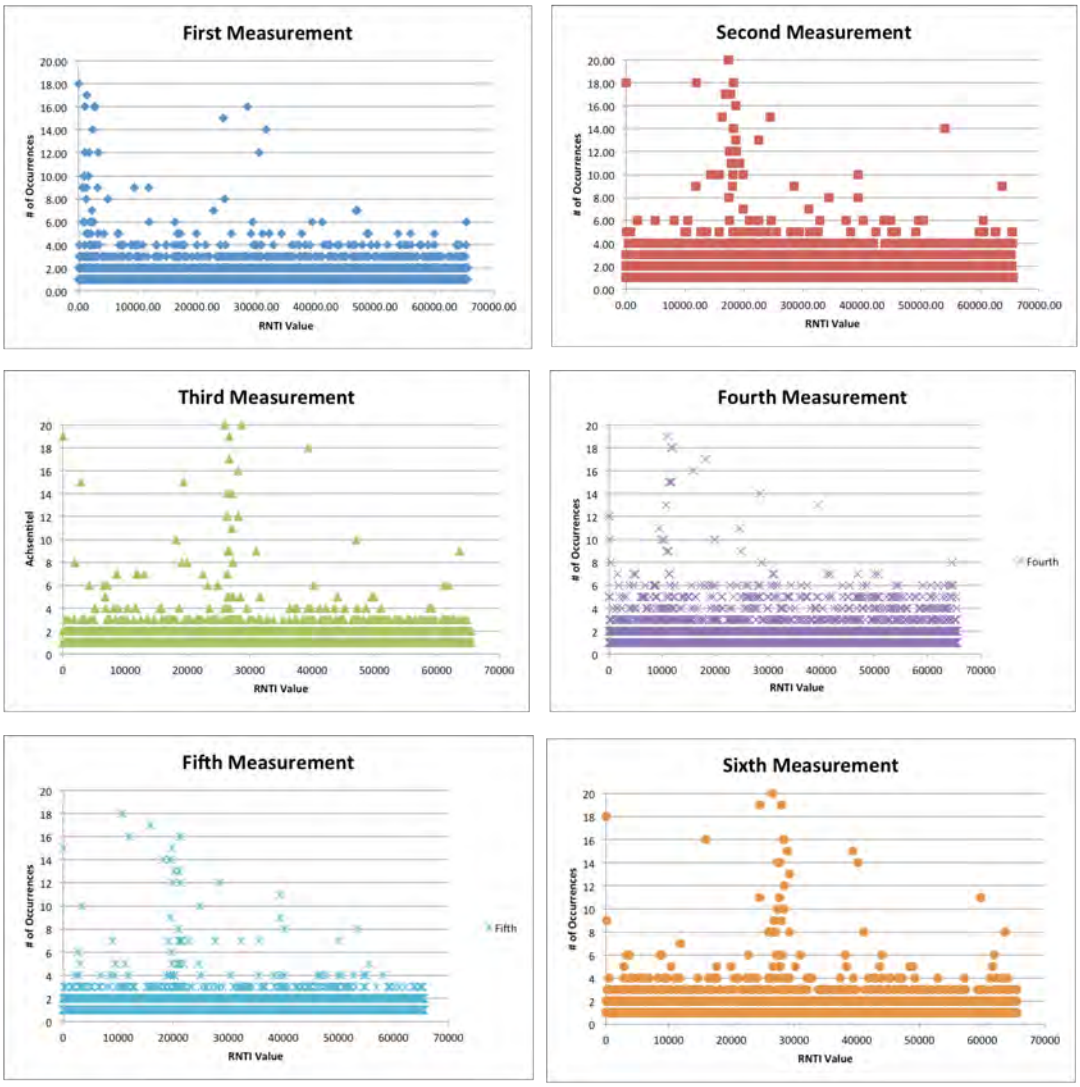

Figure 6.6: Number of *C-RNTI* occurrences in 60 seconds (measurements 1—6). The x-axis shows the number of occurrences, the y-axis the corresponding *C-RNTI* values.

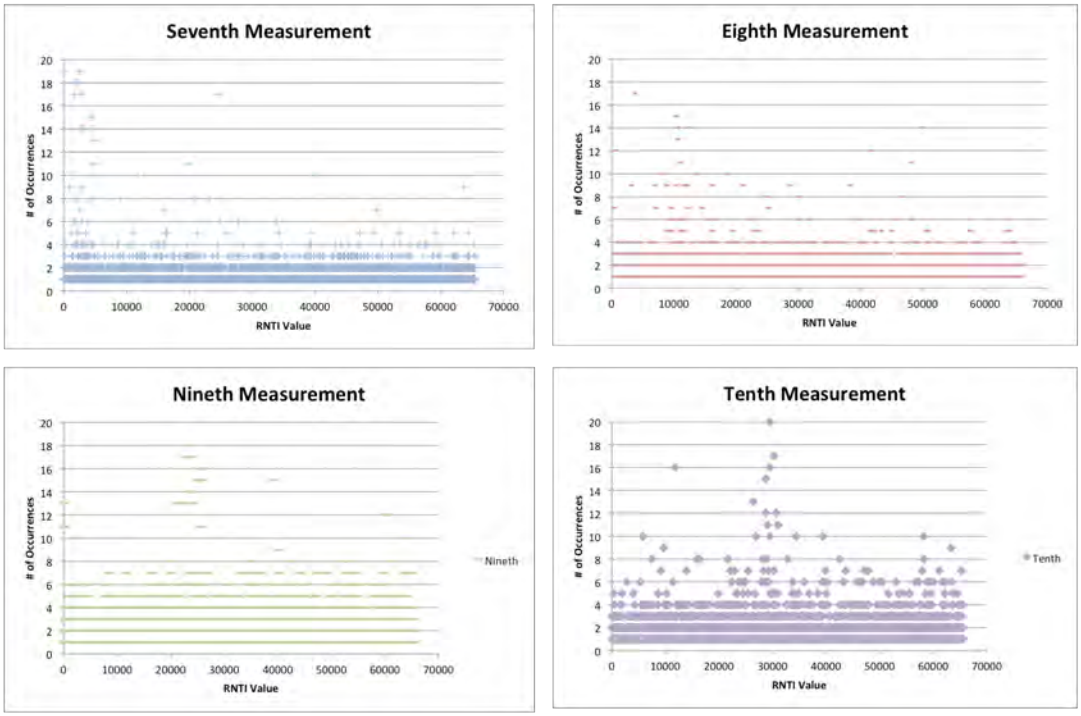

Figure 6.7: Number of *C-RNTI* occurrences in 60 seconds (measurements 7—10). The x-axis shows the number of occurrences, the y-axis the corresponding *C-RNTI* values.

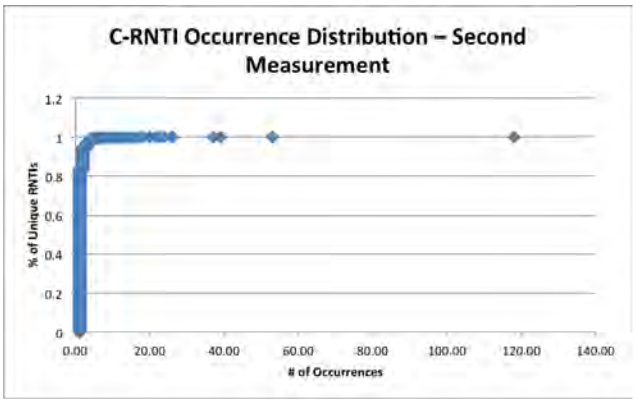

Figure 6.8: A plot of the second measurement showing all measured *C-RNTI*s scaled to a y-axis of 100%. The x-axis represents the number of occurrences for a specific *C-RNTI* value.

