Technische Universität Dresden

Real-Time Waveform Prototyping

Dipl.-Ing.

Martin Danneberg

der Fakultät Elektrotechnik und Informationstechnik der Technischen Universität Dresden

zur Erlangung des akademischen Grades

Doktoringenieur

(Dr.-Ing.)

genehmigte Dissertation

Vorsitzender: Prof. Dr.-Ing. habil. Ercan Altinsoy
Gutachter: Prof. Dr.-Ing. Dr. h.c. Gerhard P. Fettweis
Gutachter: Prof. Dr.-Ing. Berthold Lankl

Tag der Einreichung: 01.12.2020
Tag der Verteidigung: 13.04.2021

Abstract

The demand to achieve higher data rates for the Enhanced Mobile Broadband scenario and novel fifth generation use cases like Ultra-Reliable Low-Latency and Massive Machine-type Communications drive researchers and engineers to consider new concepts and technologies for future wireless communication systems. The goal is to identify promising candidate technologies among a vast number of new ideas and to decide, which are suitable for implementation in future products. However, the challenges to achieve those demands are beyond the capabilities a single processing layer in a wireless network can offer. Therefore, several research domains have to collaboratively exploit research ideas.

This thesis presents a platform to provide a base for future applied research on wireless networks. Firstly, by giving an overview of state-of-the-art prototypes and testbed solutions. Secondly by introducing a flexible, yet real-time physical layer signal processor running on a software defined radio platform. The processor enables reconfiguring important parameters of the physical layer during run-time in order to create a multitude of modern waveforms. Thirdly, by introducing a generic test infrastructure, which can be tailored to prototype diverse wireless technology and which is remotely accessible in order to invite new ideas by third parties. Using the test infrastructure, the performance of the flexible transceiver is evaluated regarding latency, achievable throughput and packet error rates.

Zusammenfassung

Mobile Netzwerke der fünften Generation zeichen sich aus durch vielfältigen Anforderungen und Einsatzszenarien. Drei unterschiedliche Anwendungsfälle sind hierbei besonders relevant: 1) Industrie-Applikationen fordern Echtzeitfunkübertragungen mit besonders niedrigen Ausfallraten. 2) Internet-of-things-Anwendungen erfordern die Anbindung einer Vielzahl von verteilten Sensoren. 3) Die Datenraten für Anwendung wie z.B. der Übermittlung von Videoinhalten sind massiv gestiegen.

Diese zum Teil gegensätzlichen Erwartungen veranlassen Forscher und Ingenieure dazu, neue Konzepte und Technologien für zukünftige drahtlose Kommunikationssysteme in Betracht zu ziehen. Ziel ist es, aus einer Vielzahl neuer Ideen vielversprechende Kandidatentechnologien zu identifizieren und zu entscheiden, welche für die Umsetzung in zukünftige Produkte geeignet sind. Die Herausforderungen, diese Anforderungen zu erreichen, liegen jedoch jenseits der Möglichkeiten, die eine einzelne Verarbeitungsschicht in einem drahtlosen Netzwerk bieten kann. Daher müssen mehrere Forschungsbereiche Forschungsideen gemeinsam nutzen. Diese Arbeit beschreibt daher eine Plattform als Basis für zukünftige experimentelle Erforschung von drahtlosen Netzwerken unter reellen Bedingungen. Es werden folgende drei Aspekte näher vorgestellt:

Zunächst erfolgt ein Überblick über moderne Prototypen und Testbed-Lösungen, die auf großes Interesse, Nachfrage, aber auch Förderungsmöglichkeiten stoßen. Allerdings ist der Entwicklungsaufwand nicht unerheblich und richtet sich stark nach den gewählten Eigenschaften der Plattform. Der Auswahlprozess ist jedoch aufgrund der Menge der verfügbaren Optionen und ihrer jeweiligen (versteckten) Implikationen komplex. Daher wird ein Leitfaden anhand verschiedener Beispiele vorgestellt, mit dem Ziel Erwartungen im Vergleich zu den für den Prototyp erforderlichen Aufwänden zu bewerten.

Zweitens wird ein flexibler, aber echtzeitfähiger Signalprozessor eingeführt, der auf einer software-programmierbaren Funkplattform läuft. Der Prozessor ermöglicht die Rekonfiguration wichtiger Parameter der physikalischen Schicht während der Laufzeit, um eine Vielzahl moderner Wellenformen zu erzeugen. Es werden vier Parametereinstellungen "LLC", "WiFi", "eMBB" und "IoT" vorgestellt, um die Anforderungen der verschiedenen drahtlosen Anwendungen widerzuspiegeln. Diese werden dann zur Evaluierung der die in dieser Arbeit vorgestellte Implementierung herangezogen.

Drittens wird durch die Einführung einer generischen Testinfrastruktur die Einbeziehung

externer Partner aus der Ferne ermöglicht. Das Testfeld kann hier für verschiedenste Experimente flexibel auf die Anforderungen drahtloser Technologien zugeschnitten werden. Mit Hilfe der Testinfrastruktur wird die Leistung des vorgestellten Transceivers hinsichtlich Latenz, erreichbarem Durchsatz und Paketfehlerraten bewertet. Die öffentliche Demonstration eines taktilen Internet-Prototypen, unter Verwendung von Roboterarmen in einer Mehrbenutzerumgebung, konnte erfolgreich durchgeführt und bei mehreren Gelegenheiten präsentiert werden.

Acknowledgements

This thesis would not have been possible without the help, support, and guidance of many people. I would like to thank Gerhard for his support and guidance but also for the very inspiring and positive working atmosphere at the Vodafone Chair! This is something special and further nurtured by all of the Vodafone chair team members. I am really thankful for my time at the Chair.

During my 12 years working as a student assistant and team member, I received much help from Kathrin and Sylvia - many thanks for everything. Project management is a though job which has been gracefully mastered by Eva - it was a pleasure to work with you, thanks. I also would like to thank Steffen - you are not forgotten!

I am very grateful to been able to work with various former colleagues of the GFDM group. I would like to thank Nicola for all the reviews and support during the years. It was a pleasure to work with you, Max - many thanks and I am really sorry for the last minute changes. Thank you, Luciano for all the inspirational discussions and the initial push to start the thesis. Another core member of the GFDM topic was Ivan - thank you for your support. I would also like to express my gratitude to Dan for all the help and guidance I received during her time as a group leader and afterwards. Building working demos would have not been possible without Paul - many thanks for that. And presenting them at events or during the course of the projects was assisted by Shahab - thank you for your support. Marwa and Tobias gave mental support over the years - thanks.

Although the research focus changed, quite a few GFDM-group colleagues continue their research at the university. Many thanks to Zhongju for all the efforts you spent building the demos, working on the FPGA development and building the testbed. And thank you Daniel for finalizing and maintaining all those works. I had a lot of fun doing experiments together with Roberto - many thanks. The mastermind behind all the waveforms is without doubt Ahmad - salute for your work and thank you, also for letting me stay in your office room.

Building a testbed and lab is not possible without active IT support. Many thanks to Zhitao and Raffael, Rüdiger and Sebastian from the side. Thanks to Richard for his energetic assistance. I had many fruitful discussions with Nick about testbed, demos and IT - thank you very much. Further, I believe the art of PhD-Hat-Design soared to new heights. I am also grateful to all the nice experiences I had and people I met during

the course of the various research projects CREW, eWINE, ORCA and the NI Lead User Program, which also pushed this thesis forward. In addition, I would like to thank Professor Berthold Lankl for accepting being the second reviewer of my thesis and the support in preparation of the defense.

Finally, I would like to thank my family and friends for all the (mental) support given over many years. And without you, Nadine, none of this would have been possible - I cannot express enough my gratitude! Thank you to my daughter Tamina for all the smiles I received during your first year.

Contents

List of Figures

List of Tables

Abbreviations

5G	fifth generation
ACK	acknowledgment
ADC	analog-to-digital converter
AFW	Application Framework
AGC	automatic gain control
AGV	automated guided vehicle
AP	access point
ASIC	application-specific integrated circuit
AWGN	additive white Gaussian noise
BNetzA	Bundesnetzagentur
BS	base station
CFO	carrier frequency offset
CIR	channel impulse response
CoMP	coordinated multi point
COTS	commercial off-the-shelf
CP	cyclic prefix
CPU	central processing unit
CR	cognitive radio
CRC	cyclic redundancy check
CS	cyclic suffix
CSMA	carrier sense multiple access
DAC	digital-to-analog converter
DARPA	Defense Advanced Research Projects Agency
DC	direct current
DFT	discrete Fourier transform
DL	downlink
DLL	Dynamic Link Library
DMA	direct memory access
DSP	digital signal processor

DSSS	direct-sequence spread spectrum
DVB-T	terrestrial digital video broadcasting
EIRP	equivalent isotropically radiated power
eMBB	Enhanced Mobile Broadband
ESA	European Space Agency
FBMC	filter bank multicarrier
FDD	frequency-division duplexing
FFT	fast Fourier transform
FIFO	first in, first out
FIR	finite impulse response
FMT	filtered multi tone
FPGA	field programmable gate array
FT	fourier transform
GFDM	generalized frequency division multiplexing
GPP	general purpose processor
GPS	global positioning system
GPU	graphics processing unit
GUI	graphical user interface
HARQ	hybrid automatic repeat request
ICI	inter-carrier interference
ID	identification
IDE	integrated development environment
IDFT	inverse discrete Fourier transform
IFFT	inverse fast Fourier transform
IoT	Internet of Things
IP	Internet protocol
IPC	interprocess communication
IQ	in-phase and quadrature
ISM	industrial, scientific and medical band
LBT	listen before talking
LLC	Low-Latency Communication
LNA	low noise amplifier
LOS	line-of-sight
LTE	Long Term Evolution
LUT	look-up table
MAC	medium access control
MC	multicarrier

MCS	modulation coding scheme
MIMO	multiple input multiple output
mMTC	Massive Machine Type Communications
mmWave	Millimeter-Wave
MSE	mean squared error
MTU	maximum transmission unit
NASA	National Aeronautics and Space Administration
NI	National Instruments
OFDM	orthogonal frequency division multiplexing
OMF	Orbit Management Framework
OMF	Orbit Management Framework
OOB	out-of-band
OS	operating system
OWL	Online Wireless Lab
PA	power amplifier
PAPR	peak-to-average power ratio
PDCCH	Physical Downlink Control Channel
PER	packet error rate
PHY	physical layer
PLCP	physical layer convergence procedure
PLL	phase-locked loop
PN	pseudo noise
PPS	pulse-per-second signal
PRB	physical resource block
PRB	physical resource block
PTP	precision time protocol
PXI	PCI eXtensions for Instrumentation
QAM	quadrature amplitude modulation
QPSK	quadrature phase shift keying
RAM	random access memory
RAN	radio access network
RF	radio frequency
RFNoC	RF Network on Chip
RTT	round trip time
RX	receiver
SC-FDE	single-carrier with frequency domain equalization
SC-FDMA	single-carrier frequency division multiple access

SC	single-carrier
SDN	software-defined networking
SDR	Software-defined radio
SISO	single-input and single-output
SNR	signal-to-noise ratio
TCP	transmission control protocol
TDD	time-division duplexing
TDMA	time division multiple access
TR-STC	time-reverse space time code
TRL	technology readiness level
TVWS	TV white space
TX	transmitter
UDP	user datagram protocol
UE	user equipment
UFMC	universally filtered multicarrier
UL	uplink
URLLC	Ultra-Reliable and Low-Latency Communication
UT	user-terminal
UWB	ultra-wide band
VI	Virtual Instrument
VLAN	virtual local area network
VM	virtual machine
WLAN	wireless local area network
ZF	zero-forcing

Notations

$\mathbb{C}$	complex number field
$\mathbb{R}$	real number field
$\mathbb{N}$	integer set
$\boldsymbol{A}$	uppercase and bold face for matrix
$\boldsymbol{a}$	lowercase and bold face for vector
a, A	scalar value
$\mathcal{A}$	calligraphic letter for set
$[\boldsymbol{A}]_{(n,m)}$	(n,m)-th element in a matrix
$[\boldsymbol{a}]_{(n)}$	n-th element in a matrix
$[\boldsymbol{A}]_{(\mathcal{N},\mathcal{M})}$	sub-matrix of size $\lvert\mathcal{N}\rvert \times \lvert\mathcal{M}\rvert$
$\lvert\mathcal{N}\rvert$	number of elements in the set $\mathcal{N}$
$\boldsymbol{F}_N$	N-DFT matrix, $[\boldsymbol{F}_N]_{(k,m)} = e^{-2\pi\frac{nk}{N}}$
$\tilde{\boldsymbol{A}} = \boldsymbol{F}_N\boldsymbol{A}$	DFT of columns
$\mathrm{vec}\{\boldsymbol{A}\}$	vectorization operation
$\mathrm{unvec}_{N\times M}\{\boldsymbol{a}\}$	reverse of vectorization
$\mathrm{diag}\{\boldsymbol{A}\}$	diagonal elements of matrix $\boldsymbol{A}$
$\boldsymbol{\Lambda}^{(\boldsymbol{a})}$	diagonal matrix of vector $\boldsymbol{a}$
$\mathrm{Re}\{\cdot\}$	real part
$\odot$	element-wise product
$\oslash$	element-wise division
$\circledast$	circular convolution
$\otimes$	Kronecker product
$< \cdot >_N$	modulo-N

K, M	number of subcarriers, subsymbols
$\mathcal{K}_{\text{on}}, \mathcal{M}_{\text{on}}$	set of active subcarriers, subsymbols
T_{sub}	subsymbol duration
T	block duration
$T_{\text{cp}}, T_{\text{cs}}$	cyclic prefix (CP), cyclic suffix (CS) duration
T_{sub}	subsymbol duration
$N = MK$	number of symbols per block
$\mathcal{N}_{\text{on}}$	$= \{n = k + mK,\ (k, m) \in \mathcal{K}_{\text{on}} \times \mathcal{M}_{\text{on}}\}$
$\boldsymbol{g}$	prototype pulse shape
$\boldsymbol{A}$	modulation matrix
$\boldsymbol{B}$	demodulation matrix
N_{b}	number of blocks per frame
$N_{\text{cp}}, N_{\text{cs}}$	number of samples per CP, CS
N_{t}	number of samples per transmitted block
N_{s}	number of block spacing samples
N_{o}	number of overlapping samples

Chapter 1

Introduction

Wireless networks of the future face very different service requirements. Besides the long-term goals of increasing the data throughput as in Enhanced Mobile Broadband (eMBB) scenarios, novel applications emerged. Those novel use cases such as Ultra-Reliable and Low-Latency Communication (URLLC) and Massive Machine Type Communications (mMTC) evoke diverse requirements on the network themselves, because they are geared towards machine type communications for the Internet of Things (IoT). Messages are intended to control machines and not to be read by humans. In contrast, the traditional focus of eMBB networks is to provide Internet access.

Novel ideas or concepts must prove their applicability through reproducible, experimental validation. Simulated and idealized performance results might otherwise be misleading when taken into account within other research areas. For instance, a robotic expert has to build upon mMTC networks to interconnect robotic applications. The network might provide a very good packet error rate (PER) on average, but if once in a while 20 consecutive packets are lost, robotic platforms such as the *Franka Emika* arm would trigger an emergency break [1]. Another wireless system might have a much worse PER, but can prevent 20 consecutive failures. As a result one can state that we are at the advent of a new wireless era. New applications connecting IoT devices are becoming networked. The impact of new applications on the network and vice versa is not fully understood. A major step forward on addressing the open research challenges would be to provide a flexible SDR hardware/software platform, aiming to understand and analyze the cross coupling of the new set of applications on wireless networks, and to adapt these accordingly. Therefore, the goal of this work is to present a hardware and software framework to prototype modern wireless networks based on Software-defined radio (SDR).

Fig. 1.6 illustrates an approach of prototyping a wireless algorithm, thus outlining the chapter of this work. This introduction chapter covers the first three items ”Idea”, ”Use case” and ”Scenario”. The second chapter introduces the background of wireless communication systems and details the model behind the proposed transceiver structure. The third chapter discusses the individual sub-items in Fig. 1.6, based on state-of-the-art

platforms and testbeds. In the fourth chapter a field programmable gate array (FPGA) design following the model of the second chapter is suggested and evaluated. Since the environment has a significant influence on the performance of the base-band signal processor, the fifth chapter studies the radio frequency (RF) frontend and the testbed architecture. The sixth chapter examines the achieved performance considering an End-to-End wireless network application. The key lessons of this thesis are discussed in the seventh chapter. A summary of the presented work is given in the eighth chapter.

1.1 Wireless applications

Amongst many applications that depend on wireless networks, industrial applications can be considered as a major employer of 5G networks [2]. Consequently, they constitute most of the examples throughout this thesis. Fig. 1.1 illustrates different wireless applications in an industrial scenario. One driving factor for industry to implement wireless networks is to increase productivity while keeping maintenance costs in check. In addition, factories of the future strive to enable mass customization, where the products are being manufactured based on customer needs [3]. In this scenario, each product has individual properties, thus requiring a high flexibility of the production line.

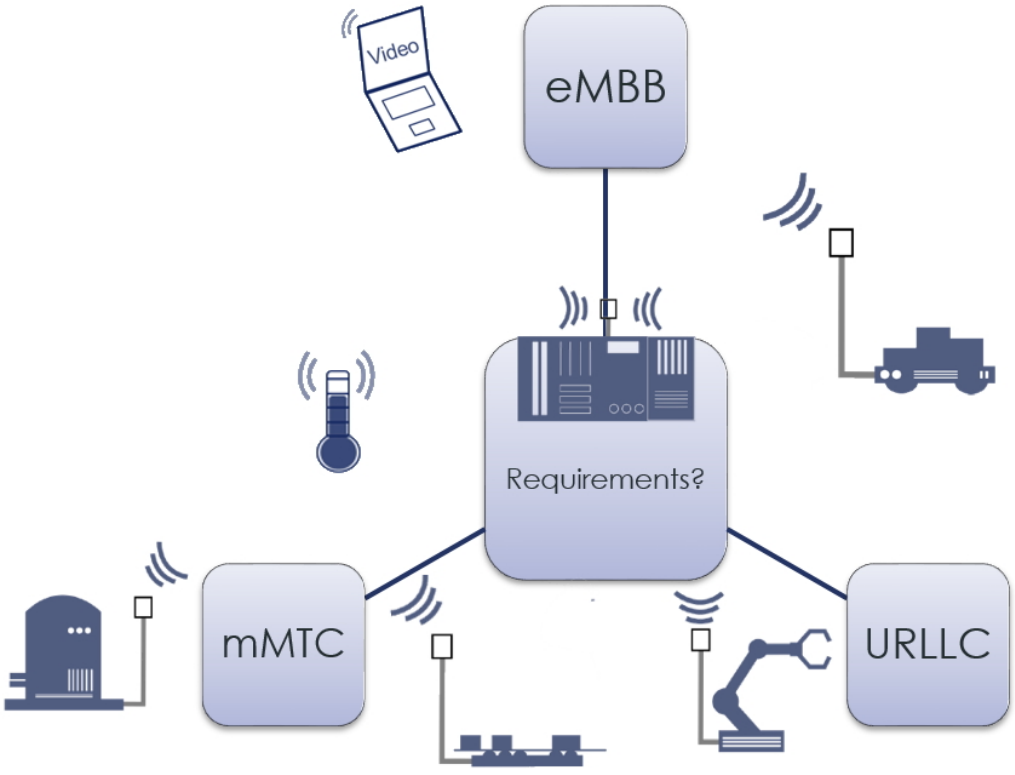


Figure 1.1: Different applications have different network requirements

Traditionally, field bus systems and cabled infrastructure take over the role of

interconnecting machines and robots, particularly in permanent installations. Notably, machines with rotating or moving elements or sensors have very strict specifications for the cables to withstand the operation conditions. Furthermore, the cables need to be monitored and constantly maintained in order to ensure a smooth operation without unforeseen malfunctioning. In case wireless networks replace those fixed permanent cable arrangements, industrial applications could be rearranged based on the production demands to achieve high flexibility, while reducing the maintenance costs of cables.

However, industrial machinery and sensors are very heterogeneous and diverse. Networked clients that need to be interconnected can have any manufacturing age and operating system and therefore follow any possible networking standard. Future installations can include private licensed networks and utilize mmWave bands. As a consequence, a wireless network needs to handle legacy and modern clients as well. They can be grouped into three types:

- ▷ Typical eMBB applications are Video streaming or file download. Here, optimizing the data rate is the dominating focus of the network planning. The latency of the data packets or the amount of devices are moderate requirements for the network.
- ▷ Wireless mMTC or Industrial Internet of Things (IIoT) networks are regarded as key enablers for the Factory 4.0 scenario [4]. Here, a multitude of clients, typically sensors, need to be stably included into the network, though with lower data rates.
- ▷ Tactile Internet [5] applications, such as augmented or virtual reality, require URLLC capabilities of wireless networks. For instance, in a real-time robot control application a respective command needs to be delivered with low jitter and not based on best effort, such as in traditional wireless networks. So, latency is important as well as data rate and stability.

1.2 Motivation

The demands outlined previously drive researchers and engineers to consider new concepts and technologies for future wireless communication systems. The goal is to identify promising technologies among a vast number of new ideas and to decide, which are suitable for implementation in future products. Fig. 1.2 gives a simplified overview of the process, which is highly iterative.