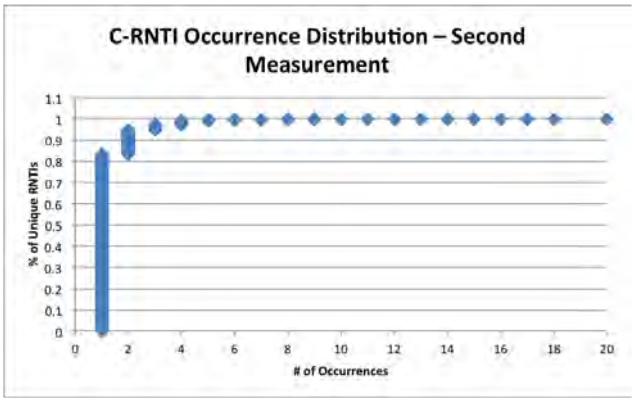

Figure 6.9: A plot of the second measurement showing all measured *C-RNTI*s scaled to a y-axis of 100%. The x-axis represents the number of up to 20 occurrences for a specific *C-RNTI* value.

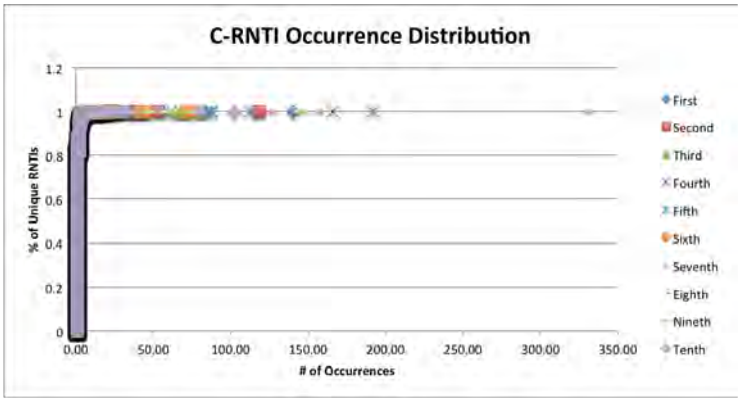

Figure 6.10: A plot of all measurements showing all measured *C-RNTI*s scaled to a y-axis of 100%. The x-axis represents the number of occurrences for a specific *C-RNTI* value.

Table 6.6: Number of extracted *RNTI*s per type for the measurements from environment 1b.

	Unique *C-RNTI*s	***C-RNTI*s**	***RA-RNTI*s**	***P-RNTI*s**	***SI-RNTI*s**
1	10896	13558	36	200	206
2	10385	13574	39	174	213
3	11107	13617	35	158	190
4	11896	15424	34	222	236
5	11095	13587	31	182	200
6	11000	13578	32	176	212
7	10904	13578	32	176	210
8	11719	15530	22	196	252
9	11406	15551	24	196	225
10	11529	15498	53	216	232

down a little, as illustrated in Figure 6.9, this effect gets even more visible. We can state that this is similar for all of our measurements, as they all have similar curves for this kind of plot (see Figure 6.10).

The analysis of the plots of this measurement series presumes that we can measure the activity of a device by means of the number of occurrences of its *C-RNTI* over a certain period. This makes sense when considering the possible *RRC* states of *idle* and *connected* devices. We can assume that only a smaller number of devices hosted by a cell is really being used. The largest number of devices is in some sort of idle mode, even if it is not in the *RRC_IDLE* state. This means, that either traffic is very low or the devices completely refrain from sending or receiving any data. Thus, the number of control messages transmitted in the downlink is very low for these devices. Consequently, the number of received *C-RNTI*s corresponding to these devices is low. This also complies with the requirements for *LTE* [23] that state that *more than 200 devices* should be able to be active simultaneously within a cell. For example, two hundred simultaneous phone calls should be possible, but a massively larger number of devices should be hosted on a cell, with either low or no activity. The second measurement and its plot in Figure 6.9, where approximately 1% (i.e. a number of approximately 100) of the devices experiences more than seven transmissions per minute absolutely lies within the requirements for *LTE*. According to this finding, at most 100 devices of our cell experience high traffic, where "high" is denoted by more than six transmissions of downlink control information per minute. These results were found to be similar for all measurements.

The measurements of environment 1b also provide statistics on how many times an *RNTI* type was encountered during a measurement (see Table 6.7 for a summary of the *RNTI* types presented in Chapter 3). The first column shows the occurrence of *unique C-RNTI*s, i.e. the number of different *C-RNTI* values that were measured. All other columns contain the overall number of measured *RNTI*s, including duplicates. As can be seen, the numbers lie within similar scopes over all measurements.

Table 6.7: Summary of *Radio Network Temporary Identifiers*.

Acronym	Description
RA-RNTI	Used for random access procedure responses.
C-RNTI	Device specific identifier (unique within a cell).
P-RNTI	Used for paging messages, e.g. triggered by phone calls.
SI-RNTI	*RNTI* used to mark the transmission of system information.

Table 6.8: Number of extracted *RNTI*s per type for the measurements from environments 2, 3a and 3b.

Measurement	*C-RNTI*s	*RA-RNTI*s	*P-RNTI*s	*SI-RNTI*s	Duration [sec]
2015-05-06	3961704	5779	30047	103538	70693
2015-05-07 Alpha	11912855	13425	102943	296963	88461
2015-05-07 Bravo	6015048	6838	51216	146743	88444
2015-05-15	6673489	7925	51155	98988	101361
2015-05-20	4137146	5835	40921	108956	92693
2015-05-21	40770751	47171	293351	746364	427196
2015-05-27	5324350	7498	53893	91216	90101
2015-05-28	2708638	3068	10816	57736	82058

6.4 Continuous Measurements in Environments 3a and 3b

The measurements discussed in this section have been made from within the tower of the *INF* building, on the same level as the roof. Environment 3a represents a simple set-up with one machine. Its results are presented and discussed in Subsection 6.4.1. Subsequently, environment 3b, consisting of two *USRP*s and two processing machines connected to the same antenna interface, is presented in Subsection 6.4.2. The number of received *RNTI* types for environments 3a, 3b as well as 2 are shown in Table 6.8.

6.4.1 Environment 3a

Several measurements lasting approximately one day have been made for the continuous observation of control signalling. They are depicted in Figure 6.11. The red lines on the x-axis mark the change of date (i.e. the location of the time *00:00*). As can be seen, both measurements that began on a Wednesday show interesting peaks, while the measurement starting on a Friday does not provide any irregular patterns and, thus, forms an exceptional case within all measurements done. Hence, we refrain from discussing it and count it as an *irregular example*.