New ideas and concepts typically undergo extensive software simulations first, which allow initial predictions on the expected performance. State-of-the-art technologies and applications on the market act as filters for the ideas that are presented as input for the simulations. After the down-selection of the candidates, chosen aspects of the envisioned system can be implemented on a hardware-accelerated platform, e.g. SDR, in order to learn about real-time behavior and over-the-air performance with real RF components. Technologies that prove promising in this stage can be further evaluated in testbeds, where

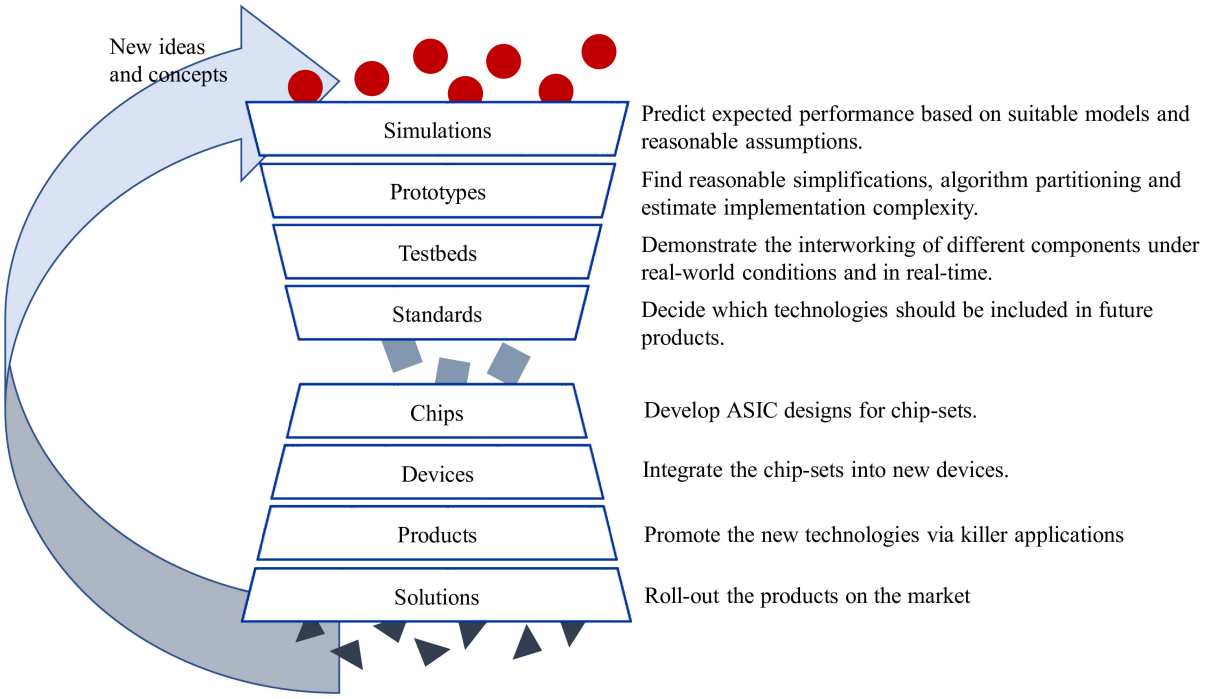


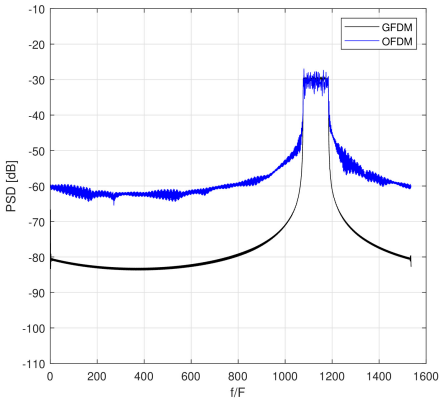
Figure 1.2: Simplified process of down-selection.

the focus shifts towards the inter-working of different technology building blocks and the realization of complete end-to-end applications. New concepts and technologies that have been proven to work in practice will ultimately find their way into new standards.

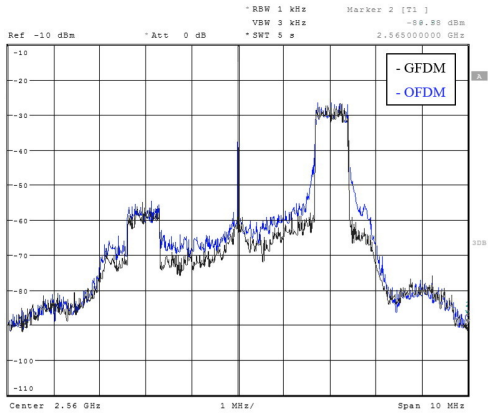
Before any technology can be successfully released, the respective chip-sets need to be developed. This is a complicated and expensive step. Therefore, the technology needs to be studied extensively and well understood. Afterwards, the chip-sets have to be integrated into devices or modules, to enable the usage for prototype applications. Finally, the industry will adopt the products in future solutions after trendsetting applications demonstrated the versatility of the new technology. According to [2], the estimated digitalization revenues for ICT players amount to $ 1,200 billion until 2026, where a large portion of approximately $ 234 billion is predicted for manufacturing use cases.

In general, most research makes assumptions to simplify the models and relies either on math derivation or on numerical simulation to evaluate the performance. For instance, in many cases the RF-frontend is considered linear to simplify the transmission model. However, highly-linear and power-inefficient RF frontends are only affordable for access points or base stations, not necessarily for the wireless client. Fig. 1.3 shows that there are severe differences between the simulations in Fig. 1.3a and the same signals transmitted in Fig. 1.3b by a commercially available LTE-chip-set used in *EURECOMs* ExpressMIMO2 SDR device. In theory, the difference is that the spurious radiations of newer transmission schemes, such as generalized frequency division multiplexing (GFDM), are significantly lower (20 dB) than state-of-the-art technology based on orthogonal frequency division multiplexing (OFDM) such as Long Term Evolution (LTE). However, the RF-frontend

in this case is designed to achieve a performance sufficient for OFDM. Thus, the GFDM concept cannot reveal its advantages.



(a) Simulated, amplitude scaled to fit measurement.



(b) Generated by a commercially available LTE chip-set.

Figure 1.3: Out-of-band emission of the OFDM and GFDM signals used in [6]

Besides the RF-frontend another common assumption in simulation is perfect synchronization, which has a very critical influence on the performance and is a challenging task in practice, because if the first stage of the receiver is already dropping the packet, no advanced receiving algorithm can recover it. Furthermore, simulations are usually carried out in floating point, whereas fixed-point is applied in hardware. Thus the numerical precision is limited.

On top of that, many simulations might consider individual imperfections, but seldom end-to-end aspects, due to the difficulties in modeling every aspect in details. In particular when several research areas depend on each other, simplifications might limit the practicability of the overall approach. For example, in a robotic scenario where several robots have to jointly interact to fulfill a common mission. Experts from the mechanical, communication as well as computer science fields need to work together.

Of course, (robotic) engineers could use commercial off-the-shelf (COTS) hardware and accept the offered performance. However, debugging COTS hardware can be challenging even for communication experts, since the manufacturer might not reveal inside information or provide open firmware. Further adapting COTS chip-sets for new concepts, e.g. for wireless local area network (WLAN) or LTE, is very difficult. Besides the missing knowledge about the firmware, another obstacle lies in the fact that those chip-sets are certified for given standards and would have to be re-certified.[1] In order to bypass such restrictions, this work presents a strategy for prototyping new wireless concepts based on SDR technology which is programmable and not limited to a certain firmware.

[1] Completely open communication devices are not available due to legal issues. Europe is currently discussing a potential way forward [7].

1.3 Software-Defined Radio

Targeting military applications first, Mitola introduced the SDR concept to the public in the early 90s [8]. The idea is that instead of using sophisticated chip-sets for the signal processing, software programmable (signal) processors take over the role. A SDR therefore consists of a RF front-end and a programmable signal processing unit. The front-end contains analog-to-digital converter (ADC) and digital-to-analog converter (DAC) units to transform the digitally processed, complex samples into analog ones, which are up or down converted to the desired frequency by an analogue RF mixer. Thus, the usage of SDR can bridge the gap between idealistic simulations and real-world experiments. Moreover, an SDR can be utilized to adapt to several transmission techniques instead of one fixed baseband chip-set. As a consequence, the goal of SDR developments is to fulfill many different application scenarios using tailored software solutions.

The first military SDR project SpeakEasy [9] started in 1991 with the goal of implementing 10 transmission techniques, using software based signal processing. One of the outcomes of this project was the fact that SDR technology can fulfill the requirements, however the complexity of the software development is huge. Thus, one focus was to increase the software portability, which lead to the Software Communications Architecture (SCA) [10].

Afterwards, SDRs gained interest by additional use cases such as for measurement equipment or cellular infrastructure, according to [11] and as shown in Fig. 1.4. Since 1997, the US military is working to include 35 very different waveforms in one re-programmable device [12]. The idea is to implement all possible communication links as a software application which can then be downloaded into SDR modules deployed in aircrafts, ships or inside the hand held devices for soldiers.

Telecommunication industry introduced the O-RAN alliance [13] to utilize SDR concepts to deploy the complete mobile network stack as software running on a cloud infrastructure. Also, research and academia picked up this topic for prototyping use cases [14, pg.63]. For example, to perform spectrum monitoring where the frequency usage is registered and classified. Later, cognitive radios [15] could use this information to avoid harmful interference by adapting the transmission parameters. Especially for cognitive radios, waveforms such as GFDM or filter bank multicarrier (FBMC) were introduced to allow a better spectral usage [16].

Currently, using SDRs is a widely adopted method of studying modern communication systems. In 2017, the European Union funded the ORCA project to support the development of fully flexible SDR solutions. The general structure of the ORCA architecture is depicted in Fig. 1.5 and shows the key components needed to utilize SDR technologies for wireless networks. The goal of this thesis is to present a concept based on scientific understanding towards an experimentation platform covering those major aspects and yet being programmable for further experiments.

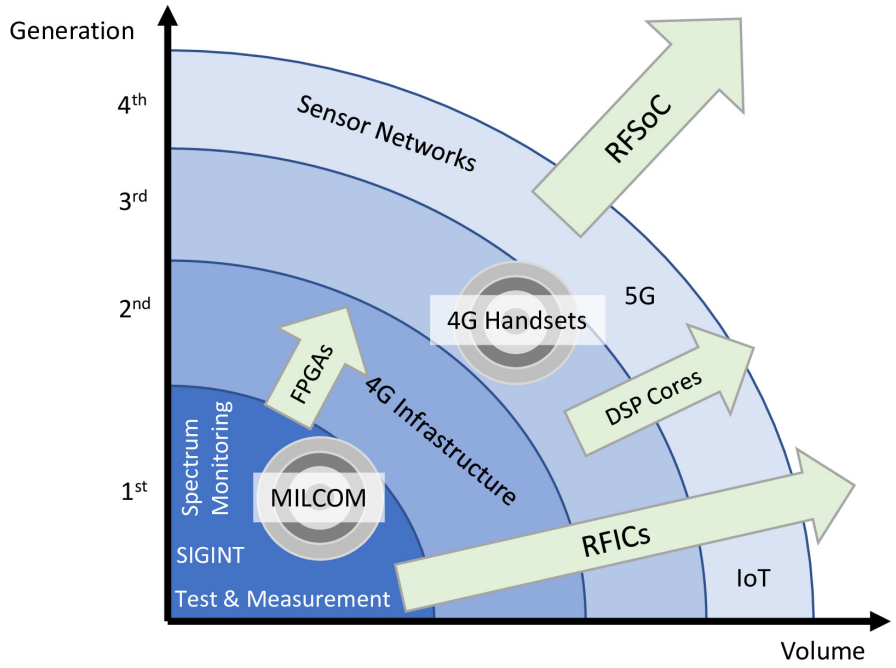


Figure 1.4: Adapted from [11]: "Successive generations of SDRs have come to dominate the radio industry and will continue to evolve."

1.4 State of the art

The core of this work consists of a flexible FPGA based SDR, which can be reconfigured depending on the application scenario. The first question that needs to be addressed is how to meet the requirements of the different wireless applications. Tab. 1.1 summarizes the demands of typical applications on the communication network. These values influence the reference parameters for the evaluations conducted in the following chapters.

The first parameter of importance is the size of the packet, due to its influence on the length of the wireless frame. The packet rate corresponds to the throughput and the latency demands of the respective application. The last parameter, the goodput, states the amount of data that needs to be transmitted successfully by the wireless network to ensure an error free communication link for the application.

	Packet Size [Byte]	Packet Rate [packets per second]	Goodput [byte per second]
Video (VLC), SD	1316	79	104 k
Video (VLC), HD	1316	1938	2550 k
Voice (G.711)	160	50	8 k
Control Loop	50	2000	100 k
Sensor	12	0.002[17] - 100[18]	0.02 - 20k

Table 1.1: Requirements of different example applications.

Tab. 1.1 also contains five different application types. Video streaming applications are represented here by the popular and open source VLC media player application, which can stream single definition (SD) or high definition (HD) videos using UDP packets. The size of those packets is close to the maximum transmission unit (MTU) of 1500 Bytes for Ethernet packets. Larger packets usually get split into smaller sizes, depending on the MTU settings for the network. An HD video stream creates a packet every half millisecond. Voice applications, as another example of traditional networking use cases, are represented by the ITU guidelines for Voice over IP telephone networks. The packet size and rate is much smaller than for the video streams.

In terms of the future factory use case, the authors of [19] collected the key performance indicators for industrial applications, summarized in Tab. 1.2. The message sizes are even smaller compared to the previous applications. However, the packet rates vary from one packet every half millisecond to one packet per day. The requirements for the "Closed Loop Motion Control" use case are relevant for URLLC networks and since this work studies low latency demonstrators in chapter 6, the respective packet size of 50 Bytes and packet rate are noted in Tab. 1.1 as Control Loop.

The payload of a typical sensor application has a size of around 12 Bytes, like the maximum message size in a *SIGFOX* sensor network [17]. Depending on the application, the rate for transmitting those sensor messages varies from a few updates per day, e.g. temperature, up to one update every 10 ms [18], for instance for motor control sensors.

Requirement	Cooperative transport of goods	Closed Loop Motion Control	Additive Sensing for Proc. Autom.	Remote Control for Proc. Autom.
Cycle time	10 ms	0.5 ms - 2 ms	1 h - 1 day	50 ms
Message size	46 Bytes	20 - 50 Bytes	100 Bytes - 10 MBytes	n/a (video stream)
Data rate per entity	50 kbit/s	1 Mbit/s - 10 MBit/s	(burst transmission)	1 Mbit/s - 100 Mbit/s
Message error rate	10^{-7}	10^{-9} - 10^{-8}	10^{-4}	10^{-7}
Latency	<10 ms	« 50% of cycle time	not in focus	50 ms
Distance of entities	up to several km	up to several 10 m	up to several km	up to several 100 m
Velocity	50 km/h	2 - 20 m/s	n/a	n/a
Traffic type	cyclic, broadcast	cyclic, uni- or multicast	cyclic, uni- or multicast	cyclic, on-demand
Entity density	2 km^2 - 30 km^2	0.1 m^2	Up to 10.000 km^2	1.000 km^2

Table 1.2: Reprinted from [19]: Overview of the diverse requirements onto the 5G network to enable industry 4.0 applications

1.5 Testbed

The applicability of new ideas needs to be tested. For this purpose, the Institute of Electrical and Electronics Engineers (IEEE) published a standard for System, Software,

and Hardware Verification, and Validation: IEEE Std 1012-2016 [20]. The aim is to "provide an objective assessment of products and processes throughout their life cycle [...], not at their conclusion." Thus, test solutions for repeatable experiments of wireless networks are needed. As a consequence, many funding programs such as GENI (Global Environment for Network Innovations) [21], Future Internet Research & Experimentation (FIRE) [22] or Platforms for Advanced Wireless Research (PAWR) [23] are financing various initiatives to build testbeds for future internet research. The European commission invested over €1917M between 2007 and 2013 for research with respect to future networks [24]. Research on 5G communication networks was supported with €700M as part of the European Horizon 2020 program between 2014 and 2020 [25].

Currently, testbeds exist in various forms and sizes. They can be realized as two prototype devices in a laboratory environment [26], as a few IoT devices in a dollhouse [27] or as city scale installations with many prototypes distributed across a large area[28]. Some testbeds [29] have advanced back-ends to automatically track and replay every state of the experiments. Still, the IEEE Future Networks Testbed Working Group states in [30] that although there are many 5G testbeds and trials, there is a lack of standards, cooperation, common planning and collaboration with respect to testbeds.

A testbed is defined in this work as several prototypes embedded into a (generic) test infrastructure to track and control experiments. Furthermore, the testbed should (partially) expand out of a laboratory environment to provide more significance for the experiments compared to close distance communication links inside the laboratory.

1.6 Summary

This work can be shortly introduced with:

1. There is a large interest and demand for prototyping solutions targeting upcoming network generations. So, versatile testbeds are needed to evaluate novel techniques in realistic conditions.

2. Moreover, new applications such as the tactile internet demand different network requirements than classical applications. For instance, through an increase in the amount of (short) messages exchanged per second and different dependencies on the packet error rate.

Thus, this work presents a concept to prototype modern wireless networks and applications using a flexible architecture based on SDR technology. The requirements in Tab. 1.1 presents that the data rates for a wireless client are in the range of up to 1 MBit, except for high-definition video streams. The next chapter shows how state-of-the-art wireless networks handle those applications.

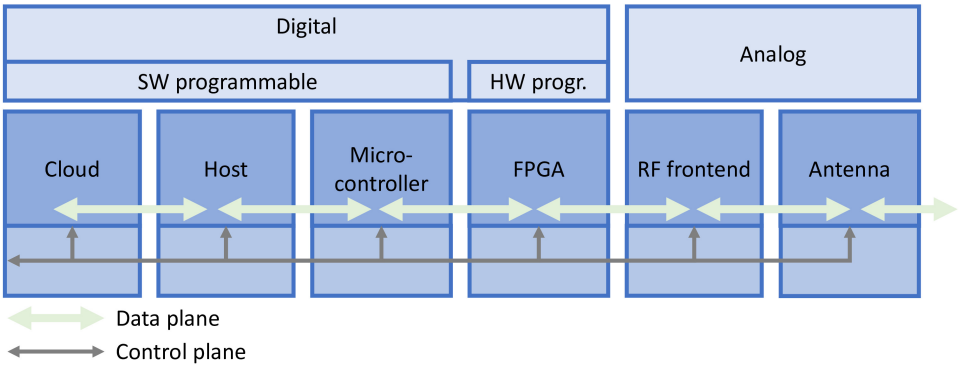


Figure 1.5: Adapted from [31]: Full Stack Overview from the ORCA project

The further work is structured based on Fig. 1.5, which indicates that the overall work consists of multiple components. These are discussed in chapter 3 and comprise the challenges and prerequisites before experiments with networked applications can be carried out. The core of most modern SDR is an FPGA in various sizes. A flexible implementation targeting different wireless networks running on the FPGA is therefore a major component, detailed in chapter 4. However, the running environment of such an FPGA has an influence on the performance of the implementation too. Chapter 5 presents an architecture to embed such SDR platforms with the respective FPGA into a prototyping playground, in order to conduct various experiments. The following chapter 6 studies the overall performance using all the components. A control loop application will act as an example to present and evaluate the hardware- and software framework. The lessons learned by setting up such a prototyping environment are discussed in chapter 7, including considerations regarding future testbeds. Finally, chapter 8 summarizes the work and gives an outlook on how the developed solutions can be improved further.

The focus of this work lies on the physical layer aspects of a communication network, such as how messages can be transferred with novel algorithms using SDR over a wireless channel. To limit the scope of this work, important topics such as secure communications are excluded although they are major concerns especially for industrial applications. This thesis is influenced by higher layer characteristics such as the operating system (OS) too, as they are necessary to study the transmission of network packets between control application and client. However, a more comprehensive review and comparison between a multitude of OSs environments is out of scope here as well.