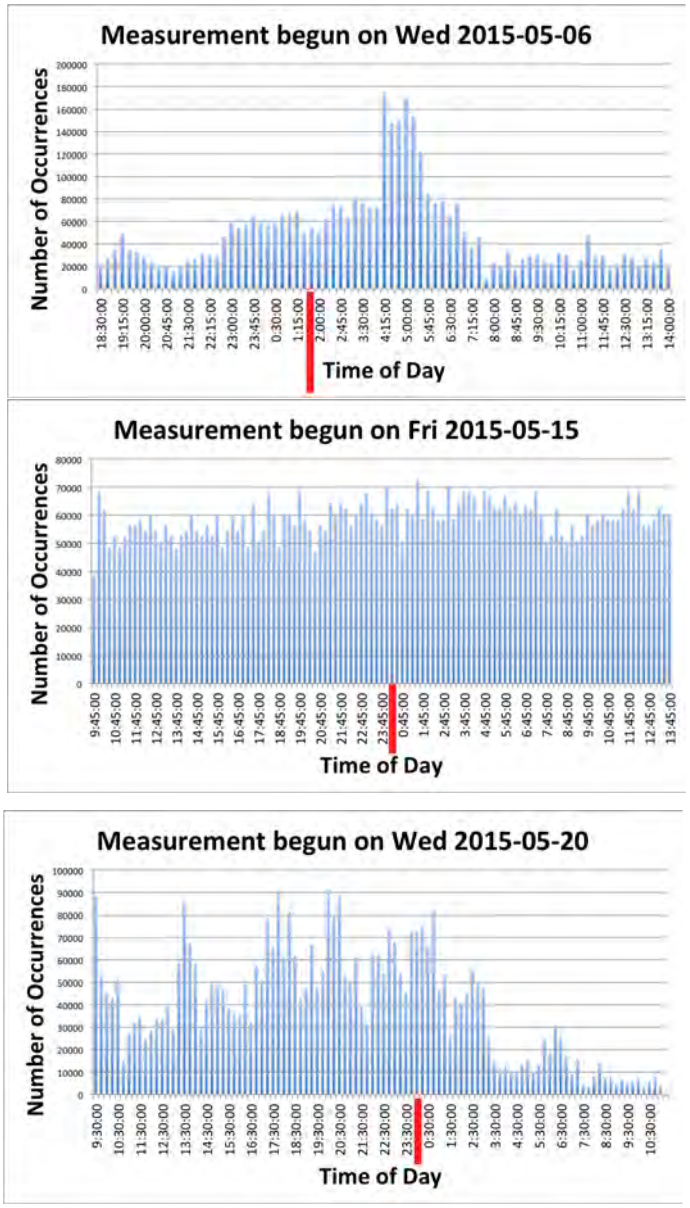

Figure 6.11: Plots showing the intensity of control signalling over time for environment 3a.

The two measurements starting on a Wednesday show an interesting but counter intuitive pattern: In particular, in the measurement starting on 2015-05-06, control signalling reaches a significant peak between approximately 4 am and 6 am of the following day. On the second measurement, starting on 2015-05-20, we see a similar peak during similar times. However, it is significantly smaller compared to the maximum bar values experienced during the preceding hours. The interpretation of these different behaviours is difficult. We would need knowledge on the operator's data. The results displayed in the plots are against the intuition that the larger amount of traffic, that would be expected during day time, also triggers a larger amount of control signalling. It is possible that the remarkable sudden increases and decreases are triggered by neighbouring cells that are switched on or off, depending on the overall network load, resulting in a sudden increase or decrease of *UE*s served by the cell measured. However, this assumption is difficult to prove. An indication of the average received signal strength (*RSS*) would be helpful, as this would indicate that the range of the cell would be increased, if the *RSS* value would be higher. The *RSS* value, however was not recorded until a later series of measurements.
In Figure 6.12 we see a measurement that has been made over several days, beginning on Thursday (2015-05-21) and lasting over Pentecost public holiday. Once again, we experience a significant peak. This time, it begins at approximately 6 am on Sunday morning and lasts approximately until 10 am on Monday morning. Once again, we could argue that this may be due to switching off a neighbouring cell, as many people go into the countryside during a holiday such as Pentecost.

6.4.2 Environment 3b

The motivation for this environment was given by the rather unintuitive results presented in the previous subsection. The aim was to ensure that the results are not influenced by overflows or processing power differences of the signal processing machines. An overflow is experienced when the incoming signals of a *USRP* cannot be processed fast enough by the processing machine, as described in Chapter 5. As a consequence, the corresponding samples are lost. Overflows occur on an irregular basis.
In order to rule out the option that the experience of overflows could have a significant influence on the measurements, two machines and their *USRP*s were connected to the same antenna interface by means of an antenna splitter. This also enabled to determine, whether the different processing power of the two machines has a significant impact on the measurements. The resulting bar diagrams should be ideally the same. However, they should at least show a similar relative structure of peaks.
As can be seen in Figure 6.13, measured on machine *Alpha*, and Figure 6.14, measured on machine *Bravo*, the measurements on both machines result in bar diagrams with similar relative values and peaks. The main difference is the absolute number of *C-RNTI*s discovered, which can be explained by the significant difference in *RAM* and external memory between the two machines. However, the relative structures of both diagrams are very similar. Hence, the type of machines or experienced overflows should not have an influence on the peaks found in a measurement's plot.

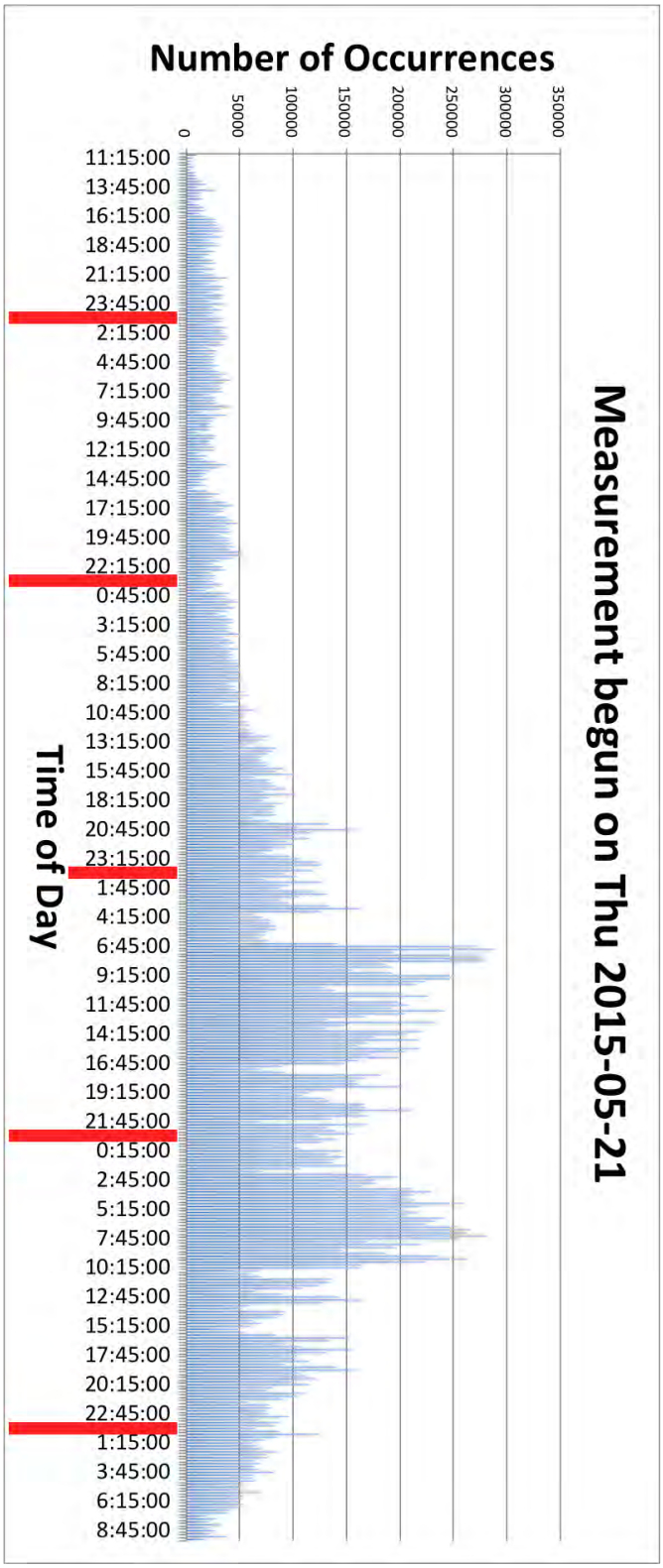

Figure 6.12: A plot showing the intensity of control signalling over time. This measurement has been made over several days in environment 3a.

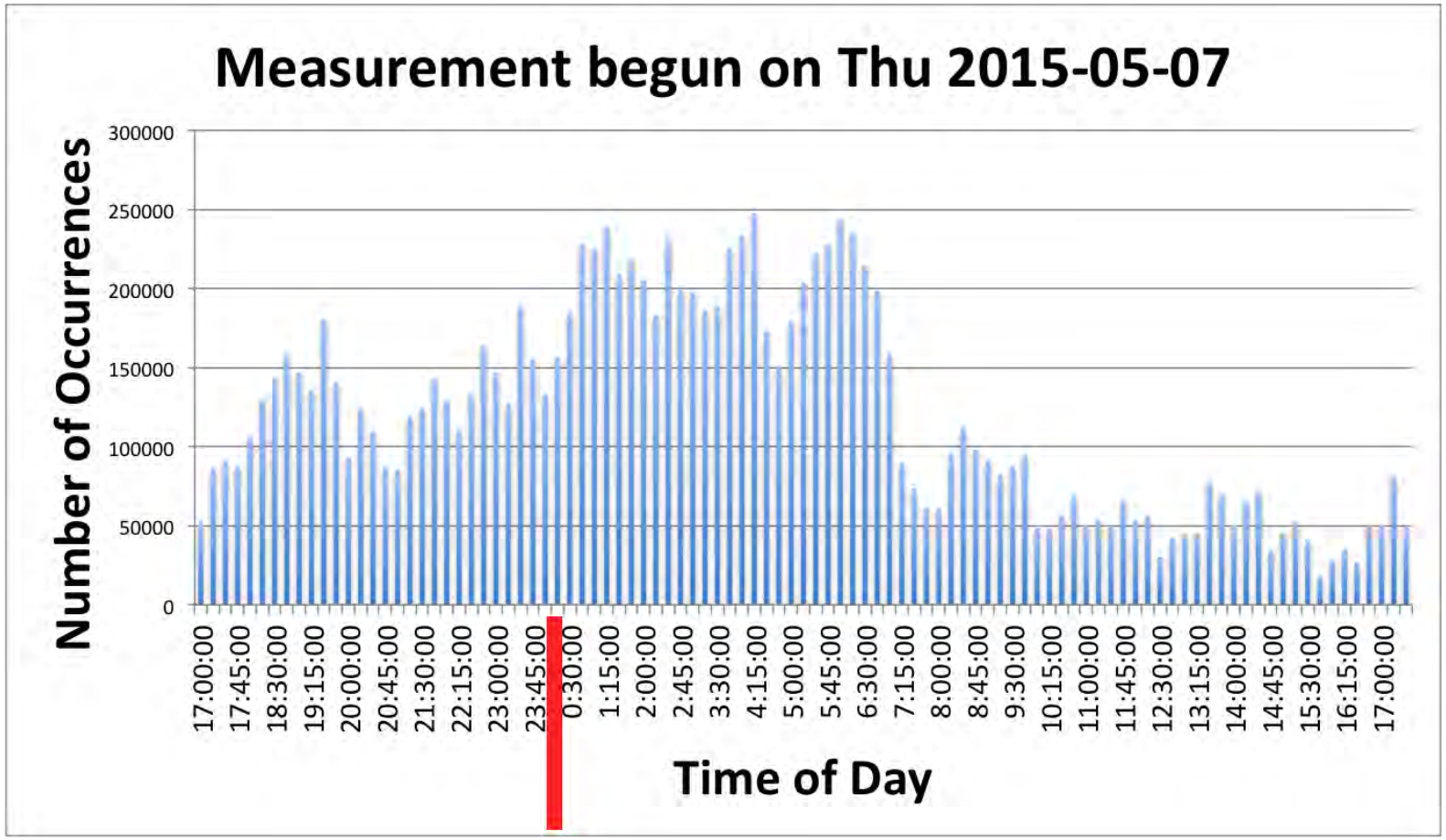

Figure 6.13: Plot showing the intensity of control signalling over time on machine *Alpha* (environment 3b).

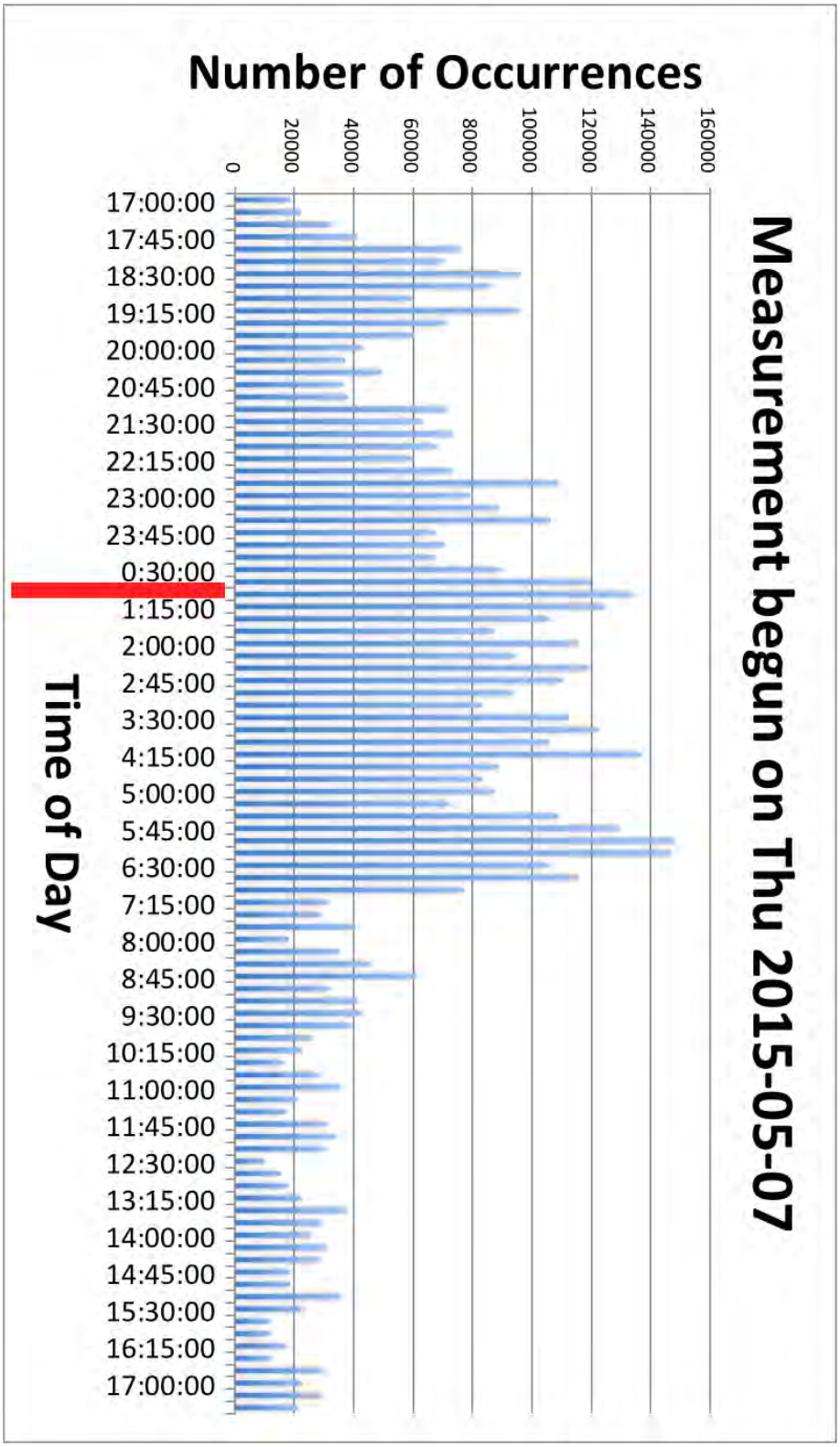

Figure 6.14: Plot showing the intensity of control signalling over time on machine *Bravo* (environment 3b).