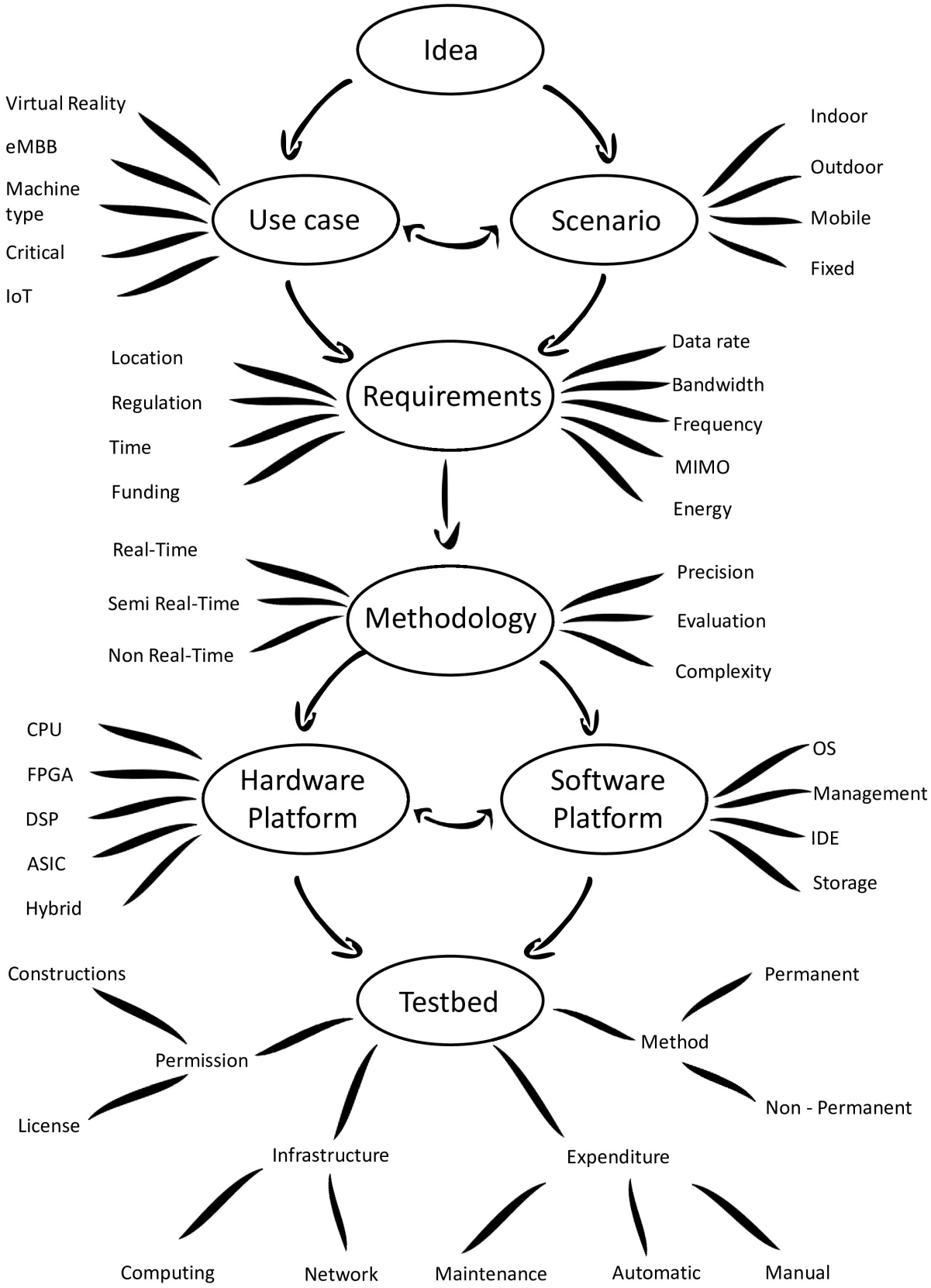


Figure 1.6: Illustrative approach to prototyping wireless algorithms

Chapter 2

Background

Any communication system will face disturbances due to real world effects and imperfections. For example, a piano piece is affected severely by the acoustic environment. Fast piano pieces will not sound pleasantly in environments with huge echos, such as in churches or domes, because the acoustic echos are long and interfere with the upcoming beats such that the whole track will sound unrhythmic. Similarly, modern communication systems are often based on multi-carrier techniques which are using several parallel frequencies to transfer messages. Depending on the environment, reflections as well as other impairments will affect the transmission. Thus, several strategies have been developed to cope with those effects.

This chapter will introduce basic communication techniques and a generalization concept namely generalized frequency division multiplexing (GFDM) to flexibly react on application requirements, as well as on environmental changes. In addition, three prominent wireless networking standards are presented to give an overview of current solutions. The focus in this work is on the physical layer (PHY) layer, which is the closest to physical impairments. Although the performance of the overall system is defined by all layers, the PHY layer is most impacted by analog environmental effects and therefore only be analyzed very restrictively by simulations only. Higher network layer are mainly managing the packet flows delivered by the PHY and therefore contribute to the system performance with latency in particular.

2.1 System Model

The assumed system model is shown in Fig. 2.1. It consists of a baseband processor, the Software-defined radio (SDR) frontend, on the transmitter side a power amplifier (PA) and on the receiver side a low noise amplifier (LNA). First, the binary sequence $\boldsymbol{b}$, created by a test application, enters the digital baseband processor. Its details are given in Section 2.3 and chapter 4. At the output the sequence of the digital transmit signal $\boldsymbol{x}$ enters

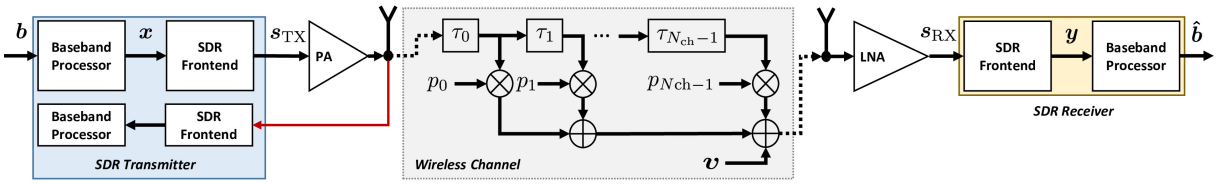


Figure 2.1: Model of the wireless communication link consisting of one transmitter, the wireless channel and one receiver.

the SDR radio frequency (RF) frontend. It consists of digital-to-analog converter (DAC), followed by a mixer to shift the signal $\boldsymbol{s}_{\mathrm{TX}}$ to the desired frequency. This SDR RF frontend is considered as a black box throughout this work, because it is part of the hardware platform.

The RF signal $\boldsymbol{s}_{\mathrm{TX}}$ needs to comply with a level specified in a wireless standard. So it needs to be amplified with a PA at the output of the transmitter. However, any amplifier has a linear and a non-linear operation region. This type of transfer function introduces non-linear distortions at the output, in case the power level of the transmit signal $\boldsymbol{s}_{\mathrm{TX}}$ is too high. Thus, it is important to drive the signal sufficiently, while ensuring that it does not get clipped.

Clipping means that signal peaks are capped or distorted, such that a sine wave for example is distorted into a rectangular-like shape. This would lead to a worse out-of-band (OOB) emission, because those unwanted signal portions harm the transmit sequence. Further, the reception is affected as well, because the transmitted sequence deviates from the intended one. For instance, the synchronization at the receiver often correlates the recorded signal $\boldsymbol{y}$ with a known sequence to find the start of a frame.

Further, if multiple signals are combined at a receiver, for instance in a multi-user set-up, then the stronger signals cover the weaker ones, because the receiving analog-to-digital converter (ADC) has a limited resolution for the quantization. Reference [32] shows how this dynamic range of a ADC affects the reception of multiple users in a Long Term Evolution (LTE) system. Unfortunately, modern multi-carrier system do have a high peak-to-average power ratio (PAPR) [33, pg.331], with high signal peaks compared to the average signal energy. So, a certain headroom between peak and average is required to prevent harmful clipping.

After the signal is amplified the transmit antenna radiates it. Here the far-field designates the distance from the antenna, where the radiated power decreases as the square of distance and the absorption does not influence the transmitter. Common assumptions with respect to the behavior of the wireless channel are applicable in the far-field region [34]. As long as dipole antennas are used, the far-field begins approximately twice the wavelength $r \geqslant 2\lambda$. In case long antennas or antenna arrays are used, such as in a massive MIMO or Millimeter-Wave (mmWave) demonstrator, the size of the antenna D has a influence: $r = 2D^2/\lambda$. So, the 26 GHz mmWave experiments in chapter 6 in the

laboratory might be influenced by the immediate surroundings and should be kept free of metallic surfaces, because the far-field begins after a distance of 4 m.

In addition, the strong transmit signal, compared to the received one has implications on the receiver. Fig. 2.1 indicates this with a red line. It can saturate the ADC such that the weak received signal is outside the dynamic range [32] and cannot be recovered. In the worst case, the transmit signal damages the receiver RF logic. Broadband SDR platforms, used in this work, are more subject to this particular issue compared to standard compliant commercial devices. They have built-in band pass filters that are adjusted to the respective wireless standard.

The wireless channel $\boldsymbol{h}(t)$ is commonly modeled using a finite impulse response (FIR) filter [35, pg.32-36], where each branch has a certain weight p_i and delay τ_i. At the receiver all branches are linearly superimposed. This models the different reflections from different objects in the surroundings of the transmitter and receiver. Each reflection has a varying distance which results in a specific time delay and path loss that affects the attenuation of the signal strength. Thus, the received signal can be seen as a linear time-invariant system:

$$\boldsymbol{s}_{\mathrm{RX}}(t) = \sum_{i}^{N_{\mathrm{ch}}-1} p_i \boldsymbol{s}_{\mathrm{TX}}(t - \tau_i). \tag{2.1}$$

The parametrization of the delays and weights can be based on the 3GPP channel models for LTE [36], where the Extended Pedestrian A model (EPA) is most relevant for this work, because of the short delays and slow mobility compared to the other models.

The experiments in this work are mainly static, i.e. the antennas are not moving. Thus, channel effects such as the Doppler effect do not have a significant impact. However, there is still a frequency shift along the signaling chain in Fig. 2.1, because the oscillators inside the SDR frontend creating the center frequency of transmitter and receiver are independent from each other. Since both oscillators are free-running, they differ from each other and this offset varies over time. This has the effect that a given sub-carrier is shifted in frequency at the receiver.

At the receiver side, the signal needs to be amplified to reach a sufficient signal strength for the analog-digital conversion. As known from the Friis formula, the first amplifier stage defines the noise for the overall system, in case the gain is sufficiently high [37]. Therefore, an LNA is selected. The analog signal $\boldsymbol{s}_{\mathrm{RX}}$ is fed to the SDR receiver frontend to convert it down to the baseband frequency and finally to get a digital quantized version $\boldsymbol{y}$ of it.

Commonly, the modeled channel effects are applied to the digital transmit signal $\boldsymbol{x}$ by the means of a convolution to get the received digital signal $\boldsymbol{y}$, where $\boldsymbol{v}$ denotes all the noise influences generated by all electronic components and is added to the channel model.

$$\boldsymbol{y} = \boldsymbol{h} * \boldsymbol{x} + \boldsymbol{v}. \tag{2.2}$$

Digital communication systems operate with sampled sequences, such that the continuous time t is replaced by a discrete variable $n = t/T_{\mathrm{s}}$, where T_{s} is the sampling period. η_i

denotes the integer tap delays, which can combine the influences of several reflections due to the limited time resolution. The digital received signal $\boldsymbol{y}$ can be given as:

$$y[n] = \boldsymbol{v} + \sum_{i}^{N_{\text{ch}}-1} p_i \boldsymbol{x}[n - \eta_i]. \tag{2.3}$$

At the receiver the baseband processor needs to synchronize to known training sequences and estimates the channel. Based on this it equalizes the channel effects to gain $\hat{\boldsymbol{x}}$. After the demodulation process the final received bit sequence $\widehat{\boldsymbol{b}}$ can be decoded.

2.2 PHY Layer Structure

For successful transmissions, the overall communication system has to agree on certain conventions. For example, in a music composition the beat provides a basic framework for the song and different players as well as the listener can adapt it. Similarly in a wireless network, a common communication medium needs to be shared by different clients and a frame structure such as depicted in Fig. 2.2 organizes it.

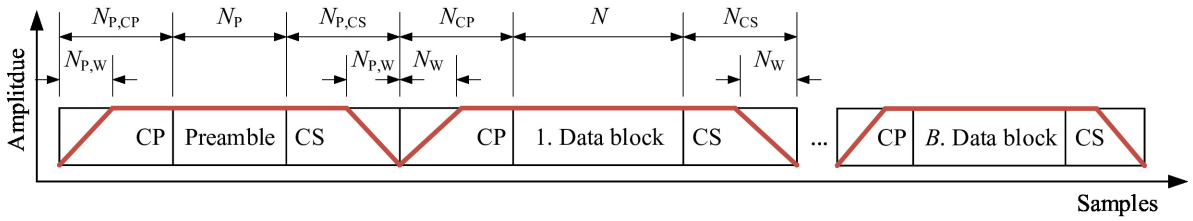


Figure 2.2: Considered generic frame structure with one preamble and B data blocks. The red line symbolizes a windowing function to smooth the signal at the edges.

A common method to signal the begin of a transmission is to use a known sequence at the start of each frame [38]. In Fig. 2.2 the so called preamble takes over this role, where N_{P} defines the length of this sequence. A receiver will constantly listen for this sequence to define the start of a transmission frame, where the data symbols of length N can be modulated in several formats. Further, multiple B data symbols can follow the preamble to transfer longer payload messages, where each individual data symbol is denoted with i.

Before the (de-) modulation process can be conducted, the influence of the channel has to be respected. A common method to mitigate the channel effects is utilizing a cyclic prefix (CP), especially for multicarrier (MC) waveforms. The CP contains N_{CP} samples of the end of a data block, which are copied to the begin [39]. This provides a guard interval against the delay effects of the channel, as long as the echos of the reflections are not longer than the time duration of the samples in the CP. Moreover, it allows using the discrete Fourier transform (DFT) for frequency domain signal processing, especially in the

channel equalizer, since the influence of the channel can be seen as a circular convolution [39].

The cyclic suffix (CS) follows the same principle of the CP, but copies N_{CS} samples of the beginning to the end. The concept of CP and CS can be applied on the preamble too. Therefore, $N_{\mathrm{P,CP}}$ defines the length of the CP and $N_{\mathrm{P,CS}}$ defines the length of the CS for the preamble. The purpose of the CS is to enable the usage of windowing to suppress OOB emissions, as already indicated in the pre-standard definition of 802.11a [40]. OOB occur due to sharp edges in the signal in time domain. For instance, a sharp edge is the abrupt start of a non-zero sequence embedded into silence, because no further signal was sent before or after. Another example for a sharp edge is the abrupt change from the preamble to a data block. A windowing function applies a ramp on the beginning and end of the respective sequences to smooth this transition. However, to ensure that no data is lost, the CP and CS provide redundant dummy samples. It is assumed that the window is symmetric, thus the length of one half is given by $N_{\mathrm{P,W}}$ and N_{W}, respectively.

The message transmitted inside the data block can be modulated using different schemes, which are called waveforms in the context of this thesis. Besides traditional modulation schemes such as amplitude or frequency modulation, three waveforms are relevant for current wireless networks: single-carrier (SC), MC and spread spectrum [39, pg.83-84].

- ▷ MC waveforms are used in many standards, such as LTE, wireless local area network (WLAN) 802.11g [41], mostly using the popular waveform orthogonal frequency division multiplexing (OFDM). The data symbols in a MC system are mapped parallel in frequency with the help of the (inverse) Fourier-transform and represent K parallel subcarriers. The OFDM subcarriers are orthogonal to each other, such that there is no inter-carrier interference (ICI) [41]. The advantage of OFDM is that the influence of a static channel can be dealt with the help of a CP. Further, multiple input multiple output (MIMO) techniques can be applied easier, than in the SC or spread spectrum case. The disadvantage of MC waveforms is the high PAPR, which denotes the ratio between the signal peak and the average signal power.
- ▷ SC is using only a single carrier to transfer the quadrature amplitude modulation (QAM) modulated data symbols M after each other in time. This has the advantage of a much better PAPR because unlike the fast Fourier transform (FFT) operation in OFDM, there are significantly less super positions of data symbols in time. This avoids high peaks in the transmitted time domain signal. Especially mobile user-terminals (UTs) benefit, since the linearity and therefore the energy efficiency of RF PA can be reduced. However, the disadvantage is the handling of the channel effects, because multiple signal copies due to the multi-path environment have to be considered. This can be avoided by introducing the CP concept, which results in single-carrier with frequency domain equalization (SC-FDE) [42]. The receiver will then perform the equalization process in frequency domain similar to OFDM. Due to the nicer PAPR properties, the UT in a LTE system use single-carrier frequency

division multiple access (SC-FDMA) for the uplink transmissions [43]. This way the PA can be designed more energy efficient than at a base station (BS). The difference to a conventional SC-FDE is the usage of a additional DFT at the transmitter to shift the signals of a certain user to a certain subcarrier frequency. The BS will then receive the different signals from multiple users like a OFDM signal.

- ▷ A spread spectrum based system maps binary data symbols to a unique sequence consisting of multiple chips. These sequences are then spread over the bandwidth [39]. Spread spectrum has a long history in military communications [44] in particular due to its robustness against jamming. In case of ultra-wide band (UWB), the signal power can be so low, that the signal levels are below noise levels. Only a receiver integrating the signal power over the same bandwidth, can recover a certain signal strength to detect and decode the transmitted messages. Further, the PAPR depends on the spreading sequence and can therefore also be considerably low. Spread spectrum have to deal differently with channel effects than OFDM or SC-FDE, e.g. by employing a Rake-receiver [35].

2.3 Generalized Frequency Division Multiplexing

GFDM has been introduced [45] as a fifth generation (5G) candidate waveform, to offer advantages like reduced OOB emissions or to provide lower latency. However, GFDM can be further seen as a generalization framework to cover multiple waveforms as well [46]. The concept of GFDM transceiver structures itself are described in [47]. Synchronization techniques of GFDM receivers are detailed in [48]. The extensions with respect to GFDM MIMO wireless systems are given in [49]. Channel estimation was detailed in [50]. This section gives a brief overview over this MC waveform, because it is combining aspects of [47] and [48] in the single-input and single-output (SISO) implementation presented in chapter 4.

The idea of GFDM is to extend the one dimensional structure of OFDM into a two dimensional one, having K subcarriers and M subsmybols. Fig. 2.3 shows the resource grid with $K \cdot M = N$ data symbols in total. For example, OFDM is a corner case with K subcarrier and $M = 1$ subsymbol and in contrast SC has $K = 1$ subcarrier and M subsymbols. In the equations, the sample count within one GFDM symbol i is denoted with n, the current subcarrier with k and the current subsymbol with m. A CP or CS is added later for the whole data block and not for each subsymbol as in the OFDM case. Each stream of data symbols $\boldsymbol{d}_{k,m,i}$ is modulated with transmitter pulse shapes $\boldsymbol{g}^t_{k,m}[n]$. Hence, the discrete transmitted signal $\boldsymbol{x}^t$ can be written as

$$\begin{aligned} \boldsymbol{x}^t[n] &= \sum_{i=-\infty}^{\infty} \sum_{k \in K_{\text{on}}} \sum_{m \in M_{\text{on}}} \boldsymbol{d}_{k,m,i} \boldsymbol{g}^t_{k,m}[n - iN_s] \\ &= \sum_{i=-\infty}^{\infty} \boldsymbol{x}^t_i[n - iN_s]. \end{aligned} \tag{2.4}$$

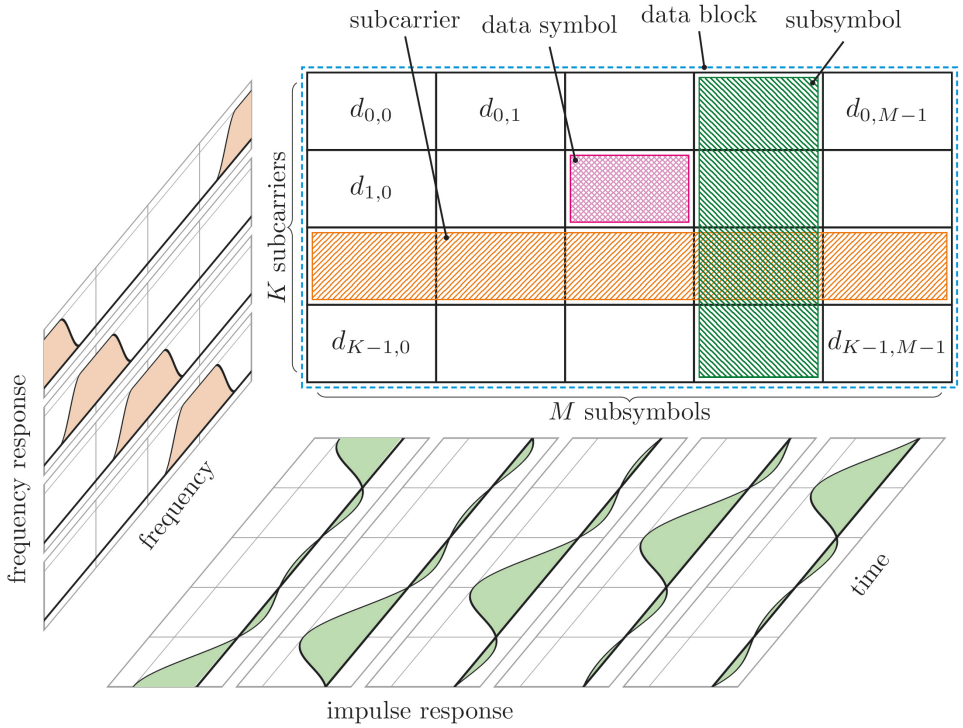


Figure 2.3: Illustration taken from [51], where the GFDM resource grid is using $K = 4$ subcarrier and $M = 5$ subsymbols.

where N_s is the symbol spacing, $\mathcal{K}_{\text{on}}$ and $\mathcal{M}_{\text{on}}$ are the sets of active subcarriers and subsymbols, respectively, and $\boldsymbol{x}_i^t[n]$ is the i-th multicarrier symbol, which is given by

$$\boldsymbol{x}_i^t[n] = \sum_{k \in K_{\text{on}}} \sum_{m \in M_{\text{on}}} \boldsymbol{d}_{k,m,i} \boldsymbol{g}_{k,m}^t[n]. \tag{2.5}$$

Utilizing fragmented spectrum or applying cognitive radio concepts was one motivation for next generation waveforms, such as GFDM as well as filter bank multicarrier (FBMC) [52] or universally filtered multicarrier (UFMC) [53]. In GFDM a circular pulse shaping filter $\boldsymbol{g}_{k,m}^t[n]$ is used for each subcarrier to suppress the OOB emissions. The two dimensional extension of GFDM is needed to keep the ramp-up and ramp-down phases of the filter within one block, where the filter tails exceeding the block are wrapped around to the beginning, such as depicted as the green impulse responses in Fig. 2.3.

The author in [47] recommends using a prototype pulse shape $\boldsymbol{g}[n]$ based on the circular raised cosine with a small roll-of factor alpha around 0, because it achieves the best PAPR performance of around 12 dB and the weakest self-interference. The filter $g_{k,m}^t[n]$ has a finite length N_t and can be generated using the prototype pulse shape $\boldsymbol{g}[n]$ through circular shift in time and frequency such that

$$\begin{aligned} \boldsymbol{g}_{k,m}^t[n] &= \boldsymbol{g}[\langle n - mK \rangle_N] e^{j2\pi \frac{k}{K} n}, \\ n &= 0, \cdots, N_t - 1, \end{aligned} \tag{2.6}$$

where $\langle \bullet \rangle_N$ denotes the modulo N operation. The modulo operation conducts the circular wrap-around of the pulse shaping filter. The exponential term shifts the data symbol from the respective subcarrier to the time-domain, like the inverse discrete Fourier transform (IDFT) in OFDM.

Paper [54] shows that a time-domain window can reduce the OOB emission further. Therefore, the windowing is applied on the overall block after the CP and CS is added by multiplying with a windowing function, defined as follows

$$\boldsymbol{w}[n] = \begin{cases} \boldsymbol{w}_{\text{rise}}[n] & 0 \leq n < N_{\text{W}} \\ 1 & N_{\text{W}} \leq n < N_{\text{CP}} + N + N_{\text{CS}} - N_{\text{W}} \\ \boldsymbol{w}_{\text{fall}}[n] & N_{\text{CP}} + N + N_{\text{CS}} - N_{\text{W}} < n < N_{\text{CP}} + N + N_{\text{CS}} \end{cases} \tag{2.7}$$

where $\boldsymbol{w}_{\text{rise}}[n]$ and $\boldsymbol{w}_{\text{fall}}[n]$ are the ramp-up and ramp-down segments of the time window, which are here raised cosine based. In Fig. 2.2 this time windowing function is marked in red.

Both techniques, pulse shaping and windowing, allow to use the fragmented spectrum more efficiently than a conventional OFDM system because the OOB emissions are reduced. For example, such as measured in Fig. 2.4, where the GFDM system has a up to 30 dB lower OOB. Here, a well calibrated SDR platform (*National Instruments (NI) PXI 5791*) without a PA was used to achieve the results. Thus a GFDM waveform can be utilized in a cognitive radio (CR) application more efficiently than OFDM because many wireless applications leave guard bands to avoid interference between incompatible systems. This has been validated in [6] and [55], where a GFDM CR utilizes free resource blocks in a LTE uplink (UL) transmission. In [55] the interference to the primary system is reduced by 9 dB in case GFDM is used.

In one GFDM block the amount of data symbols is $N = MK \leq N_t$, however this concept can be extended to other waveforms as well as described in [46]:

Let N_o be the overhead that represents a CP and/or CS, where $N_t = N + N_o$. The type of overhead is defined by the time shift n_0, For instance, $n_0 = N_o$ means pure CP. Based on that, a core block $\boldsymbol{x}_i[n]$ of length N samples can be defined such that

$$\boldsymbol{x}_i[n] = \sum_{k \in K_{\text{on}}} \sum_{m \in M_{\text{on}}} \boldsymbol{d}_{k,m,i} \boldsymbol{g}[\langle n - mK \rangle_N] e^{j2\pi \frac{k}{K} n}. \tag{2.8}$$

The relation between the core block and the actual transmitted block is given by

$$\boldsymbol{x}_i^t[n] = \boldsymbol{x}_i[\langle n - n_0 \rangle_N]. \tag{2.9}$$

Two specific cases evolve depending on N_s and N_t

- ▷ $N_s \geq N_t$: the transmitted blocks $\boldsymbol{x}_i^t[n]$ do not overlap and the total transmitted signal $\boldsymbol{x}^t[n]$ can be generated by transmitting $\boldsymbol{x}_i^t[n]$ independently. Here, $N_P = N_s - N_t$ represents a guard interval between sequential blocks. OFDM and GFDM belong to this case.

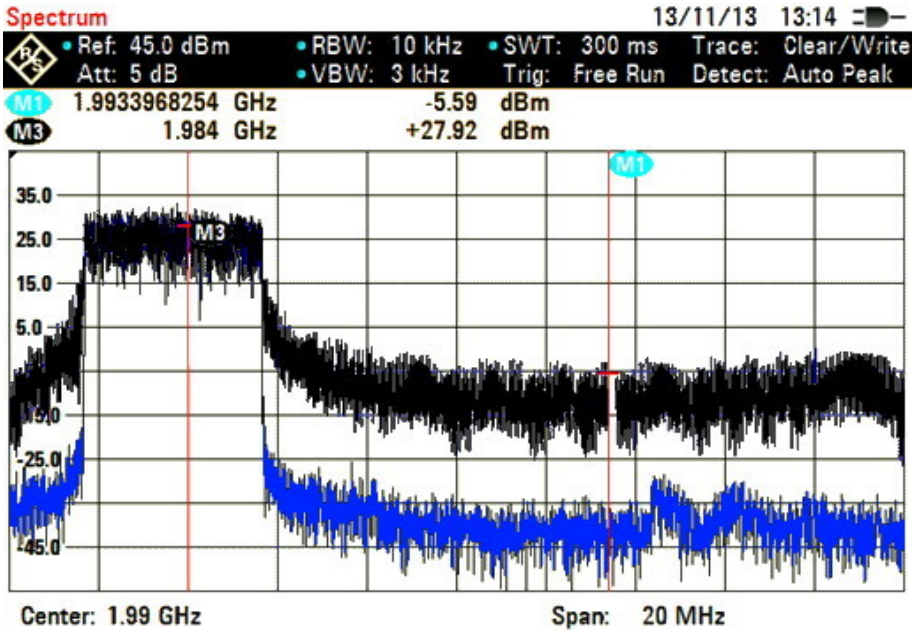


Figure 2.4: Out of band emissions of the GFDM waveform (blue) in comparison to the OFDM waveform (black), based on [55].

- ▷ $N_s < N_t$: $\boldsymbol{x}^t[n]$ can be still produced based on $\boldsymbol{x}_i^t[n]$, however, additional overlapping of length $N_o = N_t - N_s$ must be considered. In this, the last N_o samples of the current $\boldsymbol{x}^t[n]$ are added to the first N_o samples of the next $\boldsymbol{x}_i^t[n]$. An example of this case is filtered multi tone (FMT) [56] with overlapping factor M. In which, $\mathcal{M}_{\text{on}}$ contains one subsymbols and the remaining $M-1$ are non-active. The overlapping by $N_o = K(M-1)$ compensates this overhead. Another example is the weighted-overlap-and-add (WOLA) processing in [57], where the ramp-down of the $i-1$-th windowed block overlaps with ramp-up of the i-th block.

As a result, by the configuration of the prototype pulse shape $\boldsymbol{g}$ in addition to the parameters, N_s, N_t, n_0, K, M, $\mathcal{K}_{\text{on}}$ and $\mathcal{M}_{\text{on}}$ this concept can produce most of the state of the art waveforms. The scope of this work is on the non-overlapping cases. Tab. 2.1 presents considered waveform parameters to emulate different wireless standards.

2.4 Wireless Standards

In this section three prominent wireless standards are introduced very briefly as references to illustrate how different wireless applications are handled by state of the art technology. Those standards are often used as a starting point for many testbeds and wireless experiments. Fig. 2.5 outlines the standards briefly, to symbolize the basic principles. Since the GFDM methodology can emulate several waveforms, Tab. 2.1 summarizes configurations which resemble different standards.

802.11 WLAN and IEEE 802.15.4 are standards utilizing the world-wide unlicensed spectrum and thus, they have to deal with the interference created by the other wireless applications in the same 2.4 GHz frequency range. However, they are subject to various transmit power regulations. According to the specifications [58] a signal can be transmitted without listen before talking (LBT) as long as its transmission power is below 10 dBm. Normally, in the power range until 20 dBm LBT has to be used to not harm ongoing transmissions. Some frequency bands in the 5 GHz industrial, scientific and medical band (ISM) region allow higher power levels of up to 30 dBm, but with additional regulations. For instance, a device has to check if a (weather) radar system is active in the respective region [59, p. 418].

Contrary, LTE standards are mainly employed in licensed bands and are thus set-up by network operators which control all network parameters. Ultra-Reliable and Low-Latency Communication (URLLC) networks, particularly interesting for factory of the future use cases, are not standardized yet. Germany provides a 100 MHz wide band around 3.75 GHz for local usage for minimal fees, which is currently unique in the world [60]. As a result, owners of ground can apply for a spectrum license on their ground without employing network operators.

2.4.1 IEEE 802.15.4

The standard IEEE 802.15.4 defines a transmission protocol for wireless personal area networks. One popular variant is ZIGBEE, which is using a pseudo noise (PN) sequence to spread the data over the given bandwidth and is therefore a representative of a direct-sequence spread spectrum (DSSS) waveform here. Each of the 16 sequences with a length of 32 chips defines 4 data bits. The channel bandwidth depends on the used frequency. In the 2.4 GHz band it is 2 MHz. ZIGBEE uses a preamble for synchronization as well, which can be used for channel estimation and localization purposes [61]. Similar to the WLAN signaling field, ZIGBEE also indicates parameters such as the start of the frame and the frame length in a PHY header. A MAC header indicates further information about the frame, the sequence number and includes a cyclic redundancy check (CRC). In contrast to WLAN the acknowledgment of a packet is to be sent within 864 μs or can be left out.

As a mixture between WLAN and LTE, ZIGBEE offers scheduled and non-scheduled multiple access [62]. Therefore, a super-frame structure is defined. The access point (AP) initiates this via beacon packets and defines time slots for scheduled transmissions, a free access and an inactive period. The contention access period (CAP) contains 10 time slots, where LBT has to be used to access them. The contention free period (CFP) defines 5 time slots with scheduled transmissions. The duration of the super frame structure can be up to 246 s to reduce the energy consumption of the devices.

2.4.2 802.11 WLAN

Wireless local area network (WLAN) is commonly used for Internet access on premises. The modern variants use OFDM with a subcarrier spacing of 312.5 kHz and a guard interval of 0.8 µs. In case a bandwidth of 20 MHz is used this corresponds to $K = 64$ subcarriers and a CP size of 16 samples. Time domain windowing is part of the proposal for the IEEE 802.11a standard [40]. WLAN utilizes several preambles before the data block. The first preamble, called short training field is used for coarse synchronization. The second preamble, named first long training field is used for fine synchronization and initial channel estimation and contains 160 samples. Thereafter, the signaling field with 64 samples defines the transmission parameters, such as the length of the packet, which can be up to 4095 Bytes. WLAN transmits data packets in bursts as long as there is payload to be transmitted. In addition, transmitted packets have to be acknowledged by the receiver within 10 µs, otherwise a retransmission is triggered on the transmitter (TX) side.

As indicated earlier, LBT has to be used to determine if the channel is free. This happens in the DCF Interframe Space (DIFS) time interval of around 50 µs. In that period, the transmitter will sum up the received samples and if they are above a certain threshold, the medium is seen as busy. Distributed coordination function (DCF) is the underlying MAC technique [63], that combines LBT and a random back-off time, in case the medium is occupied. Then the transmitter would have to wait a given time and perform the LBT again. Besides the hidden node problem [63], there is another issue due to the LBT in case different wireless networking solutions share the same spectrum [64]. For instance, a ZIGBEE transmission only occupies 2 MHz of spectrum, so the amount of accumulated energy seen by the 20 MHz WLAN transmitter is not enough to trigger the LBT threshold. So, WLAN will see the spectrum as free, however the interference by ZIGBEE or other wireless systems [65] is still enough to disturb the transmission, such that the WLAN receiver will trigger a retransmission.

2.4.3 LTE

In contrast to the other schemes is LTE mainly designed for licensed bands. So, LTE has a more complex protocol to optimize the transmission properties for cellular networks. LTE provides a continuous stream of OFDM symbols, because it does not have to share the licensed bands with other protocols.[1] A potential receiver can therefore observe the RF signals over a longer time period and can synchronize more precisely than in case of burst transmissions such as in WLAN.

The fixed frame structure in the downlink (DL) consists of the primary and secondary training periods, which can be seen as preambles for synchronization, because they are

[1] There are different standards to utilize the ISM bands using LTE technology such as LTE-U, LAA or LWA [66].

employing Zadoff-Chu sequences, similar to WLAN. However, those sequences do not cover the whole channel bandwidth, but instead only 60 subcarriers in the center. A receiver tries to synchronize first using those sequences, before starting to decode any data. Further, it constantly tracks the timings using a phase-locked loop (PLL). In case larger LTE bandwidths are used then in Fig. 2.5, data bearing subcarriers are placed besides the Zadoff-Chu sequences. The channel estimation is pilot based using every sixth subcarrier and interpolates the channel impairments for the subcarriers in between.

The OFDM resource grid is organized in frequency domain using physical resource blocks (PRBs) which group 12 subcarriers each. However, only half of the total amount of subcarriers are utilized to leave enough guard space at the spectrum edges. The subcarrier spacing is set to 15 kHz, but the bandwidth can vary between 1.4 and 20 MHz In time-domain the resource grid utilizes radio frames, that are subdivided into 10 subframes containing 14 OFDM symbols each. The first two OFDM symbols of every subframe contain the Physical Downlink Control Channel (PDCCH) which defines PHY and MAC parameters similar to the MAC headers in WLAN or ZIGBEE.

5G networks build upon the LTE standard and are extending it with varying subcarrier spacing, windowing such as presented earlier to suppress OOB and new frequency options up to the mmWave frequency range.

2.4.4 Low Latency Industrial Wireless Communications

Despite the general 5G announcements, there has not been a specification for URLLC networks yet. Industrial wireless applications are therefore using technologies based on the 802.11, 3G or 802.15.4 [67] standards in the 2.4 GHz ISM bands.

Thus, a GFDM based frame format is defined and used later in this work as Low-Latency Communication (LLC), where the aim is to provide low latency communications, and leave the ultra reliable part of URLLC to future works. Inspired by WLAN, $K = 64$ subcarriers are used, with $K_{\text{on}} = 50$ active subcarriers. This leaves guard bands at the edges of the spectrum. Further, the center is unused too to minimize the impact of the direct current (DC) subcarrier. The amount of subsymbols depends on the payload size, typically $M = 15$ is chosen to support a packet size larger than 50 Bytes. In case even more data symbols are needed, the amount of GFDM blocks B per preamble can be increased. The sample rate is set to 40 MHz. CP and CS is used to enable windowing and is defined with $N_{\text{CP}} = 32$, $N_{\text{CS}} = 16$.

The MAC proposed for the experimental setup in this work is inspired by a mixture of different standards. All communications are triggered by the AP. A client is only allowed to answer in case the AP scheduled a time-slot, via sending a downlink packet to the respective client. To leave enough time to decode the packet, the (optional) answer is to be sent after 2 time-slots. Unlike many other wireless communication schemes, it is not considered to repeat a data frame. Therefore, acknowledgment (ACK) packets are not

necessary. The reason is that in a control loop application it is better to sent updated status messages than repeating old and potentially outdated messages, just to ensure that every packet is received. A 16 Bit CRC determines if a packet was received successfully. In case a successful data transfer needs to be ensured, higher layer protocols such as transmission control protocol (TCP) can be used. However, to still increase the reliability of the transmission, the last packet can be stored in a buffer and repeated in case no further payload data needs to be sent. The receiver uses a packet counter inside the MAC header to remove repeatedly received data messages.

2.4.5 Summary

The content in this chapter can be summarized in the following statements.

1. The GFDM based waveform framework supports very different configurations to adapt wireless links to the scenario as well as to emulate different modulation schemes [54].

2. Four parameter settings ”LLC”, ”WiFi”, ”eMBB” and ”IoT” have been chosen to evaluate the implementation presented in this work.

Tab. 2.1 highlights the considered parameter settings used in the evaluation chapters. Those settings are adapted to represent different types of wireless networking standards.

Type	Waveform	K	K_{on}	M	M_{on}	Payload	CP / CS	B	Format	Sample rate
LLC	GFDM	64	52	15	13	71 Bytes	32 / 16	1	Burst	40 MHz
WiFi	Block OFDM	64	52	15	15	83 Bytes	32 / 16	1	Burst	20 MHz
eMBB	OFDM	1024	600	1	1	143 Bytes	75	2	Continuous	15,36 MHz
IoT	SC	1	1	1024	1024	121 Bytes	32 / 16	1	Burst	2 MHz

Table 2.1: Considered waveform parameters, where the maximum payload size per frame is related to the most robust modulation coding scheme available in the implementation based on code rate ½ and QPSK.

The first set of GFDM parameters ”LLC” emulate a possible wireless network for low-latency applications in on-premises networks, because WLAN is a very flexible and fast broadband transmission scheme compared to LTE in terms of latency while providing more data throughput than the Zigbee standard. The number of subcarriers are similar to the WLAN specification to ensure sufficient robustness of the communication, however the used sample rate is doubled to minimize the time duration of a packet. Since not all subcarriers are allocated, the configuration should utilize a bandwidth of around 33 MHz such that the 100 MHz of available spectrum in Germany could be utilized by 3 independent networks. The GFDM waveform is used to suppress the OOB emissions further.

The second set of parameters "WiFi" is taking a 20 MHz 802.11a WLAN system as reference, such that the implementation can be compared to the commercially available *NI* 802.11 Application Framework (AFW), which is utilizing the same SDR hardware as well as standard WLAN USB chip-sets. Still, the waveform is a bit different to OFDM because one CP and CS is used for 15 subsymbols and therefore called Block OFDM.

The third set of parameters "eMBB" is inspired by a LTE system configuration for 10 MHz channel bandwidth. Here the data is generated continuously, compared to the burst transmissions of the previous categories. Also the overhead with respect to training sequences is adapted to LTE, such that one preamble, that is utilized for synchronization as well as channel estimation, is used for two OFDM symbols ($B = 2$).

The last category "IoT" is using a single carrier waveform for a nice PAPR performance. The sample rate of 2 MHz is inspired by the ZIGBEE specification for 2.4 GHz ISM bands. The length of the frame is long for typical IoT use cases, however this way all parameter sets have a similar number of data symbols $N \approx 1000$ per waveform symbol. The maximum payload of a ZIGBEE packet is 125 Bytes. The CP is needed to perform SC-FDE techniques at the receiver and CS is used to apply a time domain window on the frame.

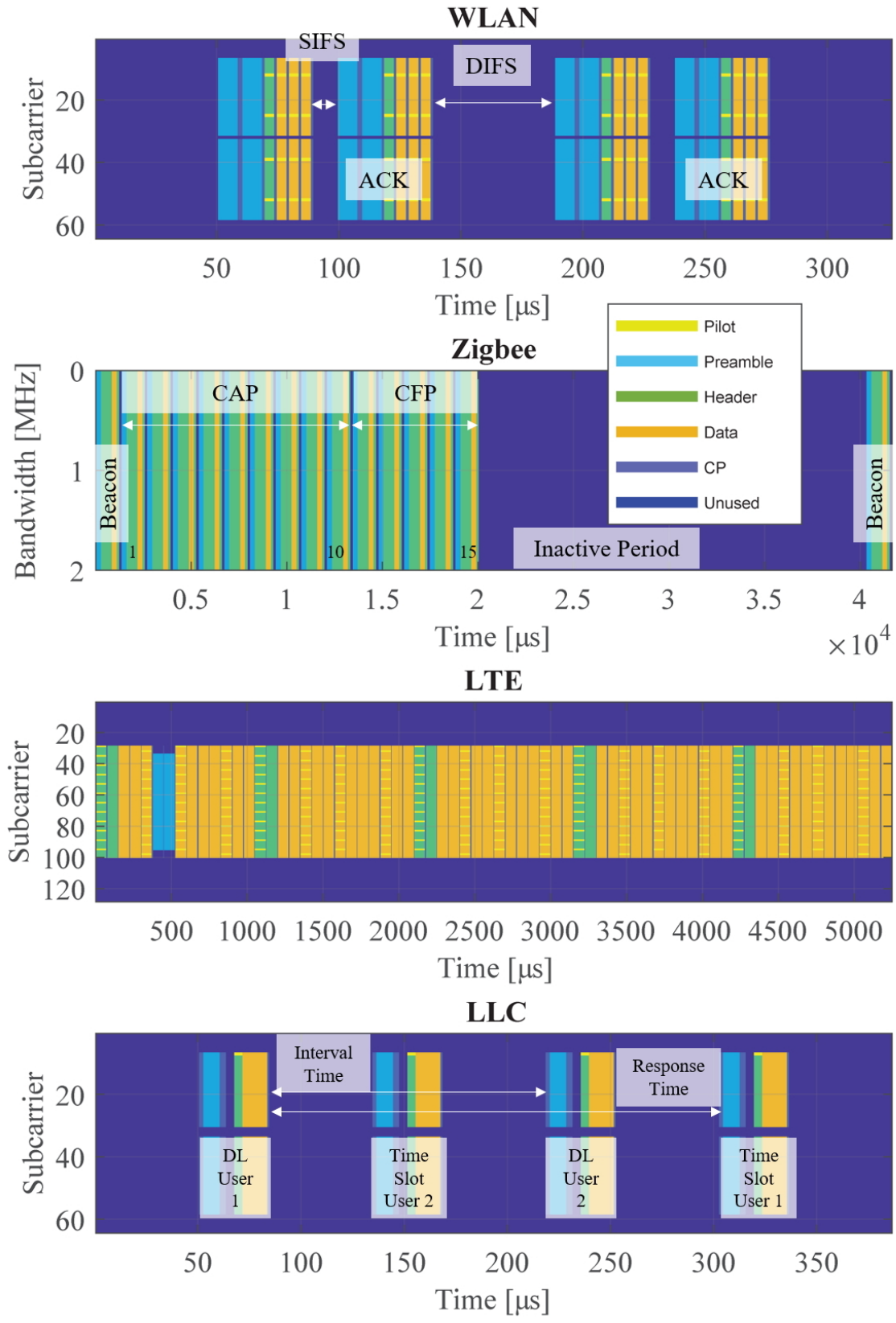


Figure 2.5: Different frame configurations of the wireless standards WLAN 802.11g, ZIGBEE 802.15.4, LTE for 1.25 MHz bandwidth and the proposal for low latency networks used in this work. All channels not used to transmit payload information are seen here as header or medium access control (MAC) information.