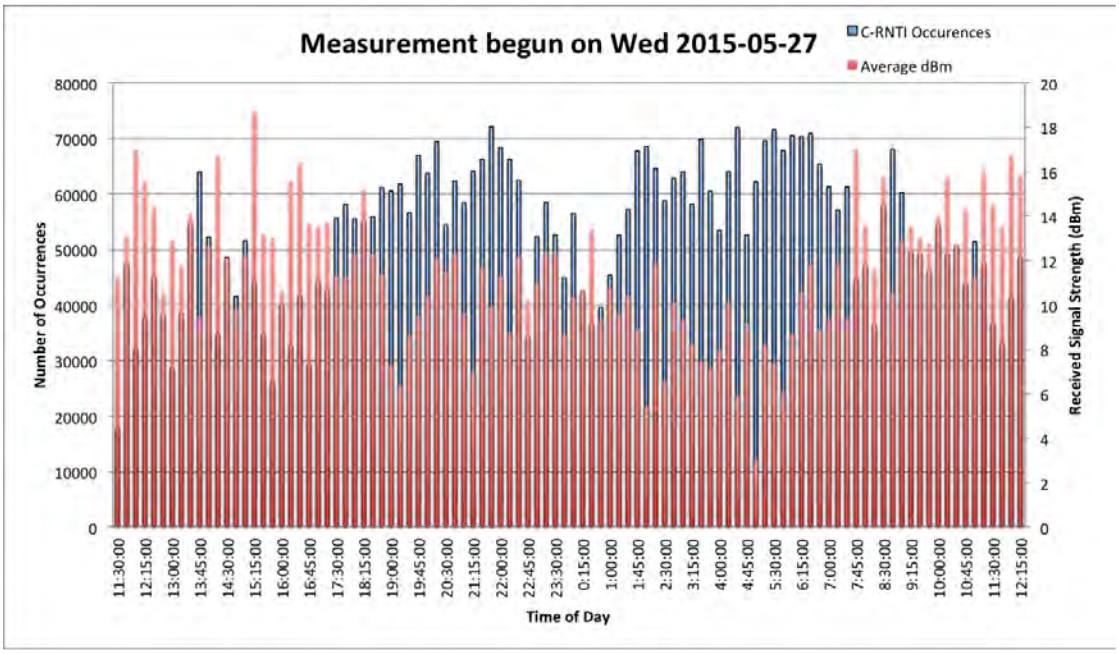

Figure 6.15: Plot showing the intensity of control signalling, measured on the roof of the institute building (environment 2).

6.5 Continuous Measurements in Environment 2

We have seen the rather unintuitive results of the measurements within the institute building's tower in the previous section. Even though it is unlikely that the reception of the signal is negatively influenced at this location of the building, some measurements on the outside of the roof have been accomplished in order to get results from an environment with even better reception. The results of these measurements can be seen in Figures 6.15 and 6.16. We see that the plots resemble the findings of the previous sections. However, when observing more thoroughly, the peaks experienced are less pronounced. Nevertheless, an increase of control signalling is still experienced rather during night times. This further supports the hypothesis of neighbouring cells that are being switched off and their *UE*s being transferred to the measured cell. Yet, the less pronounced difference between the peaks and the surrounding data could be interpreted as a result of an outside measurement as follows: If the theory concerning the shutdown of neighbouring antennas would be true, then the serving *eNodeB* would need to increase its transmission power in order to reach *UE*s in a greater area. This increase of transmission power could trigger high peaks indoors, as the rate of correctly transmitted and decoded data would be influenced positively, when the reception environment is not perfect. On the outside position on the roof, however, the reception of signals from the *eNodeB* would not be influenced equally: As the general signal reception quality is already higher, the difference caused by an increase of signal transmission power is less significant.
The plots in Figures 6.15 and 6.16 additionally contain bars indicating the average received signal strength of each 15 minutes time interval indicated in *dBm*. However, the *RSS* behaves in a

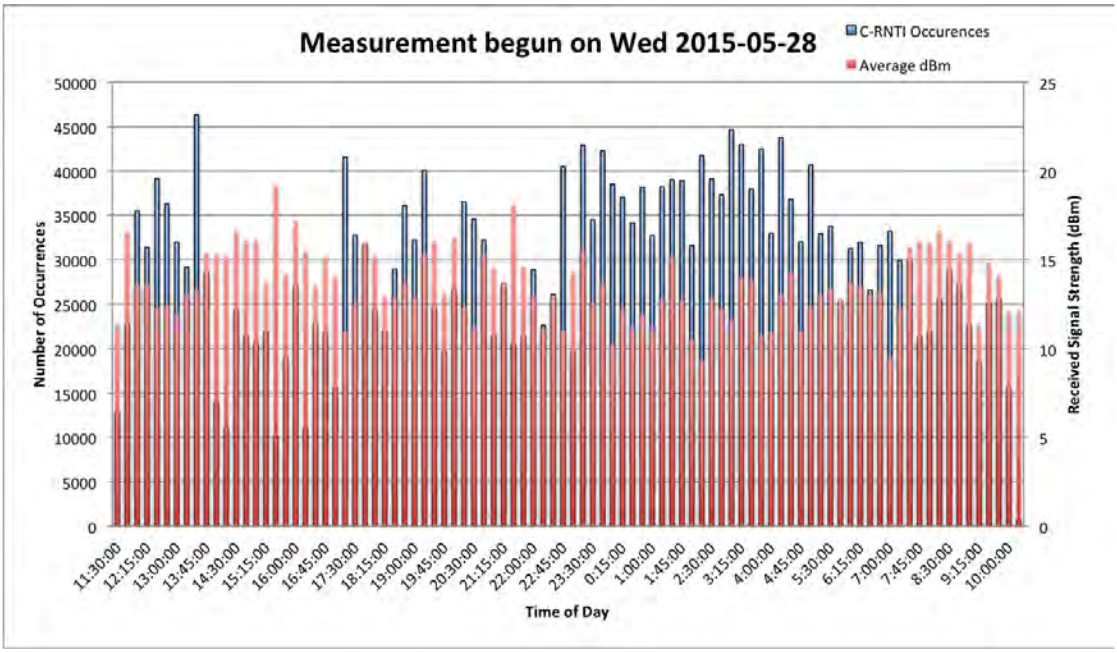

Figure 6.16: Plot showing the intensity of control signalling, measured on the roof of the institute building (environment 2).

rather inversely proportional manner towards the number of measured *RNTI* values. This does not comply with the hypothesis of the increased range of the cell during night times.
Another interpretation of the found control signalling was found: In Chapter 3, we cited that *the C-RNTI assigned to a user may change quite often if it transmits sporadically* [31]. Thus, it is thinkable that low traffic of *UE*s registered in a cell could yield higher control traffic than that of *UE*s with high user data traffic. On the one hand, this interpretation would make sense when considering the overall structure of the bar diagrams generated from the measurements, However, the massive overhead of control traffic experienced during night times does not seem reasonable considering the efficient system design of *LTE*. Yet, no other explanation could yet be found concerning the behaviour of control signalling and *RSS* measurements. Therefore, further research in this domain needs to be recommended as future work.

Chapter 7

Conclusions and Future Work

In the past chapters, we summarised the development of a means of signal analysis for *LTE*. First, a motivation for signal analysis was given and specific goals were defined that would need to be reached in order to implement such a signal analyser. Subsequently, we introduced the theoretical background needed for our specific approach of signal analysis in *LTE*. Additionally, an implementation environment was defined by means of selecting specific hardware and software components and extending them appropriately. Various locations with different characteristics were found for executing measurements and the results discussed.
In this section, a short conclusion shall be drawn from the findings of the previous chapters in Section 7.1. As the process of continuous research and development in an area is practically inexhaustible, some ideas and recommendations on future work are proposed in Section 7.2.

7.1 Conclusions

When resuming the motivation and procedure for development outlined in Chapter 1, we see that a progress in *LTE* signal analysis could be achieved: First of all, by defining the *Shopping Mall Application Scenario*, we gave a clear motivation for work in the domain of *LTE* signal analysis with the long-term goal of device localisation. Its definition forms a good starting point for developments in *LTE* signal analysis. Subsequently, we defined that device distinction would first need to be achieved and motivated its real-world application in the *Event Management and Security Scenario*. Furthermore, the examination and illustration of an iteration development cycle was laid out and followed during the implementation: First of all, a clear environment was defined by means of selecting specific and affordable hardware. Also, by building on already available software components, the advantages of them could be used. Moreover, the implementation of *RNTI* decoding in the *PUCCH* enabled device distinction, accomplishing a first and important step towards the implementation of localisation and user tracking.
These two measures—the definition of motivation scenarios and the orientation on a development iteration cycle—form an important contribution of this thesis towards further finding in *LTE* signal analysis. Furthermore, the usage of *passive* components, which are not part of the *LTE* network, and real-time decoding of the signals are important contributions of this work to the domain of *LTE* signal analysis.