Chapter 3

Wireless Prototyping

The introduction chapter presented the Software-defined radio (SDR) concept where every waveform could be (re-)implemented in software to adapt to very different wireless standards and applications. Even though a SDR implementation sounds like a perfect solution to minimize maintenance and improved upgrades, it does entail further challenges. For instance, licensed software typically require a license server. If such a license server fails, whole communication networks can be shutdown, such as the O2 network in England in 2018 [68]. Therefore, appropriate test environments are inevitable, because SDR solutions have disadvantages as well.

According to [69], the military Joint Tactical Radio System consortium is still looking for the optimal SDR software solution, due to the many drawbacks of a complex software framework. After 20 years of development 6 out of 35 waveforms were supported [70] and with a cost increase of the device of around 14 times its initial projection [69], despite having a total development budget of 6 billion $. The main reason is that the overall complexity of the system is underestimated [71], even with a fully standardized architecture such as the open Software Communications Architecture (SCA). So in 2018, the United States Congress granted another 4 billion $ project to fix the issues with the Joint Tactical Radio System [69]. Another downside concerns the energy consumption, which is seen as a major drawback of software network stacks for cellular use cases such as O-RAN [13]. So, the National Science Foundation (NSF) encourages the research using SDR and software-defined networking (SDN) technologies with 100 Million $ [72] across different testbeds in the United States via the Platforms for Advanced Wireless Research (PAWR) program. A portion of the funding is given by companies. Canada follows a very similar public-private 5G partnership model with a total of 400 Million $ to study modern networks in the ENCQOR (Evolution of Networked Services through a Corridor in Quebec and Ontario for Research and Innovation) program [73].

Selecting an optimal SDR solution depends therefore on several factors, where the targeted application and use case are the principal element. For instance, the latency requirements define the hardware platform, e.g. if the software needs to run in real-time (FPGA) or not

(CPU). This has influence on the development effort. For instance, the field programmable gate array (FPGA) solution might not be as flexible and is much more development time consuming than conventional C developments. On the other hand the FPGA has typically a much higher processing speed.

However, communication research or test is not limited to SDR devices only. Many ideas, especially concerning higher network layers, can be tested with proven commercial off-the-shelf (COTS) technology, too. For instance, wireless local area network (WLAN) router such as the popular *Linksys* WRT54GL supports customized firmware. Popular open-source firmware amongst others are OpenWRT or DD-WRT. Although the router WRT54GL has been introduced in December 2002, it is still officially available due to the popularity of the firmware alternatives. However, those modifications are limited to networking functionalities beyond the physical layer (PHY) and medium access control (MAC) layer considered in this work, because manufacturer are reluctant to offer programmable WLAN modules. Only very few exceptions are known such as the WiLink8 product family from *Texas Instruments*, which seems to grant a bit further access to the network stack on request. For instance, *3RComs* is using those chip-sets in their Echoring modules to allow more reliable communications for industrial use cases based on modified WLAN [74]. As a consequence, many experiments use IEEE 802.15.4 (ZIGBEE) based solutions, where most of the MAC functionality is implemented in software running on a micro controller [75], because the intention is to keep IEEE 802.15.4 devices as simple and cheap as possible.

The aim of Fig. 1.6 in the first chapter is to sketch out a strategy to design prototyping environments. Further, it outlines that there is no universal or generic way, because several trade-offs have to be made. Important is, that a novel idea need to be tested in the targeted environment to prove the practicability. The following sections contribute with a conceptual strategy for designing testbeds. The contributions are based on [76].

Most device examples in the following sections are from the company *National Instruments (NI)*, because they target a large open-source community around the GNU Radio project and therefore disclose detailed data sheets as well as prices. This allows to compare the different device classes, ranging from credit-card sized models to complex measurement equipment. Other vendors offer various devices too, but not across the different classes.

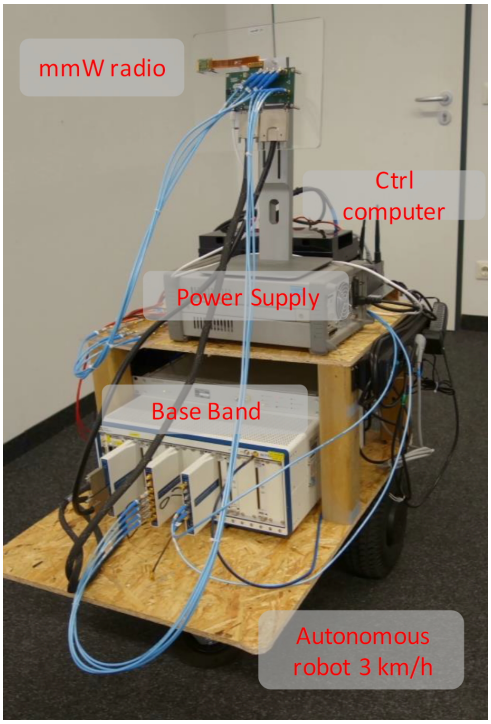
Prior to discussing hardware (Section 3.6) or software (Section 3.7) details, a overview over different testbed examples is given in the next section. The following sections study prerequisites that should be considered before planning set-ups for experiments, because they possibly have impacts on the development efforts. The deployment and operation of a testbed is subject for Section 3.8, because buying equipment is ”simple” compared to maintaining continuous operation. This chapter concludes discussing the introduced testbed examples.

3.1 Testbed Examples

In the past decade, many testbeds were built, capturing different aspects of communication networks. Many of them being optimized for a certain use case. A common definition is that one testbed or experimental node consist of a control or host computer with one or several wireless devices, such as SDR or COTS wireless adapter. The testbeds can be roughly grouped in three categories ”PHY”, ”MAC” and ”Network” depending on the targeted use case and are briefly introduced here.

3.1.1 PHY - focused Testbeds

These test facilities concentrate on the PHY layer algorithms and often perform link-level experiments, circumventing the challenges concerning multiple users and higher network protocols. Thus, MAC or higher network layer, are simplified to basic functionalities, for example to keep a link alive.



(a) Reprinted from [26]: 60 GHz mmWave beam tracking set-up

(b) *NI* massive MIMO demonstrator with 24 antennas

Figure 3.1: High throughput prototyping platforms

The mmWave experimentation platform presented in [26] is one example, where the hardware platform consists of several FPGAs to process around 1.5 GS/s sample rate. This set-up can achieve GBit/s data rates due to the parallel processing implemented as part of the European project miwaves [77]. Here, MAC functions such as beam-steering

are implemented as well on a real-time operation system to cope with the channel effects of higher frequencies. The set-up is mobile using a specialized robot platform to carry 100 kg as shown in Fig. 3.1a.

Another prominent example is the massive MIMO testbed in Fig. 3.1b from Lund University, the LuMaMi-Testbed [78]. Here, up to 100 SDR platforms are connected coherently to process 384 Gbps of real time baseband data to serve 10 users simultaneously. Together with the University of Bristol, it achieved the world record in spectral efficiency of up to 145 b/s/Hz [79], serving 22 users sharing the same time-frequency resource.

A similar set-up is used in the Defense Advanced Research Projects Agency (DARPA) spectrum challenge with the aim to compare different cognitive radio (CR) solutions from different development teams against each other [80]. The goal is to transfer as much data as possible over emulated wireless channels under the presence of other CRs and interferers. Doing this, no manual interactions are allowed, thus the CR has to operate automatically. Each CR consists of a server platform and a *NI* USRP X310. This combination permits to use either a solution running solely on general purpose processor (GPP) systems or using the FPGA inside the USRP as well.

The hardware set-up of the massive MIMO system is used to build a 4-tap wireless channel emulator USRPs [81]. Colosseum is a 256-by-256 RF channel emulator and based on 128 USRP X310 developed by the military research institution DARPA. It can calculate and simulate more than 65,000 channel interactions among 256 wireless devices in real-time [82]. DARPA is using Colosseum for the spectrum challenge to test several SDR solution against each other. Here, the radio frequency (RF) signals are down-converted to apply the channel model and up-converted again for the receiving set of USRP and server. Further, the USRPs of the channel emulator are interconnected to simulate crosstalk of all RF signals to all teams. After the competition ended in 2019 the Colosseum testbed was moved to Northeastern University Boston in the United States to allow access from interested researches for further experiments as part of the PAWR research program [83].

The EASY-C project [28] studied coordinated multi point (CoMP) techniques for Long Term Evolution (LTE). As part of this project extensive field trial campaigns where carried out from 2008 to 2012 with the aim to evaluate the capacity enhancements compared to conventional transmission systems. The idea of CoMP is that inter-connected base stations exchange information about the received signals such that transmissions of user-terminals (UTs) can be evaluated jointly. In case a UT is located at the edge between two cells, a conventional transmission system would suffer because of the interference caused by the non-coordinated base stations. In contrast a CoMP system can combine the signals to achieve higher reliability or data rates [28].

Before the EASY-C project, this novel concept was analyzed using simulations. To prove the feasibility of the idea, two major testbeds were built. One in Berlin consisting of three sites [28, pg.331] and one in Dresden with up to 7 sites with 16 base stations (BSs) [28, pg.320]. Typically, one site had several sectors covering 120°. Each sector was equipped

with an LTE prototyping system from *Signalion*, which consisted of a SDR hardware and software platform that was tailored to support basic requirements of LTE release 8 [84]. Unlike a standard-compliant base station, only the Physical Downlink Control Channel (PDCCH) was implemented in real-time to operate the link.

During the experiment, each base station recorded all received in-phase and quadrature (IQ) samples on a hard-disk. Thereafter, all data was stored centrally, because the evaluation of the novel concept was conducted using Matlab based offline signal processing. This way the development overhead for a real-time implementation could be kept low, yet proving the idea. Further, also the transmission parameters were close to LTE deployments by using LTE band 7 with a uplink frequency of 2500-2570 MHz and a downlink frequency of 2620-2690 MHz with a bandwidth of 20 MHz. Commercial available antennas from *KATHREIN* and power amplifiers (PAs) of up to 100 W output power were used. Each experimental site was co-located with commercial deployments from network operators.

Despite offering "only" basic link functionalities, the hardware platform consumed 12 rack units to provide space for 8 FPGAs, an embedded GPP system and 2x2 multiple input multiple output (MIMO) RF frontends, on the BS as well as on the UT side. Therefore enough rack space was needed and a UTs could only be operated using measurement trucks and custom-manufactured bicycles. This led to extensive field trials, where one measurement truck and up to 3 bicycles were operated to emulate 5 UTs. Still, the trials successfully demonstrated CoMP for downlink (DL) and uplink (UL) LTE and measured joint decoding rate gains of up to 200% [85] compared to conventional LTE detection schemes.

In the examples listed above, the FPGA platforms are bulky, energy-hungry, complex and development time consuming. Further, it is costly to employ those systems in more relevant environments outside of laboratories. However they were necessary to demonstrate and analyze the performance of algorithms and ideas beyond the state-of-the-art.

3.1.2 MAC - focused Testbeds

Here, the main interest is in performing experiments on local wireless networks with many clients. Thus, standard-compliant MAC-PHY platforms (e.g. WiFi or Zigbee-Cards) are commonly used to abstract away the challenges at the lower network layers.

One example is the TWIST testbed where 103 sensor nodes are distributed over several floors in a institutes building [86]. The intention of this testbed is to evaluate various localization methods. In comparison to the systems before, each node achieves a data rate of up to 64 Kbps.

The wilab testbed shown in Fig. 3.2 is another example, where very different devices are used in several facilities [87]. One of those is located above a clean room with 100

nodes distributed. Each node is equipped with a *Intel* NUC embedded PC that has one Bluetooth, Zigbee and WLAN card attached. Additionally, some are equipped with a SDR or LTE module. Further, remodeled vacuum cleaners act as moveable nodes with the same capabilities as the mounted ones. Those can be booked as part of an experiment and programmed to follow certain routes, emulating moving users. Part of the testbed is organized as portable units which can be deployed in different locations.

Figure 3.2: Reprinted from [87]: wilab facilities

Opposite to the PHY testbeds, MAC testbeds allow experiments with current PHY technologies, e.g. to stress its capabilities. In addition, the devices can be small and energy efficient to fit on robots or drones, or they can be used to form portable testbeds. Since, they are relative small they can be employed in different locations to test networks under different environments [88]. On the downside, they are limited to state-of-the-art technologies. Further, experience showed that testbeds deploying commercial LTE equipment often require remote-help from the vendor to configure and control them, because the respective interfaces are sealed.

3.1.3 Network - focused testbeds

Testbeds such as the Virtual Wall [89] emulate a massive amount of nodes without wireless connections to target evaluating network performance in general. That means many computers are attached to programmable network switches to perform experiments such as load balancing or broadcasting multiple streams efficiently. This is useful to stress modern network protocols, e.g. to understand the behavior of user datagram protocol (UDP) or transmission control protocol (TCP) in comparison to new proposals from companies like *Google*, such as Quick UDP Internet Connections (QUIC) which extends UDP with

TCP features. Typically, COTS network cards are used, but programmable FPGA based network cards [90] are offered too. Optimizing those network protocols is very relevant, since every connected application depends on it. The experiments in chapter 6 will show that the operating system (OS) and the used network protocol has an influence on the end-to-end latency.

The PlanetLab project [91] was a federation of up to 717 network focused testbeds across 48 countries in the world. This way networking experiments could be conducted in a universal way on very different installations. A summary of the experiences is given in [92]. This federation is continued in Europe as part of the OneLab project [93].

3.1.4 Generic testbeds

This section concerns testbeds which can be used for multiple purposes. The goal is to keep the structure as generic as possible, such that mainly SDR are connected to a server back-end. The most prominent testbed is the ORBIT facility [29] as a testbed containing all previous categories at Rutgers University and which is going to be extended with an outdoor testbed called COSMOS [94] in Harlem, New York City. There are three types of experimental nodes: Large and medium nodes on rooftops to resemble BSs and light nodes mounted on lantern posts or similar locations as UTs. So, the hardware set-up varies among the nodes, where some have additionally permanently mounted Millimeter-Wave (mmWave) RF frontends. Further, the experimental area spans over several blocks of buildings around the campus of the Columbia University and the City College of New York. This way the experiments are conducted in public areas similar to the EASY-C testbed. However, the COSMOS testbed is designed for real-time experiments [95]. Therefore, the backbone of the testbed is based on fiber optics to support data rates of 100 GBits per fiber and utilizes multiple fibers in parallel to aggregate all signals of all nodes. At least one dedicated fiber is required for all types of nodes using 10 GBits network interfaces. The server backend of the testbed consists of 64 servers with attached graphics processing unit (GPU) and FPGA cards. Each x86 based server has a dual 25 GBits network interface and each FPGA card a dedicated 100 GBits network interface. This way any of the raw IQ sample streams can be processed on any server.

Another example is the IRIS testbed [96] where *NI* USRP N210 devices are utilized to stream complex symbols to the network. The PHY layer consuming these symbols is computed on servers to allow a very generic signal processing, such as presented in [97]. This host-based signal processing keeps the flexibility for the sake of increasing the latency and reducing the throughput. The testbed is located in an indoor laboratory for easy access to perform experiments. A feature of the testbed is, that Python scripts can be applied via a web-page, which are then automatically executed based on the availability of the nodes.

3.2 Considerations

The authors in [98] present considerations prior designing an experimental prototype because a carefully planned design has huge influences on the later successful developments. For this reason, the National Aeronautics and Space Administration (NASA) introduced the technology readiness level (TRL) scale in the 1970s as a method to estimate the maturity of technologies. Before planning an experiment or even a testbed, this scale could serve as a starting point to estimate the ambitions for a certain technology demonstration. Further, the submission of project proposals employing prototypes and demonstrators, such as ORCA, might require an (self-)evaluation of the TRL level. The European Space Agency (ESA) took over the scale and adapted it for Europe:

TRL 1	Basic principles observed
TRL 2	Technology concept formulated
TRL 3	Experimental proof of concept
TRL 4	Technology validated in lab
TRL 5	Technology validated in relevant environment (industrially relevant environment in the case of key enabling technologies)
TRL 6	Technology demonstrated in relevant environment (industrially relevant environment in the case of key enabling technologies)
TRL 7	System prototype demonstration in operational environment
TRL 8	System complete and qualified
TRL 9	Actual system proven in operational environment (competitive manufacturing in the case of key enabling technologies)

Table 3.1: Technology readiness levels (TRL) considered for European projects [99]

Of course, the first two levels apply to the design of simulations to prove a certain concept. Already at this point, testbeds can serve by capturing the behavior of the later environment, for instance. As part of the EASY-C program, channel sounding measurements took place to capture the characteristics of urban channels, as presented in a overview in [100, p. 7-19]. Here, a special transceiver at the base station replayed sequences using the desired KATHREIN antennas for the BSs. A measurement car drove along the streets and recorded the signals using a specialized antenna and hardware platform. The captured raw IQ data was evaluated offline and allowed to estimate channel impulse responses as well as the angle-of-arrival of the signals. Those channels can either be applied to simulations instead of theoretical models or to real-time systems using channel emulators like the Colosseum Testbed or utilizing measurement equipment such as vector signal transceivers (e.g. *NI* PXIe-5644).

The third and fourth level applies to systems such as the mmWave demonstrator presented

in section 3.1.1. Here complex set-ups in laboratory enable the research of cutting-edge technology under assumptions such as that all clocks are tied to a common reference via cables.

The fifth and the sixth level are related to testbeds utilizing state-of-the-art technologies, such as WLAN or LTE where the experiment focuses on specific parts of the network, for example the resource scheduling. The further development of novel technologies into prototypes to higher TRL levels is very complex and demands large development teams, which is beyond the capabilities of academic research.

3.3 Use cases and Scenarios

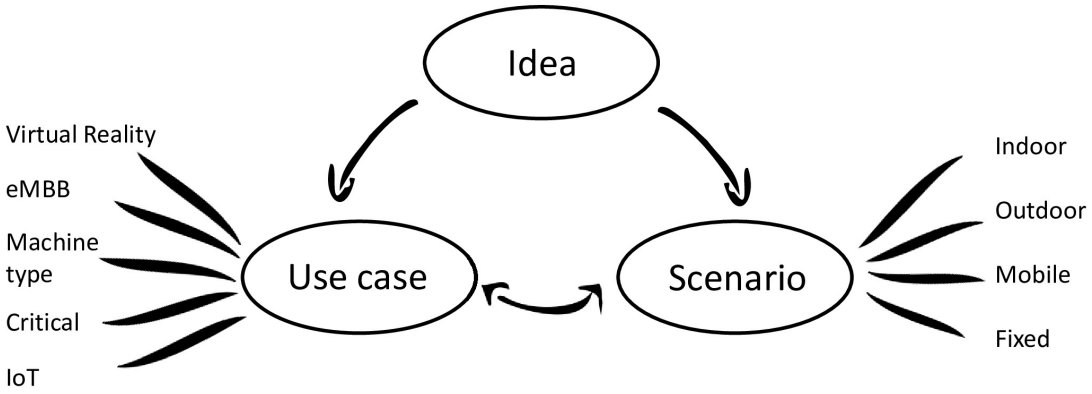


Figure 3.3: The application scenario is needed to define the requirements

After, the TRL targets are set, the use cases need to be defined. However, both application use case and scenario have to be respected jointly together as outlined in Fig. 3.3. For instance, outdoor experiments need a proper housing to keep the temperature in the operation range. Especially the characteristics of RF elements depend on the temperature.[1] Further, permissions to operate have to be acquired. Another difficulty regarding outdoor experimentation is that it is complicated to replicate those experiments. For example, in city scale testbeds, such as EASY-C or COSMOS, finding a consistent parking spot for a measurement van is difficult and the wireless situation changes permanently.

Particularly, very novel or complicated ideas need a reference point to validate the sanity of the experiment. The authors in [102] proposed therefore an indoor Doppler channel emulator instead of extensive field trial experiments using cars on public roads to measure the influence of the Doppler effect on wireless links.