Moreover, the definition of a theoretical workflow and its implementation by means of a continuously executed flowgraph represent a major contribution of this thesis to the available implementations of *LTE* signal analysis tools.
Additionally, an analysis of the provided implementation could be fulfilled by repeated measurements of downlink control information and corresponding transmissions and decodings of *C-RNTI*s. Even though these findings cannot yet be fully interpreted, they pose important questions that can guide further research and development cycles.
However, the provided implementation also illustrated the limitations currently encountered: On the one hand it is the limitation met when processing data, as overflows are encountered on the processing machines. On the other hand, the lack of accessibility to operator information and scheduling data poses a severe problem, as the findings of the measurements cannot be be answered with satisfactory certainty and no information on the implemented algorithms applied by the operators is available.
Therefore, we conclude that this work provides a motivation and outline of how work in the *LTE* signal analysis domain should be done. It provides contributions in defining a theoretical workflow and accomplishing its real-world live-processing implementation. This is done by means of a passive network capturing component. Finally, a thorough insight into the domain of signal analysis possibilities in *LTE* is achieved: This brought forth a series of measurements that initiated additional questions which give inspiration for further work. Nevertheless, it also shows that the very broad field of signal analysis offers a great variety of possibilities in the domain of signal analysis findings, which would necessitate further and very intensive investigations in the specific domain of interest. We finally state, that this work brings forth a platform, which can be used as a fundament and arbitrarily extended for further signal analysis aims.

7.2 Future Work

We have already mentioned that vast possibilities of additional work in the domain of *LTE* signal analysis could be approached. An overview of some concrete examples of what could further be done is now given.
We define three categories of future work. The first category aims at conducting further measurements and is presented in Subsection 7.2.1. A second category is defined for realising an *eNodeB* emulation for further measurements. It is discussed in Subsection 7.2.2. The third category, finally, consists of an implementation in the domain of uplink signal analysis and is discussed in Subsection 7.2.3

7.2.1 Additional Measurements

A straight-forward approach for future work is to execute additional measurements. The aim of this work could be to find further results leading to more concrete interpretation possibilities. There are various parameters that could be changed in relation to the measurements presented in this thesis. First, the locations could be changed. Measurements could be undertaken in other urban areas or, on the contrary, in the countryside. These findings could be used to determine whether similar patterns are found or certain cells are even switched off completely at certain

times.
Additionally, the measurements could be extended to further operators. As different operators may choose different parameters, scheduling mechanisms or power regulations, the flow of control messaging to the *UE*s may look different.
The choice of investigated parameters can be arbitrarily changed within the scope of available transmitted parameters. This can be done independently of measurement locations or operators chosen and could also lead to further findings concerning a more concrete interpretation of downlink control messages.

7.2.2 *eNodeB* Emulation

One of the major drawbacks during the analysis has been that we do not know what decisions the network operator takes when serving an *LTE* network. Hence, the analysis of a network in which we can determine the parameters chosen or the type and quantity of traffic would be helpful. This would impose the emulation of both *eNodeB*s and *UE*s by means of *SDR*. By choice, also real *UE*s could be connected to this network. Furthermore, a passive analysis node would have to listen to the downlink traffic in this network. This would provide the possibility of setting the measurement results into relation with triggered cell traffic and control messages. Hence, it could provide the possibility for emulating similar traffic scenarios as the ones found in real-world cells. Such findings would provide a more founded basis for signal analysis interpretations.

7.2.3 Implementation of Uplink Analysis

Finally, as we have defined that the *Shopping Mall Application Scenario* motivates *UE* tracking and localisation, implementation approaches into this direction could be tackled. However, signalling in the uplink is generally more complex, as it presumes a preliminary and continuous synchronisation on the downlink of a cell in parallel to passive uplink signal reception.
Furthermore, the analysis of downlink signalling relies on the adequate set-up of signalling procedures and *CRC* generation by means of the *C-RNTI*. Such procedures would first have to be determined for the uplink. Even though certain publications [32] claim to have achieved *C-RNTI* detection in the uplink, no implementation of neither off-line nor on-line processing of uplink signals has been found. Furthermore, no procedures suitable for such analysis that would be similar to the downlink signal analysis have been found in the uplink procedures up to this point.

Appendices

Appendix A

Downlink Measurement Software Installation and Execution

This appendix lists the commands necessary in order to install and run the *LTE Signal Analyser* software. All of the sources necessary for installation and execution are located in the *LTE_Analysis* folder on the DVD containing the thesis software.
The upcoming sections provide the details for the installation and execution. Section A.1 lists the software's requirements, such as software dependencies and recommended platforms. Subsequently, Section A.2 lists the installation commands and Section A.3 shows how the execution is accomplished.

A.1 Software Requirements

The software provided with this thesis relies on the *GNURadio* framework with all its components and dependencies. Specifically, *GNURadio* version 3.7.6.1 should be installed from the project's website [52]. Furthermore, the driver for an appropriate *SDR* device needs to be set up prior to the installation. It is generally recommended to install the software on a *Linux* platform, such as a recent version of *Ubuntu*.

A.2 Installation from Source

The software provided with this thesis is contained in a folder labelled *LTE_Analysis*, which has to be copied to the local file system. Subsequently, the following commands should be executed from this folder:

```
$ mkdir build
$ cd build
$ cmake .. && make
$ sudo make install && sudo ldconfig
```

A.3 Execution

In order to execute the software, first of all the execution engine has to be started as follows:

```
$ LTE_fdd_dl_scan
```

This invokes the engine to be started and wait for commands that trigger a specific execution. The execution can be controlled from a *telnet* interface, which needs to be invoked as follows in a separate terminal window (the port may need to be adjusted to the appropriate value shown after executing the foregoing command):

```
$ telnet localhost 30000
```

This opens a telnet session to a text interface for interaction with the *LTE Signal Analyser*. A variety of commands can be invoked. They are listed and described by invoking the *help* command.

Appendix B

Measurement Plotting

This appendix contains the scripts used in order to prepare the measured data for the plotting of graphs. As two slightly different scripts were used for the evaluation, both of them are listed in Sections B.2 and B.3. As both of the scripts take pure csv files as input, the output of the measurements first needs to be adjusted. For this purpose the bottom and top part of the measurement output—containing basic information on a measurement and statistics on the number of received different *RNTI* types—have to be removed. The commands for this purpose and the commands used to invoke the scripts are described in Section B.1.

B.1 Preparation of the Measured Data and Script Execution

We assume that a measurement file has been created by the *LTE Signal Analyser*. It has contents similar to Listing B.1.

Listing B.1: An example of measured data.

```
RNTI,time,type,
65535,2015-05-06 17:45:59,SI-RNTI,
47400,2015-05-06 17:45:59,C-RNTI,
...
53638,2015-05-06 18:27:45,C-RNTI,
65535,2015-05-06 18:27:45,SI-RNTI,
Number of received C-RNTIs: ,70512,
Number of received RA-RNTIs: ,84,
Number of received RESV-RNTIs: ,0,
Number of received M-RNTIs: ,0,
Number of received P-RNTIs: ,817,
Number of received SI-RNTIs: ,2662,
Started at: ,2015-05-06 17:45:56,
Finished at: ,2015-05-06 18:27:45,
Measurement duration: ,2509,
Number of measurements: 74076
```

Hence, the first line and the bottom ten lines of a measurement file need to be removed before it can be processed by any of the two scripts. For a file of the name *measurement.csv*, this is done by means of the following commands:

```
$ head -n -10 measurement.csv > measuredData.csv
$ tail -n 10 measurement.csv > testerProperties.csv
$ tail -n +2 measuredData.csv > measuredData1.csv
```

The resulting file called *measuredData1.csv* may now be processed commands depicted in the following part. The execution of the second script is applied adequately. After the commands, the aggregation of the measurements is listed in steps of $15min$ in the file *stats.csv*.

```
$ ./MeasurementPlotter measuredData1.csv > stats.csv
```

B.2 Plotter Script Version 1

Listing B.2: Code of script 1.

```
#!/bin/bash

echo "sourcing .bashrc"
source ~/.bashrc
echo "done"
IFS=","

#Some global vars
windowsSize_min=15
windowsSize_sec=$(( $windowsSize_min*60 ))
baseTime=0
crnti="C-RNTI"
numberOfOccurrences=0

#initialise Window Infimum using first line of measurements
firstline=$(head -n 1 $1)
read -a elements <<< "$firstline"
dt1=${elements[1]}
echo $dt1
t1=`date --date="$dt1" +%s`
baseTime=$(( $t1-$(( $t1%$windowsSize_sec ))))
echo "baseTime: $(date --date=@${baseTime} +"%Y-%m-%d %T")"

#go through all lines
while IFS="," read -ra ENTRY; do
  RNTI=${ENTRY[0]}
```

```
  dt2=${ENTRY[1]}
  RNTI_type=${ENTRY[2]}

  t2=`date --date="$dt2" +%s`
  tDiff=$(( $t2-$baseTime ))

  if [[ "$RNTI_type" != "$crnti" ]]; then
    #skip if entry is not a C-RNTI
    continue
  elif [ "$tDiff" -lt $(( 60*15 )) ]; then
    #if still in window add occurrence
    (( numberOfOccurrences+=1 ))
  else
    #otherwise print result of current window and go to next
       ↪ window
    formattedBaseTime=$(date --date=@${baseTime} +"%Y-%m-%d %T"
       ↪ )
    echo "$formattedBaseTime,$numberOfOccurrences"
    baseTime=$(( $baseTime+60*15 ))
    numberOfOccurrences=0
    (( numberOfOccurrences+=1 ))
  fi
done < $1
formattedBaseTime=$(date --date=@${baseTime} +"%Y-%m-%d %T")
echo "$formattedBaseTime,$numberOfOccurrences"
```

B.3 Plotter Script Version 2

Listing B.3: Code of script 2.

```
#!/bin/bash

echo "sourcing .bashrc"
source ~/.bashrc
echo "done"
IFS=","

#Some global vars
windowsSize_min=15
windowsSize_sec=$(( $windowsSize_min*60 ))
baseTime=0
crnti="C-RNTI"
```

```
numberOfOccurrences=0
amp_sum=0
watt_sum=0
dbm_sum=0

#initialise Window Infimum using first line of measurements
firstline=$(head -n 1 $1)
read -a elements <<< "$firstline"
dt1=${elements[1]}
echo $dt1
baseTime=$(( $dt1-$(( $dt1%$windowsSize_sec ))))
echo "baseTime: $(date --date=@${dt1} +"%Y-%m-%d %T")"

#go through all lines
while IFS="," read -ra ENTRY; do
  RNTI=${ENTRY[0]}
  dt2=${ENTRY[1]}
  RNTI_type=${ENTRY[2]}
  amplitude=${ENTRY[3]}
  watt=${ENTRY[4]}
  decibel=${ENTRY[5]}

  tDiff=$(( $dt2-$baseTime ))

  if [[ "$RNTI_type" != "$crnti" ]]; then
    #skip if entry is not a C-RNTI
    continue
  elif [ "$tDiff" -lt $(( 60*15 )) ]; then
    #if still in window add occurrence
    (( numberOfOccurrences+=1 ))
    amp_sum=`echo $amp_sum+$amplitude | bc -l`
    watt_sum=`echo $watt_sum+$watt | bc -l`
    dbm_sum=`echo $dbm_sum+$decibel | bc -l`
  else
    #otherwise print result of current window and go to next
       ↪ window
    formattedBaseTime=$(date --date=@${baseTime} +"%Y-%m-%d %T"
       ↪ )
    amp_avg=`echo $amp_sum/$numberOfOccurrences | bc -l`
    watt_avg=`echo $watt_sum/$numberOfOccurrences | bc -l`
    dbm_avg=`echo $dbm_sum/$numberOfOccurrences | bc -l`
    echo "$formattedBaseTime,$numberOfOccurrences,$amp_avg,
       ↪ $watt_avg,$dbm_avg"
```

```
    baseTime=$(( $baseTime+60*15 ))
    numberOfOccurrences=0
    amp_sum=0
    watt_sum=0
    dbm_sum=0
    (( numberOfOccurrences+=1 ))
    amp_sum=`echo $amp_sum+$amplitude | bc -l`
    watt_sum=`echo $watt_sum+$watt | bc -l`
    dbm_sum=`echo $dbm_sum+$decibel | bc -l`
  fi
done < $1
formattedBaseTime=$(date --date=@${baseTime} +"%Y-%m-%d %T")
echo "$formattedBaseTime,$numberOfOccurrences,$amp_avg,
   ↪ $watt_avg,$dbm_avg"
```

Bibliography

[1] (2015) Ihre Netzabdeckung auf dem mobilen Breitbandnetz von Sunrise. [Online]. Available: http://www1.sunrise.ch/Netzabdeckung-cb7abAqFI.yXYAAAFBesF2SHLu-Sunrise-Residential-Site-WFS-de_CH-CHF.html

[2] Salt. Network coverage. [Online]. Available: https://www.salt.ch/en/coverage/

[3] Swisscom. Willkommen auf dem schnellsten Netz. [Online]. Available: https://www.swisscom.ch/de/privatkunden/mobile/mobilnetz/4g-lte.html

[4] A. Ghosh and R. Ratasuk, *Essentials of LTE and LTE-A*, ser. Cambridge Wireless Essentials Series. Cambridge University Press, 2011.

[5] E. Dahlman, S. Parkvall, and J. Sköld, *4G: LTE/LTE-Advanced for Mobile Broadband*, second edition ed. Elsevier, 2014.

[6] T.-T. Tran, Y. Shin, and O.-S. Shin, "Overview of enabling technologies for 3gpp lte-advanced," *EURASIP Journal on Wireless Communications and Networking*, vol. February, no. 54, February 2012. [Online]. Available: http://jwcn.eurasipjournals.com/content/2012/1/54

[7] The MathWorks, Inc., "Synchronization signals (pss and sss)." [Online]. Available: http://ch.mathworks.com/help/lte/ug/synchronization-signals-pss-and-sss.html

[8] Ettus Research. Usrp n210. [Online]. Available: http://www.ettus.com/product/details/UN210-KIT

[9] ——. Usrp b210 (board only). [Online]. Available: http://www.ettus.com/product/details/UB210-KIT

[10] Bundesamt für Kommunikation BAKOM, "Faktenblatt 3GPP-LTE. Die Luftschnittstelle "Long Term Evolution"," *Eidgenössisches Departement für Umwelt, Verkehr, Energie und Kommunikation UVEK*, 2011. [Online]. Available: http://www.bakom.admin.ch/themen/technologie/01397/03794/index.html?lang=en&download=NHzLpZeg7t,lnp6I0NTU042l2Z6ln1ad1IZn4Z2qZpnO2Yuq2Z6gpJCDeoN6fWym162epYbg2c_JjKbNoKSn6A--

[11] Technical Specification Group Radio Access Network, "3gpp ts 36.101 evolved universal terrestrial radio access (e-utra); user equipment (ue) radio transmission and reception," 2014. [Online]. Available: http://www.3gpp.org/dynareport/36101.htm

[12] ritwikdubey, "Lte frame structure (continued)," *3gpptutorial*. [Online]. Available: https://3gpptutorial.wordpress.com/

[13] RedOrbit, "The history of mobile phone technology," *RedOrbit*, 2014. [Online]. Available: http://www.redorbit.com/reference/the-history-of-mobile-phone-technology/

[14] Wimax forum mobile 4g. WiMAX Forum. [Online]. Available: http://www.wimaxforum.org/mobile-4g