Prototypes targeting the lower TRL levels are often evaluated in laboratory environments. Here the close range can lead to simplifications such as non-complex wireless channels and

[1] Manufacturers of control cabinets give good considerations for heating or cooling systems [101].

close distances do no need PAs, which bring potentially non-linearities. Further, the close range allows to use clock-synchronization via dedicated cables. Or all transceivers are cabled together for additive white Gaussian noise (AWGN) conditions, where a channel can be emulated in software such as in [81].

3.4 Requirements

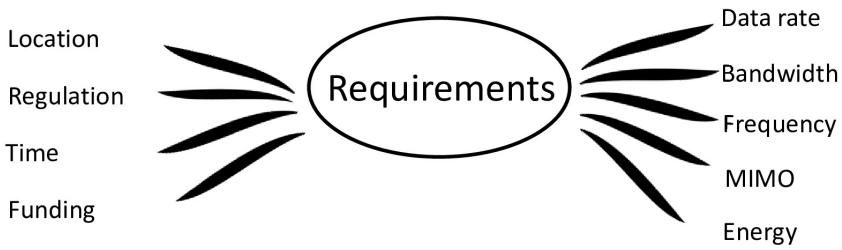


Figure 3.4: Based on the requirements, the methodology can be defined.

The technical requirements, such as bandwidth, frequency, number of antennas or energy are necessary to assess the measurement methodology and hardware platform. The technical and non-technical influences are depicted in Fig. 3.4.

The targeted bandwidth influences the data interfaces between the RF front-end, especially the digitizers and the signal processing platform and between the signal processor and the host platform. For example, the *NI* USRP E310 [103] offers a RF bandwidth of 56 MHz, whereas the data-rate to the host is only 10 MS/s. Either the samples have to be buffered or the bandwidth has to be reduced.

In particular demonstrators using high-frequencies in mmWave bands have several options. One option is that, a costly measurement-grade system is used, such as the mmWave system presented in [26] and [104]. Another solution is to use SDR or COTS platforms for sub 6 GHz bands using mmWave up and down-converter as presented in [105] or as a commercial device [106]. Here, the 2.4 GHz signal will be up-converted to the mmWave frequency and back respectively at the receiver. The authors in [107] show a similar set-up using baseband signals which are captured and evaluated offline. The first system requires specific knowledge and tool-chains, the other ones can reuse existing hard- and software platforms, but with certain limitations in respect to the achievable bandwidth or real-time capabilities.

Multi-antenna systems can be seen in a similar matter, either as a complete system as presented in [108] or by emulating multiple antennas through moving a single antenna system with high precision X-Y tables [109].

Energy consumption is an important topic for mobile experiments. On one hand powering complex systems used in [26] for mobile experiments requires heavy 50 kg batteries, on the other hand if power is available on-board such as in airplanes, it is unclear how stable and

noisy the power supply is, because it will affect noise sensitive components such as RF elements. Further, in case direct-current converters are used for cars, some are inverting the voltage levels, such that on the converted side the positive voltage is equivalent to the ground plane of the car and ground plane on the converted side is negative with respect to the ground plane of the car. This results in a short circuit in case the antenna of the transceiver platform touches the metal surface of the car as experience showed.

Due to the scarcity of the radio spectrum, many wireless testbeds are located in indoor environments [96], [110], or the outdoor section is available on request and/or requires the physical presence of the experimenter such as for the ORBIT [29] or EASY-C testbed. Hence, wireless experiments can be conducted either in the free industrial, scientific and medical bands (ISMs) or dedicated experimental licenses have to be acquired. For example, in Germany the application is handled by mail and needs to be renewed often to the national regulator [111]. Public universities are exempt of license fees. The United States are a step ahead, because trial licenses can be requested via a web-page from the Federal Communications Commission (FCC). Here, the applicant belonging to a university, company or other related institution has to draw the targeted area on a virtual map, fills in a electronic form and sends it to the FCC [112]. If there are not any complains from operators or other frequency users within a given time span, the applicant gets the permission to experiment for a given duration.

The goal of the EASY-C testbed was to demonstrate CoMP techniques for cellular LTE systems in the later deployed frequency range around 2.6 GHz to be as realistic as possible. This allowed a very realistic appraisal of the CoMP performance gains. On the other hand, all installations including the hardware components where limited to these frequency bands and got obsolete when the commercial operators started to deploy regular LTE base-stations in 2012. Reusing especially the RF parts was difficult and costly. In case frequency bands around 2 GHz could have been used as well, the operation of the testbed would have been possible until 2016, when commercial satellite operators took over the frequency licenses. This way the time span for experiments would be doubled.

EASY-C experiments were organized in measurement campaigns due to the administrative overhead because the testbed spread over the inner city of Dresden. The operation required to have manually operated UTs, e.g. by driving a measurement van, because permanently mounted UTs where not available. Permanently mounted UTs could have been utilized as reference locations, however the evaluation of CoMP required moving nodes. So, experiments where conducted from time to time and required several researchers. One example is the urban channel measurement campaign together with TU Illmenau [100, p. 7-19].

The bureaucracy behind testbeds should not to be underestimated. It took 5 years to realize the structures presented in chapter 5, because responsibilities are often unclear in public institutions. The indoor part of the testbed required fire protections simulations and certificates. Funding the testbed devices required several sources, because European

projects could finance personal costs but not hardware due to the depreciation costs, such that federal support was required as well. Further, experience showed that the European commission wants to establish (paid) testbed-as-a-service offers similar to the Fed4Fire project[113], that requires a considerable amount of human resources. This is difficult to realize for academic institutions and converts research platforms into businesses. Other examples for bureaucracy are procurement directives for public institutions, where the acquisition of costly (>€5000) equipment can take half a year, although the devices are in stock at the manufacturer.

3.5 Methodology

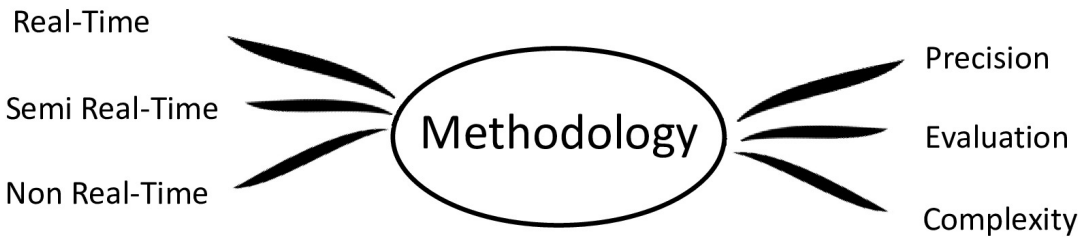


Figure 3.5: Before selecting a actual hardware and software platform, it is necessary to assess the complexity of the targeted demonstrator.

The main reason to assess the complexity of the targeted demonstrator as presented in Fig. 3.5 is the length of the development design cycle, which depends on the selected hardware and software architecture shown in Fig. 3.6. Here, the targeted performance indicators for the evaluation define the conditions. For instance, standard compliant network testing requires that all timings are respected. In a LTE system for example, a hybrid automatic repeat request (HARQ) requires that the acknowledgment (ACK) is sent after four sub-frames [114], i.e. after 4 ms. Contrary, a 802.11g WLAN system requires an ACK to be sent after the Short Interframe Space (SIFS) [115] which is 10 μs long [116].

So, experiments studying or using higher network layer protocols need real-time capable platforms or choose a different strategy. In EASY-C the evaluation of CoMP was conducted using offline evaluation for instance, such that the novel algorithms did not need to run in real-time.

Hence, the different architectures can be grouped into

- **Real-time** operation, where all algorithms meet the application requirements, e.g. LTE open source implementations [117]
- The opposite is **Non Real-time** operation, here the IQ samples are captured and evaluated afterwards, e.g. channel sounding equipment [118]

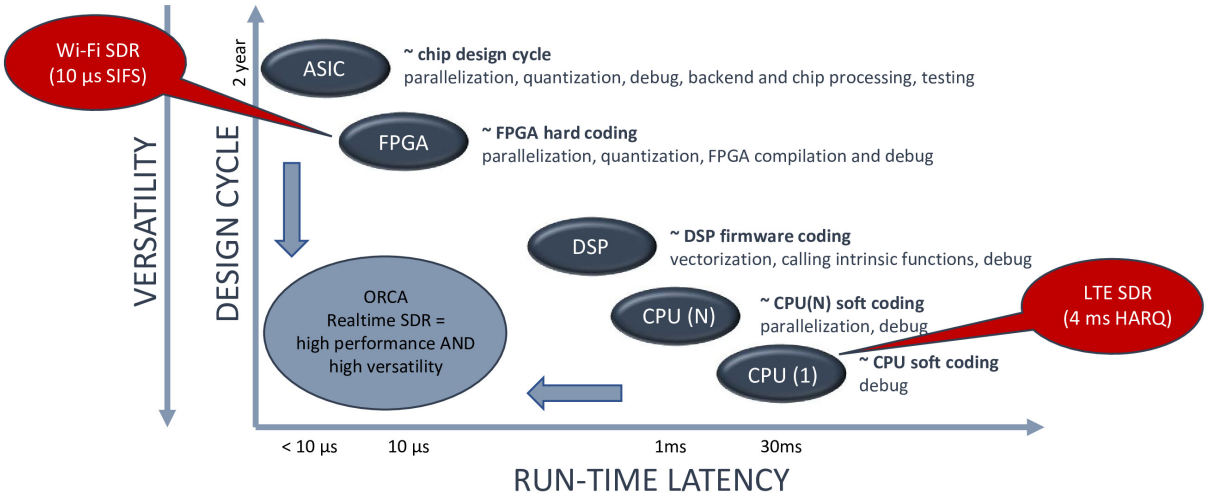


Figure 3.6: Adapted from [31]: Design cycle of base band processors for wireless networks

▷ **Semi Real-time** is a mixture between the other categories. Some functionality is implemented to meet the timings but data is evaluated offline too. For instance, in the EASY-C testbed real-time operation was implemented for the control channels, but data samples are captured too for offline processing.

3.6 Hardware Platform

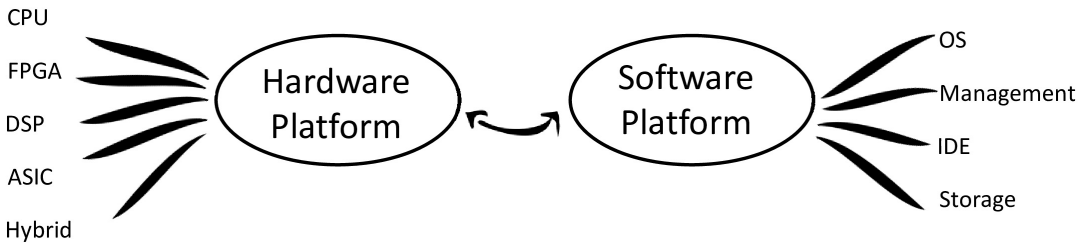


Figure 3.7: The hardware and software platform are dependent from each other and define the testbed capabilities.

Although COTS chip-sets are enabling wireless research as well, the focus of this section is on SDR since they allow a maximum of flexibility. Further, these capabilities are used in later sections of this work. SDR platforms exist in various fashions, where both hardware and software influence each other. Fig. 3.8 and this section presents four popular architectures. Common to all is that a computing platform is connected to the RF front-end and that the operation of the SDR is controlled by a host computer. The majority of SDRs is using FPGAs as signal processors, although digital signal processors (DSPs) could perform signal processing tasks too [9]. The connection to the control computer can be achieved by different options with different latency and throughput results.

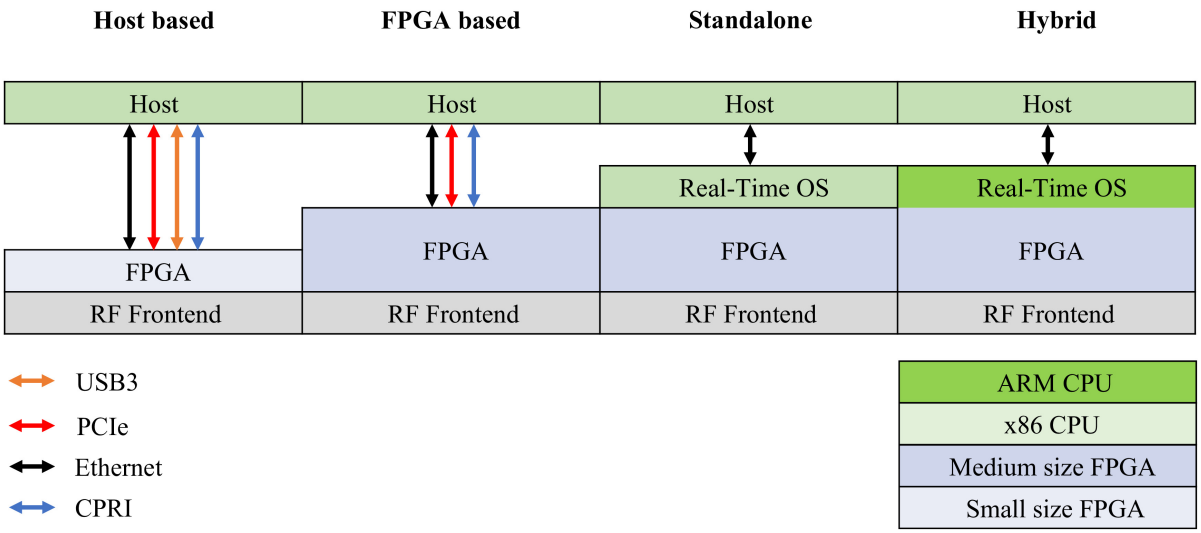


Figure 3.8: Overview of different hardware platforms.

3.6.1 Host

Smaller sized platforms such as the USRP B205 mini from *NI* perform tasks like sample-rate conversion in the FPGA. The communication related processing is carried out on the x86 standard computer. So, it is the most flexible and universal option. Modern central processing units (CPUs) architectures and operating systems offer various signal processing opportunities. The disadvantage is that the latencies can be long and the throughput low. The emulation of wireless systems with strict timing requirements such as WLAN is difficult to achieve, however the signals can still be received [119]. In contrast, standard compliant LTE signal processing is possible, since an acknowledgment has to be calculated within 4 ms. So, different LTE implementations are available, such as OpenAirInterface [117] or srsLTE [120].

Amongst many SDR platforms, the USRP series is popular, because of the large open source community supporting it. The smallest platform is the credit card sized USRP B-series using USB to connect to the host [121]. The probably most popular platform is the USRP N200 series which is attached to a host system via Ethernet. However, there are various other platforms available as well, the simplest form are terrestrial digital video broadcasting (DVB-T) USB receivers based on *Realtek* RTL2832U chipsets, which can forward the IQ samples to the host [122].

3.6.2 FPGA

Contrary, implementing complete transceiver on FPGAs introduces certainly long development times. However, FPGA can provide the high throughput and low latencies in terms of performance. *NI* provides with the USRP X300 series a very popular platform. It can be connected to the host computer via PCI-Express or Ethernet. This platform is

used in very different roles. For example, the flexible transceiver presented in this work is implemented on a USRP X310. But it can also be used as a channel emulator for example to emulate vehicular channels [123].

Noteworthy are that development boards from FPGA suppliers such as *XILINX* can be extend with RF front-ends as well, e.g. the *Analog Devices* FMCOMMS boards.

3.6.3 Hybrid

Hybrid platforms try to bridge the gap between the two previous cases, where a GPP system is connected in-between the host system and the FPGA. One option are PCI eXtensions for Instrumentation (PXI) based systems. Here, one or several FPGAs and other I/O cards are connected via a PXI bus. This bus is standardized by the PXI alliance formed by measurement companies such as *NI* or *Keysight* and enfolds a wide range of modules and extension cards. The GPP controller of such a system can run a real-time operation system to orchestrate the signal processing operations.

The mmWave testbed described in section 3.1.1 is using that functionality. The MAC layer is running on a *VxWorks* operating system and controls the resource scheduling and beam-steering. The low-latency link to the PHY layer is provided by the back-plane of the PXI chassis that connects all 6 FPGA cards with each other.

Massive MIMO testbeds use the same concept with multiple USRP X310 as input devices and to perform pre-processing [78]. The backplane consists of multiple PXI chassis with additional FPGA cards to gather the symbol streams and to do the final signal processing. The same architecture can be used as a channel emulator as well [82].

A small form factor version of this system is the USRP 2974 where an x86 PC platform is embedded into one chassis with the USRP X310 SDR. Here, a PCIe bus connects both components and a MXI link allows to attach further USRPs or other FPGA cards. Another similar option is the *Xilinx* ZYNQ platform, where an ARM processor and an FPGA fabric are located on the same die. This allows very low latency (1.4 µs) data exchange between both in comparison to the other platforms [124]. However, the ARM is not as powerful compared to the embedded x86 PC in the the USRP 2974.

All those options allow interesting split ups where the PHY signal processing remains on the FPGA and the more flexible MAC calculations are executed on a GPP platform. Further, in case specific algorithm such as new decoding schemes are under test, they could be offloaded to the host, whereas the rest of the necessary signal processing is kept on the FPGA.

3.6.4 ASIC

Custom chip-sets or application-specific integrated circuits (ASICs) are the most advanced platforms and most close to the final products. Due to the related implementation efforts

[125], those are mostly provided by hardware vendors. In the case such an approach can be utilized for experiments, than the performance with respect to energy efficiency, throughput or latency can be much better than the previous programmable solutions.

Another possibility is using Multi-Processors System on Chip (MPSoC) designs such as the Tomahawk [126] or *Kalray* MPPA Manycore, because they are on the one hand programmable, but on the other hand already tailored for specific instructions. This allows to reduce the energy consumption of 71 % compared to GPUs [127]. Therefore, *THALES* evaluated the *Kalray* platform as part of the European eWINE project [128] and brought in the results into the standardization groups such as the Wireless Innovation Forum. One drawback of this very customized solutions is the limited amount and speed of the input and output ports for the data exchange. So, a SDR could not be directly attached to the *Kalray* platform and always required a host system to transfer the data, via PCIe or Ethernet for instance.

3.7 Software Platform

There are three types of software platforms relevant for wireless prototyping. The first one providing a management work-flow to control testbeds or experimental data. The second type are development frameworks or tool-chains to program the signal processing modules. Thirdly, ready-made applications implement a given wireless standard, with the aim to be open such that they can be extended.

3.7.1 Testbed Management Frameworks

The ORBIT facility is controlled by the Orbit Management Framework (OMF) framework, with the intention to reproduce experimental results [129]: ”Compared to other scientific research fields, such as physic sciences, the networking field so far has not developed a culture of rigorous peer verification of experimental results. This is mostly due to the fact that even if the same or a similar experimental infrastructure is available (i.e. through a federation of world-wide testbeds), there is currently no unambiguous way to describe and instrument an experiment enabling others to repeat it.”

The goal of OMF is to come up with a generalized instruction set of commands and parameters to a) setup the testbed environment for an experiment, b) control the involved nodes, c) start the experiment, d) log the results, e) store the results in a database and f) finish the experiment and freeing the respective nodes. NEPI developed by Technische Universität München is offering a same concept as OMF [130].

In both cases, a messaging protocol, e.g. XMPP similar like ”Whatsapp” for machines, abstracts the communication between the experimental controller and the nodes or resources such as network switches. Finally, a script orchestrates the experiment. Either

this script is manually written or created by tools like jFed [131], where the whole experiment can be programmed in a visual work-flow. Those testbed control frameworks are very helpful if many nodes have to be controlled simultaneously. Further, the frameworks store results in a generic way. This is very suitable for higher layer network experiments, where achievable data rates or latency results are studied. Those frameworks also provide higher layer parameters and instructions, such as start a given application with given parameters, e.g. as presented in [132]. This works well if the application or communication system under test is well advanced in the development process. For instance, a WLAN driver under test needs to be invoked using different parameter settings to study the behavior and performance of the network.

Contrarily, specialized tool flows such as in PHY oriented testbeds like for mmWave or EASY-C are very difficult to control. Either because the specialized hardware platforms do not allow an external control of the parameters, the implementation is not matured into a self-containing application or it is difficult to achieve because of missing documentation. For example, for the EASY-C Testbed control, applications were developed which emulated mouse clicks on the configuration tools provided by the hardware vendors. Here, it is difficult to justify those time-consuming adaptations to the control frameworks versus manual operation. Further, the created amount of data might push already the hardware to its limits, for instance the mmWave system [104] with its 1.5 GS/s digital-to-analog converters (DACs). In case this data is written to a generic database, the storage capabilities have to be well prepared.

3.7.2 Development Frameworks

The most popular open source programming environment for SDR is GNU Radio [133]. The idea behind is, that each individual function, such as presented in the next chapter, is programmed in the C/C++ language and runs on the host computer. These functions are organized as libraries of modules which can be interconnected in a graphical way to keep an overview and to form a signal processing application. This way complete transceiver chains can be realized, but also programs to perform channel sounding. The underlying SDR hardware is abstracted such that different devices from different vendors are supported by the same application. This approach is limited to GPP architectures only and therefore sets constraints with respect to latency, throughput and accurate timings. To overcome this limitations, RF Network on Chip (RFNoC) was introduced to allow parts of the signal processing to run on FPGA as well. It is using the AXI (Advanced eXtensible Interface) infrastructure of *XILINX* FPGAs to route the data between different FPGA and Host modules.

Besides the open source software, LabVIEW [134] offers a proprietary solution to program SDR without being limited to. LabVIEW has been developed since 1983 and offers a data flow representation where signal processing elements are interconnected to form

complete applications. The difference to GNU Radio is that it is fully graphical without textual source code. This implies that basic programming elements such as numerical or Boolean operations are connected with wires similar to schematics used for digital circuit designs. The operations are grouped as Virtual Instruments (VIs), which are compiled prior execution to offer a faster processing performance than interpreted languages such as MATLAB. VIs can be run on Windows or Linux Real-Time OSs as well as on FPGA. In the latter case, a VI is translated to VHDL code and processed by *Xilinx* tool chains to generate the FPGA bitfile. Although LabVIEW was developed primarily to automate test and measurement equipment, it provides a complete tool flow from designing to debugging host or FPGA applications. In contrast, GNU Radio needs external programs for the FPGA developments and pure FPGA designs need matching applications running on a host computer for control.

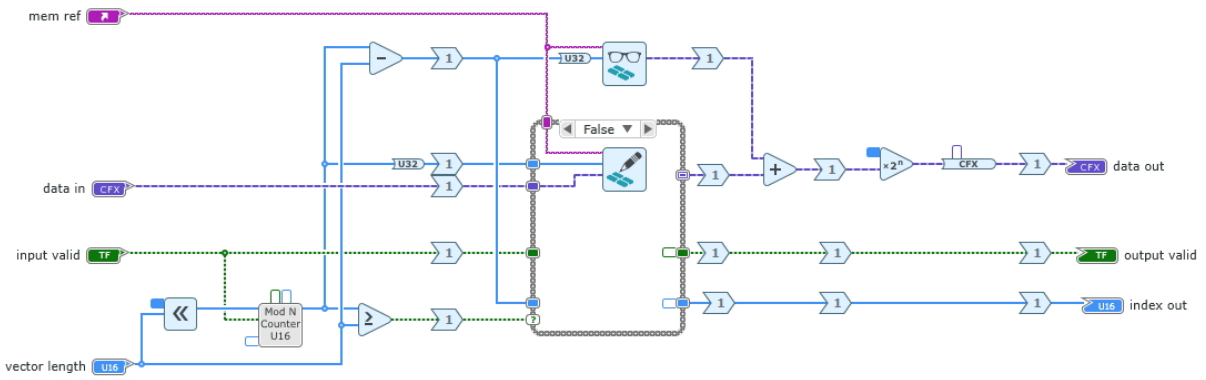


Figure 3.9: LabVIEW FPGA source code for averaging two vectors, which can be seen as graphical VHDL where every sample has to be respected.

Although LabVIEW already offers a more simplified entry to FPGA programming than VHDL, there is even a further simplification called the Multi-Rate-Diagram. One example is printed in Fig. 3.10, which shows the whole generalized frequency division multiplexing (GFDM) modulator fixed to a given waveform configuration. The Multi-Rate-Diagram is designed to process streams of data rather than considering individual samples. Thus, the programmer specifies how many samples each block needs to consume and the compiler realizes this with a hand-shaking logic. The resulting machine-code will not be optimized or flexible, but it allows a quick assessment of the capabilities of a given algorithm in real-time. This concept exists for other development tools as well, for example as in *Mathworks* Simulink.

The third major development workflow with respect to SDR is using the *Xilinx* tool chains [135]. This enables to use all features of the hardware platforms, in contrast to GNU Radio or LabVIEW, where an abstraction layer already simplifies the hardware interactions. However, highly specialized developers might be needed, especially to interface the RF modules.[2] Popular programming languages are VHDL and Verilog, but also the more

[2] Experience showed that providing a RF driver for a standardized USRP takes half a year.

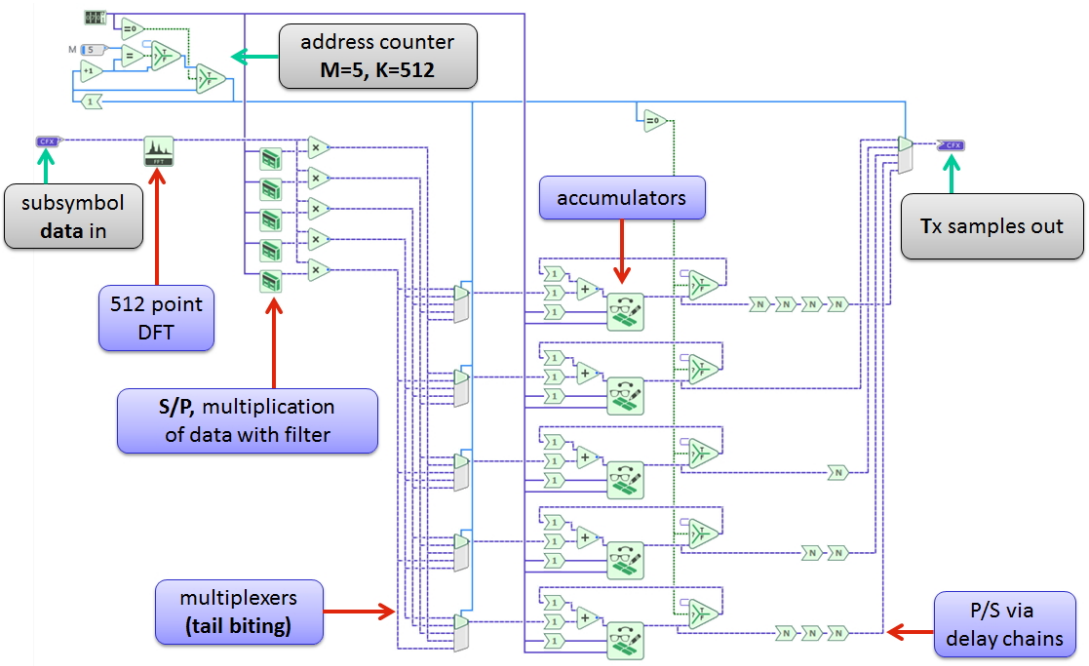


Figure 3.10: The LabVIEW Multi-Rate Diagram considers streams of data instead of individual samples.

abstract work flow myHDL [136]. Using high-level synthesis tools [137], such as *Xilinx* Vivado HLS, offer more options for hardware implementations using C like code as input.

3.7.3 Software Implementations

Besides developing applications, experimenters can also use ready-made applications, like the *NI* Application Frameworks for LTE and WLAN using the LabVIEW tool flow. They provide most important parts of the LTE or WLAN standard. Furthermore the signal processing is kept on FPGA to respect the timings and provide the realistic throughput. The limited standard compliance leaves space for own developments, e.g. to add a GFDM modulator as presented in [138].

Pure GPP and open source LTE implementations such as the Open Air Interface (OAI) provide full standard compliance, such that even commercial modules can connect to. For instance, in [6] the measurement set-up is using a commercial LTE dongle to connect to the LTE base station created by the OAI. In addition, a CR utilized non occupied spectrum leftover by the LTE transmission. During the experiment the achieved good-put at the LTE dongle was measured under the impairments created by the secondary CR transmission. As a result, the authors in [6] show that a GFDM based CR applies lesser interference to LTE than a conventional orthogonal frequency division multiplexing (OFDM) based CR. The main reason are the reduced out-of-band (OOB) emissions of GFDM as shown in Fig. 2.4. Since a commercial LTE dongle was part of the set-up, the standard compliant OAI allowed a very realistic study. However, full standard compliance

easily leads to complicated software structures, such that own extensions are difficult to achieve.[3]

3.8 Deployment

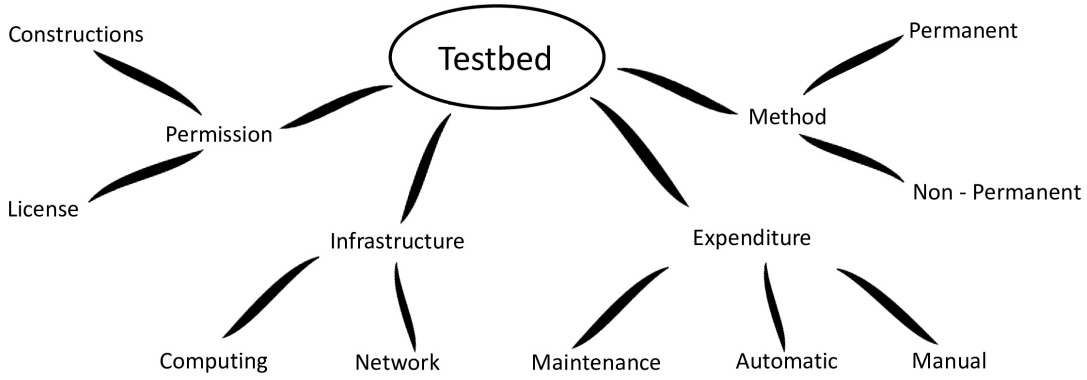


Figure 3.11: The testbed deployment combines all previous work but requires constant attention.

Although, the deployment is summarizing all the previous work but still requires attention. The maintenance overhead of big testbeds such as wilab or ORBIT is shared by a university department or institute. The EASY-C testbed in Dresden had a three person team to operate it but also to add functions and features to the system for experiments. Additionally, the measurement campaigns in EASY-C required one person for each UT to conduct experiments in the city. Within this work developed Online Wireless Lab (OWL) testbed [139] requires at least one person for operation with occasional support, especially for the network and computing infrastructure. Testbed software frameworks such as OMF can reduce the amount of manual inputs and allow unsupervised experimentation,but contribute with constant overhead, e.g. bug fixing, updating software components or maintaining the documentation.

Several of the various testbed deployments located[4] in Europe and the United States support federations such as Fed4Fire [113]. Fed4Fire itself is a series of European projects to develop and maintain the Fed4Fire web-based framework, supporting testbed operators by providing a framework to handle users, experiments and offers visibility. This framework consists of a overview page, which constantly monitors the status of the individual testbeds. Experimenters can register at the web-page to use any of the compatible testbeds. Further, jFed is developed as an entry point prior using a testbed [131]. The user can configure the testbed for the intended experiment using a graphical interface. Afterwards, jFed creates a XML based description that is sent to testbed

[3] It can take already three days to set-up a working Open Air Interface instance without any extension.
[4] A detailed overview over the locations can be found here: `https://fedmon.fed4fire.eu/map/`

management server along with the user certificate. This way a testbed does not need a own customized control interface.[5]

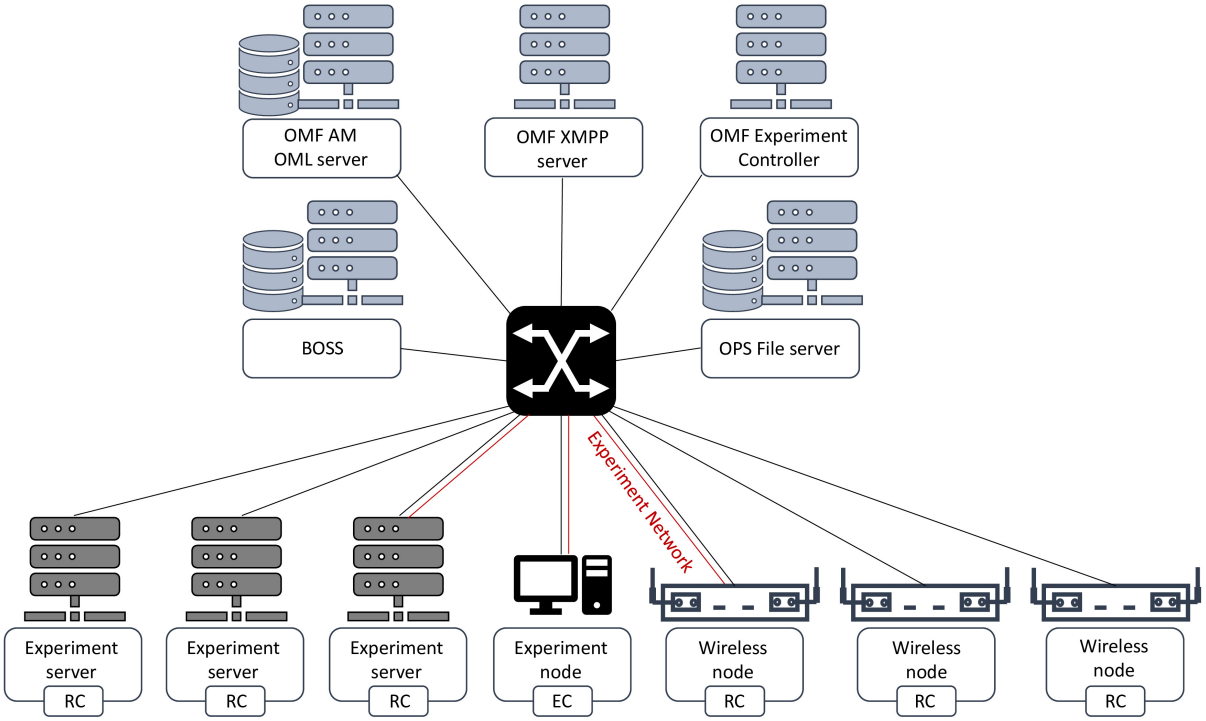


Figure 3.12: Adapted from [87]: w-ilab network elements including the OMF framework.

Many (Fed4Fire) testbeds, such as wilab.t or ORBIT, have a server and network infrastructure to control and provide all functionalities. Fig. 3.12 shows the structure of the wilab.2 testbed. Typically the testbed controls and the individual user experiments are using virtually separated network connections. The aim is to minimize the influences on each other, e.g. in case the experimenter applies a misconfiguration to the network elements, and to protect the intellectual property of the potentially different user groups. The w-iLab.2 testbed uses virtual local area networks (VLANs) for the user separation and the control backbone. The predominant protocol to interact with all components is the Secure Shell (SSH) protocol in those testbeds.

Fig. 3.12 further indicates the server back-end consisting of several (virtual) machines of the w-iLab.2 testbed [87]. The main control server is called ”BOSS” and provides services such as internet access, controls the network and distributes user-specific software for the individual nodes. All data is stored on ”OPS”. The other three servers are related to the OMF framework. The OMF Aggregate Manager (AM) server reads the jFed created XML specification for the testbed and applies it via ”BOSS” onto the testbed infrastructure. The other two servers ”OMF XMPP” and ”OMF Experimental Controller” are realizing the remaining OMF functionalities. The experiment servers and nodes are dedicated for

[5] A reservation platform, to ensure that the devices are available, is not provided by Fed4Fire and needs to be realized by the testbed operators.

experiments and do not take over testbed related tasks.

Although it may be implied, a testbed is not necessarily a permanent installation. IMEC created the portable testbed [88] as a small form factor copy of the wilab testbed. The aim is to have a solution for relative quick deployments such as in industrial premises. This allows measurement campaigns and experiments in very different environments to enable comparisons among them.

3.9 Discussion

As indicated earlier, many testbeds focus on a specific use case, such as increasing the data rate via using multiple antennas or using ZigBee devices for localization experiments. However, also the multi-purpose and generic testbeds limit potential experiments due to given design decisions, such that there is not a generalized way. For example, the IRIS testbed is located within one laboratory room and based on a Linux architecture. As a result, the transmission conditions are potentially simple, such that they might limit the significance of the experiments in terms of the channel conditions. In addition, the OS architecture makes it difficult to support a Windows based tool-flow such as LabVIEW.

This also applies to the ORBIT testbed at RUTGERS university, which offers already a very customized work-flow to support various experiments. However, the replication of this very advanced work-flow for the architecture presented in chapter 5 failed. The reason was that LabVIEW was considered as development tool as well and required Windows as OS. The provided guidelines and examples for the usage of Windows in the ORBIT testbed were obsolete because a) the presented underlying server OS was outdated, b) the used OMF framework version was outdated and c) the Windows system version was outdated as well. Further, the documentation was not sufficient enough to understand how to adapt Windows to the advancing versions of OMF. The issue here is that OMF controls the whole flow of an experiment, from the provisioning of the OS, starting all programs, starting the experiment and finally collecting the results. Although this provides a very powerful experimentation tool, which can even replicate an experiment in another testbed, it only works as long as the experiment sticks to a given development method.

LabVIEW itself as a development tool allows a very unique way for prototyping new ideas. Its visualized programming structure is especially suited for algorithms. The implemented algorithms are runnable on Host, Real-Time OS or FPGA. In addition, it can connect very different research domains due to many extension cards or libraries and still keeps the block diagram like programming and the very intuitive data flow. On the other hand, it is very restrictive because of the commercial orientation. If LabVIEW does not support a certain feature, it is very hard to overcome that limitation. Further, supporting multiple research domains also increases the amount of documentation, which is sometimes very sparse in the different LabVIEW versions. Especially, the FPGA tool-flow is very unique,

yet very difficult to understand and sparsely documented. So, a developer has sometimes to trying out all variations to understand certain behaviors of LabVIEW. This reduces the development time savings because of the nice work-flow, drastically.

The COSMOS testbed is utilizing the SDR platform USRP 2974 with various RF front-ends and is build on outdoor roof-tops in New York city. This provides a very realistic environment for cellular experiments and the trials have to take place in public space which raises the significance further. On the other hand, finding a consistent parking spot for a measurement van might be difficult. So having a constant reference point, for instance to ensure the sanity of the experiment, is difficult. Further, 5G networks and beyond focus on machine-type applications that make use of the low latency capabilities. However, typical applications such as autonomous robots or drones are difficult to test, because the relevant experiments will take in public areas.

In addition, city scale testbeds such as EASY-C or COSMOS are seen as representative environments for wireless experiments to justify the maintenance expenses. Yet, Dresden or New York City[6] are very unique city's, so it is unclear whether the results are transferable to other locations. Similar like camera pictures of a building or a landscape, although some elements are comparable everywhere, there are still major differences, e.g. in the propagation conditions especially if mmWave experiments are conducted. Further, the selection of the location of the experimental nodes is limited to local permissions and therefore they are subject to random restrictions.

Another unresolved question is how to store a) experimental data, b) the firmware/software version to create that data, including OS, drivers and the particular software, c) the software to analyze the measurements, d) a complete description/documentation of the experiment and maybe a simplified presentation of it. Traditionally, each of these four domains would have different mechanisms to keep the developments. However, they have to be linked to reproduce the results. For instance, a replication of one experiment would require that the complete software stack, including outdated OS and drivers have to be deployed on modern machines, for 10 years. It is assumed that specialized hardware components are still available for experiments, this is the case for instance for the USRP platform which is produced since 2005 [141].

The data captured in the EASY-C project are kept in files of various formats and sizes (up to 26 GB per file), despite having a similar content of raw IQ symbols. Of course, they are stored in different flavors depending on the measurement equipment or potential compression. However, this prevents experimentally playing and combining of the elaborately obtained measurement results. The IEEE therefore started the initiative to store experimental results on IEEE DataPort cloud servers. Locally hosted databases

[6] The area where many EASY-C trials took place is designed in the 1950s and consists of buildings of the series "Altneubau"[140] in a typical, regular layout of Eastern Germany, compared to the much denser, taller and varying buildings in New York City.

such as MongoDB could be an alternative, especially if intellectual property rights have to be respected.

3.10 Conclusion

Selecting a (wireless) prototyping solution depends on several factors with very different impacts on the outcome:

1. The TRL classification allows to assess the expectations versus the efforts required for the prototype.
2. These defined preconditions determine the requirements for the hardware/software platform and the respective trade-offs for the evaluation later, e.g. real-time or non-real time.
3. The development expenses are driven by the selected platform. However, the selection process is complex due to the amount of available options and their respective (hidden) implications.
4. Finally, the testbed encompasses all efforts to demonstrate the achievements, but solicits attention due to the necessary infrastructure and maintenance.

Some of the factors are predefined, for example if the hardware is already given or the choices are limited by the funding sources, e.g. by an industrial project. Other conditions are random due to structural situations. Changing those surrounding circumstances requires patience and even a bit of luck, such that a fitting funding framework program is set-up to finance required modifications.

Interesting is that the efforts to maintain and operate testbeds, according to experience rather increase than reduce. The authors in [92] state that the maintenance overhead was already an issue in 2001 for the networking testbeds in PlanetLab. Nowadays, for instance, the COSMOS testbed extends those networking/cloud infrastructure with multiple SDRs across a huge public area. Another example is that in former times a (control-) web-page was often a ”simple”, straightforward PHP implementation. Now it has to follow standards such as GDPR [142], EN 301 549 for accessibility [143], while keeping the security under control.

The biggest challenges however are to find out which hardware but even more which software methodology should be followed. There are uncountable number of hardware devices or components available where many are supposed to be compatible to each other, but in practice it turns out they are not. For instance, utilizing modern switches requires to find a suitable SFP Module that connects the switch with the medium, e.g. fiber or copper. The price range of such modules spans over two magnitudes and still they are not

accepted by all network components.[7] More options are available with respect to software, where applications exist in various forms, versions or flavors. Sometimes the best solution has to be found out by simply trying all options.

As a conclusion, the aim of the following chapters is to show one strategy how real-time wireless networks can be prototyped. The next chapter focuses on a FPGA design for the USRP X310 SDR and the following chapter discusses the developed testbed infrastructure.

[7] Each SFP module is shipped with a dedicated type number in the firmware and the network element decides if it wants to accept it or not. There are methods to set this type flags correctly. Afterwards the module works because everything else is standardized.