[15] 3GPP, "Overview of 3gpp release 8 v0.3.3 (2014-09)," 2014. [Online]. Available: http://www.3gpp.org/ftp/Information/WORK_PLAN/Description_Releases/Rel-08_description_20140924.zip

[16] ——, "Overview of 3gpp release 10 v0.2.1 (2014-06)," 2014. [Online]. Available: http://www.3gpp.org/ftp/Information/WORK_PLAN/Description_Releases/Rel-10_description_20140630.zip

[17] J. Brühl, "5G-Mobilfunktechnik. Revolution der Geschwindigkeit," *Süddeutsche Zeitung Digitale Medien GmbH / Süddeutsche Zeitung GmbH*, June 2015. [Online]. Available: http://www.sueddeutsche.de/digital/mobilfunktechnik-eine-frage-der-kommunikation-1.2512935

[18] C. Janssen, "Internet of things (iot)," *techopedia*. [Online]. Available: http://www.techopedia.com/definition/28247/internet-of-things-iot

[19] Ettus Research. Usrp hardware driver and usrp manual. [Online]. Available: http://files.ettus.com/manual/index.html

[20] "Transitioning from wimax to lte: How one malaysian operator is making it work," *FireceWirelessTech*, June 2011. [Online]. Available: http://www.fiercewireless.com/tech/special-reports/transitioning-wimax-lte-how-one-malaysian-operator-making-it-work

[21] IEEE. Ieee 802.16: Broadband wireless metropolitan area networks (mans). IEEE. [Online]. Available: https://standards.ieee.org/about/get/802/802.16.html

[22] I. Aldmour, "Lte and wimax: Comparison and future perspective," *Communications and Network*, no. 5, pp. 360–368, 2013.

[23] S. Yi, S. Chun, Y. Lee, S. Park, and S. Jung, *Radio Protocols for LTE and LTE-Advanced*. Wiley, 2012.

[24] J. Wannstrom, "Lte-advanced," *3GPP*, 2013. [Online]. Available: http://www.3gpp.org/technologies/keywords-acronyms/97-lte-advanced

[25] Rohde & Schwarz, "Lte / lte-advanced fundamentals," *Rohde & Schwarz*, 2015. [Online]. Available: http://www.rohde-schwarz.com/en/technologies/cellular/lte/lte-technology/lte_information_52292.html

[26] D. Fritz, "Swisscom startet LTE-Netzwerk," *netzwoche*, November 2012. [Online]. Available: http://www.netzwoche.ch/de-CH/News/2012/11/28/Swisscom-startet-LTE-Netz.aspx

[27] H. Steier, "Lte advanced. swisscom schneller als sunrise," *NZZ*, January 2014. [Online]. Available: http://www.nzz.ch/digital/sunrise-lte-advanced-start-schweiz-1.18231473

[28] Sunrise. Das Sunrise Mobilfunknetz wächst. [Online]. Available: http://www1.sunrise.ch/Mobilfunknetz-cbjonAqFI.CboAAAFB_k4ib3Mk-Sunrise-Residential-Site-WFS-de_CH-CHF.html

[29] M. Milatz, "Profit durch analyse," *Handelszeitung*, November 2014. [Online]. Available: http://www.handelszeitung.ch/management/profit-durch-analyse-689756

[30] EURECOM. (2015) Open air interface. EURECOM Campus SophiaTech. 450 Route des Chappes, CS 50193 - 06904 Biot Sophia Antipolis cedex, FRANCE. [Online]. Available: http://www.openairinterface.org/

[31] S. Kumar, E. Hamed, D. Katabi, and L. E. Li, "Lte radio analytics made easy and accessible," *SIGCOMM'14*, 2014. [Online]. Available: http://www.mit.edu/~swarun/papers/lteye-sigcomm2014.pdf

[32] S. Kumar, E. Hamed, L. E. Li, and D. Katabi. (2014) Lteye: Code release. wireless@mit. [Online]. Available: http://www.mit.edu/~swarun/lteyecode.html

[33] The MathWorks, Inc. Matlab. [Online]. Available: http://ch.mathworks.com/products/matlab/

[34] B. Wojtowicz. (2015) Openlte. [Online]. Available: http://sourceforge.net/projects/openlte/

[35] OsmocomSDR. Osmosdr. [Online]. Available: http://sdr.osmocom.org/trac/

[36] ——. Rtl-sdr. [Online]. Available: http://sdr.osmocom.org/trac/wiki/rtl-sdr

[37] M. Ossmann. Hackrf one. Great Scott Gadgets. [Online]. Available: https://greatscottgadgets.com/hackrf/

[38] Ettus research - the leader in software defined radio. Ettus Research. [Online]. Available: http://www.ettus.com/

[39] EURECOM. Openairinterface twiki. [Online]. Available: https://twiki.eurecom.fr/twiki/bin/view/OpenAirInterface

[40] "World's most complete open source lte base station (enb) software supports usrp software defined radio," *EURECOM Service communication et presse*, 2014. [Online]. Available: http://www.openairinterface.org/openairfiles/openAirInterface_nov_2014.pdf

[41] GNU. Gnu octave. [Online]. Available: http://www.gnu.org/software/octave/

[42] 3GPP. 3gpp specification series: 36series. [Online]. Available: http://www.3gpp.org/dynareport/36-series.htm

[43] J. T. J. Penttinen, Ed., *The LTE/SAE Deployment Handbook*. John Wiley & Sons Ltd., 2012.

[44] G. Fairhurst and L. Wood, "Advice to link designers on link automatic repeat request (arq)," Network Working Group, https://tools.ietf.org/html/rfc3366, Tech. Rep. RFC 3366, 2002.

[45] E. Soljanin, R. Liu, and P. Spasojevic, "Hybrid arq with random transmission assignments," *Advances in network information theory*, 2004.

[46] Technical Specification Group Radio Access Network, "3gpp ts 36.321 medium access control (mac) protocol specification," 2015. [Online]. Available: http://www.3gpp.org/dynareport/36321.htm

[47] F. Xiong, *Digital Modulation Techniques*. Artech House Inc, 2000. [Online]. Available: http://www.artechhouse.com/uploads/public/documents/chapters/Xiong_863_CH04.pdf

[48] H.-J. Zepernick and A. Finger, *Pseudo Random Signal Processing: Theory and Application*. Wiley, 2013.

[49] Technical Specification Group Radio Access Network, "3gpp ts 36.211 physical channels and modulation," 2015. [Online]. Available: http://www.3gpp.org/dynareport/36211.htm

[50] ——. (2014) 3gpp ts 36.321 physical channels and modulation. [Online]. Available: http://www.3gpp.org/dynareport/36211.htm

[51] Ettus Research. Usrp b200 and b210 faq. [Online]. Available: http://www.ettus.com/kb/detail/usrp-b200-and-b210-faq

[52] J.-P. Lang. Gnuradio. [Online]. Available: http://gnuradio.org/redmine/projects/gnuradio/wiki

[53] Canonical, Ltd. Ubuntu. [Online]. Available: http://www.ubuntu.com/

Erklärung

gemäss Art. 28 Abs. 2 RSL 05

Name/Vorname:

Matrikelnummer:

Studiengang:

Bachelor ☐ Master ☐ Dissertation ☐

Titel der Arbeit:

LeiterIn der Arbeit:

Ich erkläre hiermit, dass ich diese Arbeit selbständig verfasst und keine anderen als die angegebenen Quellen benutzt habe. Alle Stellen, die wörtlich oder sinngemäss aus Quellen entnommen wurden, habe ich als solche gekennzeichnet. Mir ist bekannt, dass andernfalls der Senat gemäss Artikel 36 Absatz 1 Buchstabe r des Gesetzes vom 5. September 1996 über die Universität zum Entzug des auf Grund dieser Arbeit verliehenen Titels berechtigt ist. Ich gewähre hiermit Einsicht in diese Arbeit.

Ort/Datum

Unterschrift