Chapter 4

Flexible Transceiver

After understanding the design flow the key challenge is now to implement a digital baseband processor. The architecture itself is a problem independent of the hardware and software platform. Hence, this chapter describes the basic architecture design framework of a flexible waveform processor.

Especially, the fifth generation (5G) of mobile cellular systems needs to support various services that can have very different requirements, e.g., ultra-low latency in the Tactile Internet scenario, low power consumption in large scale machine-to-machine communication and low out-of-band emission in dynamic spectrum access. A physical layer (PHY) with unprecedented flexibility will benefit users because it allows multiple applications to be addressed efficiently by reconfiguring the modem. From the implementation perspective, it is challenging to achieve flexibility, because this typically entails increased computational complexity of the modulation and demodulation algorithms. Such a flexible PHY necessitates a proven signal processing concept with many degrees of freedom, which is introduced by generalized frequency division multiplexing (GFDM).

Since, the goal is to support low-latency use cases, this signal processing framework targets a field programmable gate array (FPGA) based Software-defined radio (SDR) platform, because of the better throughput and latency performance. The USRP X310 is chosen because it supports any frequency up to 6 GHz and the XILINX Kintex7 FPGA offers enough resources, such that a Long Term Evolution (LTE) [144] or wireless local area network (WLAN) [145] PHY implementation fits into one device. The LabVIEW Communications System Design Suite is used because it enables the rapid development of real-time prototypes [146]. The relevant hardware specifications are given in Tab. 4.1 [147]. The contributions in this chapter are based on [148], [149], [150], [151] and [46].

4.1 Signal Processing Modules

This section contributes with a strategy to implement and demonstrate the feasibility of this flexible scheme. An overview of all signal processing modules is given in Fig. 4.1 and discussed in the following sections in detail. The colors indicate which modules are discussed jointly. Each section is accompanied by a table which summarizes the parameter and FPGA resource utilization. Further, an overview of the FPGA implementation is given in the respective figures for each section. The latency values are defined as the time between the first sample entering a module and the first sample leaving it.

The objective of the development, besides implementing a complete transceiver, is to create a library of signal processing blocks which can be reused for further experiments. Therefore, an approach to handle the data and parameters between them is presented. In addition, many blocks in the design follow a handshaking protocol, where a signal processing unit is indicating to the previous unit if it can accept more data samples. This way samples cannot get lost during the processing and a lower radio frequency (RF) sampling rate slows down the signal processing by back-pressuring [152].

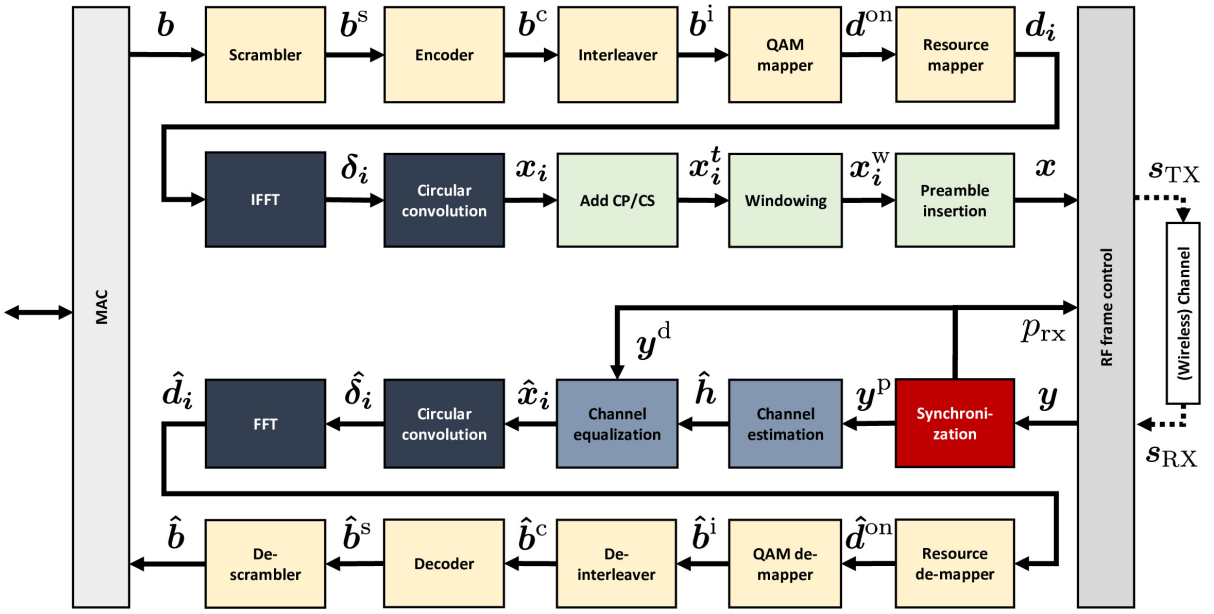


Figure 4.1: Block diagram of relevant signal processing modules in the transceiver. The colors indicate which modules are discussed jointly.

As presented in the previous chapter, a SDR is connected to a host or control computer, which transfers the payload to the FPGA via data input and output buffer. So, the first assumption is that the host computer ensures that the payload is already adapted to the packet lengths of the data frame at the transmitter and it will assemble the packets together at the receiver.

The second assumption concerns the bit precision used inside the digital signal processing. The presented implementation can cover very different parameter settings. The maximum

configuration values cover LTE like settings and the smallest values target Internet of Things (IoT) use cases, such as indicated in Tab. 2.1. Therefore, the requirements for the word length change accordingly. For a fixed implementation such as the *National Instruments (NI)* LTE Application Framework the bit precision can be optimized using different tools inside the LabVIEW [134] integrated development environment (IDE). Here, the required word length strongly depends on the parameter selection and since the aim is to enable wireless prototyping, the bit precision can be adjusted for the individual module via separate parameters. In general, the data format is 32 bit wide, having 16 bit for the real and 16 bit for the imaginary values, but some modules internally extend the data processing up to 64 bit. Thus, the results need to be scaled down using a dedicated scaling parameter.

The third assumption is, that the control of the RF elements is handled by the USRP driver. This includes parameters such as setting the center frequency, adjusting the bandwidth or the power level of the amplifier in the SDR frontend. Both the analog-to-digital converter (ADC), digital-to-analog converter (DAC) generally work with the maximum sample rate. A non-disclosed algorithm up- or down-samples the signals to the desired sample rate inside the driver. The arrangement of the modules in the RF loop are depicted in Fig. 4.2.

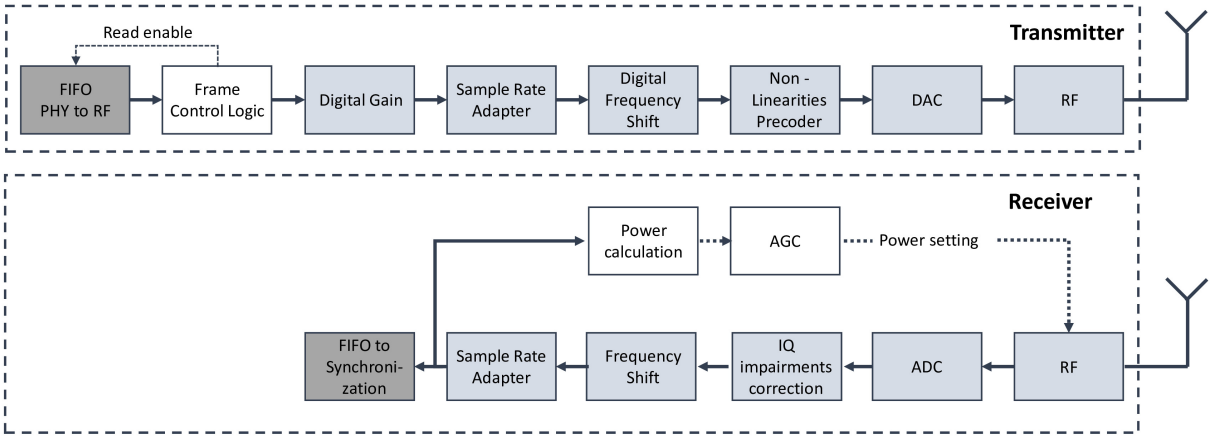


Figure 4.2: Block diagram of relevant signal processing modules in the RF driving loop. The light blue signal processing blocks are taken from a LabVIEW library.

4.1.1 MAC interface

The first processing block is a simplified MAC interface, which is responsible for adding a minimal medium access control (MAC) header. Table 4.3 summarizes the control parameters. The resource utilization cannot be given because the implemented logic is spread over several modules. The MAC interface is designed for the Low-Latency Communication (LLC) frame configuration depicted in Fig. 2.5 and follows the principle depicted there.

Parameter	Value/Range	Comment
BW_{TX}	up to 160 MHz	Sample Rate
$f_{\mathrm{c,\ TX}}$	10 MHz to 6 GHz	Center frequency
DAC resolution	16 bits	
BW_{RX}	up to 160 MHz	Sample Rate
$f_{\mathrm{c,\ RX}}$	10 MHz to 6 GHz	Center frequency
ADC resolution	14 bits	
p_{TX}	0 to 31	Uncalibrated transmit power
p_{RX}	-31 to 0	Uncalibrated receive attenuation
c_{Data}	200 MHz	Data clock speed
Daughterboard type	2 x UBX	one full duplex RF frontend each

Xilinx Kintex7 410T FPGA Resources		
Register	12% 61085 of 508400	
digital signal processors (DSPs)	18% 272 of 1540	
Block Memory	13% 103 of 795	
Look-up-Tables	12% 30141 of 254200	
Slices	25% 15587 of 63550	

Table 4.1: Parameter of the USRP hardware and driver / streaming example based on LabVIEW NXG 3.1

PLCP header	MAC header				Payload	CRC	Encoder stuffing
PHY Configuration	Frame ID	Direction	Sender ID	Receiver ID			
5 Bytes	5 bit	1 Bit	5 Bit	5 Bit	x Bytes	2 Bytes	1 Byte

Table 4.2: Configuration of the frame. The PLCP header might be added as part of future works and would contain information about the waveform configuration relevant for decoding the payload data.

The MAC header contains an identification (ID) byte to identify the transmitter and the intended receiver. Further, a 16 Bit cyclic redundancy check (CRC) is created and added to the packet. Then, the binary data $\boldsymbol{b}$ is fed to the PHY signal processing byte-wise. After the PHY layer, the created transmit sequence $\boldsymbol{x}$ is given to a block called frame control. This can be seen as a MAC functionality too, because it is responsible for ensuring that the transmission of the frames is restricted to a predefined time-grid. There are two options. Either, transmitting in bursts when a processed frame is available or a continuous transmission where the last transmitted frame is repeated, in case a new frame has not been provided. In both cases a minimum time distance between two transmit frames can be defined. On the receiver side the MAC layer checks the CRC and compares the IDs. If both match, the packet is transferred to the host computer, otherwise it is discarded.

A basic scheduling mechanism can be added on top for multi-user experiments. Here one transceiver is defined as access point (AP) (Sender ID $= 0$) which initiates all communications to the clients (Sender ID $\neq 0$). A client is only allowed to respond within a given time if it has received a message from the AP. It is not necessary for the client to respond to the AP, it can keep silent or use the assigned time slot to send the packet to

Parameter	Value/Range	Comment
Sender ID	$1, ..., 32$	
Receiver ID	$1, ..., 32$	
Repeat Data	True/False	to create a continuous transmission
Mode	AP / Client	
Distance	32 bit unsigned integer	time distance in samples between two frames
Scheduling Table (AP only)	Vector	contains the receiver IDs in the respective order
Response Time	32 bit unsigned integer	time distance in samples for the reply

Table 4.3: Parameter for the MAC layer

another client. This can be seen as a scheduled device-to-device communication link. The persistent scheduling order can be programmed via the host, where the AP iterates over a vector of receiver IDs to provide time slots for the clients. The AP either takes payload bytes from the host as input for the communication with the client or it uses dummy data.

4.1.2 Encoding and Mapping

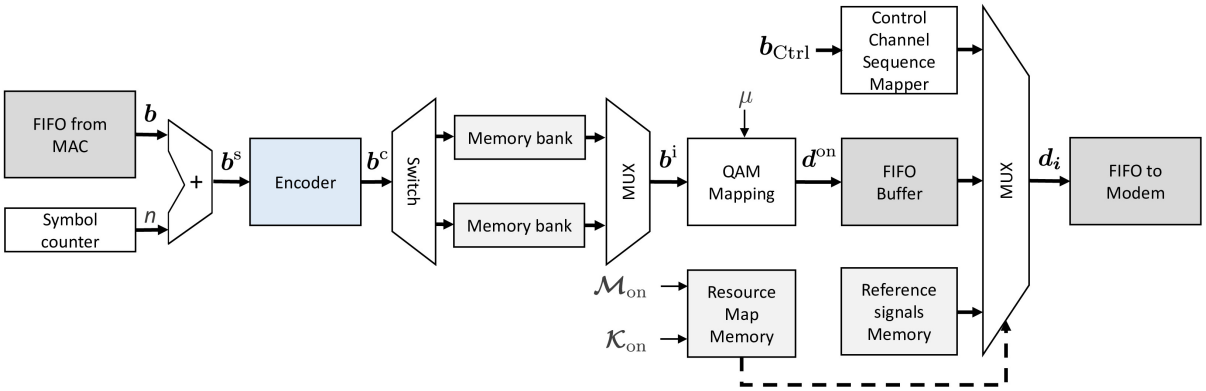


Figure 4.3: Diagram of the encoder taken from a library and other signal processing blocks to map the binary data into an IQ symbol stream.

The PHY signal processing starts, as visualized in Fig. 4.3, when the data is given to a scrambler to avoid repetitive patterns inside the bit sequence $\boldsymbol{b}$, which could lead to unnecessary peaks in the transmitted signal. This is achieved by adding the index n of the data byte to itself. The following channel coding is realized by a convolutional code with code rate ½. Both in the FPGA design as well as in this thesis, the Channel Coding and the viterbi decoding is considered as black boxes, taken from an external library. The coded bits $\boldsymbol{b}^{\mathrm{c}}$ from the encoder are interleaved by a transpose function to fully utilize the redundancy introduced by the coding. Here, the data $\boldsymbol{b}^{\mathrm{c}}$ is written row-wise into a Matrix like memory and read out column by column. Depending on the selected modulation order μ several bits in $\boldsymbol{b}^{\mathrm{i}}$ are grouped into one QAM symbol.

Parameter	Value/Range	Comment
μ	QPSK, 16-QAM, 64-QAM	Modulation order
N	$1, ..., 16$	Amount of samples per Block
$\boldsymbol{\mathcal{K}}_{\text{on}}$	Vector of size K	Active subcarriers
$\boldsymbol{\mathcal{M}}_{\text{on}}$	Vector of size M	Active subsymbols
$\boldsymbol{r}$	Vector with $B \cdot N$ samples	Contains resource map
B	$1, ..., 255$	Number of blocks per preamble
t_{Coding}	$2 + 3 + 8 \cdot \boldsymbol{b}$	Latency in cycles
t_{Mapping}	$4 + 2 + 7 + 2 + 1 + N_{\text{on}} + N$	Latency in cycles
c_{Coding}	90	Clock Speed [MHz] of the coding
c_{Mapping}	150	Clock Speed [MHz] of the Resource Mapper, Interleaver

FPGA Resources		
Register	7% 33135	
DSPs	0% 6	
Block Memory	6% 49	
Look-up-Tables	13% 32766	
Slices	18% 11679	

Table 4.4: Parameter for the mapping processes

Thereafter, the symbols $\boldsymbol{d}^{\text{on}}$ need to be mapped into the two-dimensional time-frequency resource grid of size $K \cdot M$. The amount of used data bins is defined by the vector of active subcarriers $\boldsymbol{\mathcal{K}}_{\text{on}}$ and subsymboles $\boldsymbol{\mathcal{M}}_{\text{on}}$. Furthermore, additional symbols such as pilots can be inserted by the resource mapper into the final symbol stream $\boldsymbol{d}_i$ of length N. To accomplish this, the resource mapper as depicted in Figure 4.3 has several input sources. Whenever a new data block is going to be created the multiplexing pattern is read out from a memory to control the switch at the output of this block. This mechanism allows to support any user-defined resource grid which can be adapted to the respective standard as also pre-defined training-sequences can be read out of a memory. The parameter B defines how many (GFDM) symbols are appended to one preamble, where every data symbol is indexed via i. The control channel input allows to add waveform configuration data to the packet. The 6 bit wide configuration data is encoded in sequences containing 48 samples each to ensure a robust transmission.

4.1.3 Modem

GFDM can be implemented using several ways: The authors in [153] are exploring the waveform using frequency-domain processing, this thesis author [148] is using time-domain expressions, which are build upon here. A third option combines this two approaches [138]. The aim of the implementation presented here is that the same structure can also be used at the receiver, just with a different order, as explained further in[46]:

Fig. 4.4 illustrates that the process of the waveform generation using the time-domain expression is divided into two main stages. The GFDM modem receives the symbol

stream $\boldsymbol{d}_i$ for the i-th symbol organized in frequency domain, as any other orthogonal frequency division multiplexing (OFDM)-based multicarrier system. Hence, a fast Fourier transform (FFT) will convert the data symbols to time-domain $\boldsymbol{\delta}_i$. Afterwards, the modem will convolve each data symbol $\delta_{k,m,i}[n]$, stored in memories for each subcarrier k and subsymbol m, with the pulse shaping filter $\boldsymbol{g}$. Starting point for the derivation will be equation (2.8):

$$\boldsymbol{x}_i[n] = \sum_{k\in K}\sum_{m\in M} d_{k,m,i}\boldsymbol{g}[\langle n - mK\rangle_N]e^{j2\pi\frac{k}{K}n}. \tag{4.1}$$

The data symbols $d_{k,m,i}$ for the ith transmit block in a sequence $\boldsymbol{d}_i$ are reorganized subcarriers before subsymbols using the relation $d_i[mK + k] = d_{k,m,i}$. Then, as shown in [154], the generation of the transmit sequence $\boldsymbol{x}_i[n]$ at the transmitter side can be written as

$$\boldsymbol{x}_i[mK + k] = \sum_{m'=0}^{M-1} g[\langle (m - m')K + k\rangle_N]\boldsymbol{\delta}[m'K + k] \tag{4.2}$$

$$\text{with } \boldsymbol{\delta}[m'K + k] = K \cdot \text{IDFT}\{d_i[m'K + \bullet]\}[k], \tag{4.3}$$

where a inverse discrete Fourier transform (IDFT) of size K maps the data $\boldsymbol{d}_i$ from frequency to time-domain for the currently processed subsymbol m'.

Similarly, at the receiver side, a linear receiver using the receive filter $\boldsymbol{\gamma}[n]$ is applied to the equalized data block $\hat{\boldsymbol{x}}_i[n]$ by

$$\hat{\boldsymbol{d}}_{k,m,i} = \sum_{n=0}^{N-1} \hat{\boldsymbol{x}}_i[n]\boldsymbol{\gamma}[\langle n - mK\rangle_N]e^{-j2\pi\frac{nk}{K}}. \tag{4.4}$$

In analogy to the transmitter, this can be reformulated to

$$\hat{\boldsymbol{\delta}}_i[mK + k] = \sum_{m'=0}^{M-1} \boldsymbol{\gamma}[\langle (m - m')K + k\rangle_N]\hat{\boldsymbol{x}}_i[m'K + k] \tag{4.5}$$

$$\hat{\boldsymbol{d}}_i[mK + k] = \sum_{k'=0}^{K-1} e^{-j2\pi\frac{kk'}{K}}\hat{\boldsymbol{\delta}}_i[mK + k'] \tag{4.6}$$

$$= \text{DFT}\{\hat{\boldsymbol{\delta}}_i[mK + \bullet]\}[k]. \tag{4.7}$$

Apparently, the modulator and demodulator have the operations (4.2), (4.5) in common, which are denoted as the circular convolution operation here. In addition, at the transmitter an inverse fast Fourier transform (IFFT) is employed before the circular convolution whereas at the receiver side, the according FFT is applied after the circular convolution. Hence, the same filtering and FFT[1] implementation can be reused at both transmitter and receiver side.

Fig. 4.4 shows that the first task of the convolution block is to split the incoming IDFT output stream δ into M vectors containing one sub-symbol with K samples each, e.g. δ_0,

[1] Switching between FFT and IFFT does usually incur only a negligible overhead.

δ_1, δ_{M-1}. In the following, each individual sub-symbol has to be repeated M times for the applied filter $\boldsymbol{g}$ with N samples. To accomplish this, the K samples of each sub-symbol are stored inside an independent sub-symbol data memory bank. These M parallel banks are read out sample by sample in parallel.

The filter $\boldsymbol{g}$ is stored in M parallel filter banks, too. Each of the filter memory banks contains K samples of the filter which are representing a sub-symbol as similar to the data. To perform the convolution, the filter has to be multiplied with the data in a circular shifted way. This is implemented by connecting a different filter bank to the multiplier for each data memory bank. For instance, first, the K samples of the first sub-symbol $\boldsymbol{\delta}_0$ are multiplied with $\boldsymbol{g}_0$. Afterwards the same K samples are multiplied with $\boldsymbol{g}_1$ until they have been multiplied with all filter coefficients. In the case of the second sub-symbol, the K samples stored inside the memory are multiplied first with $\boldsymbol{g}_1$ and then with $\boldsymbol{g}_2$. The last step of the process is to accumulate the contributions from all M parallel branches to get the transmit signal, finally.

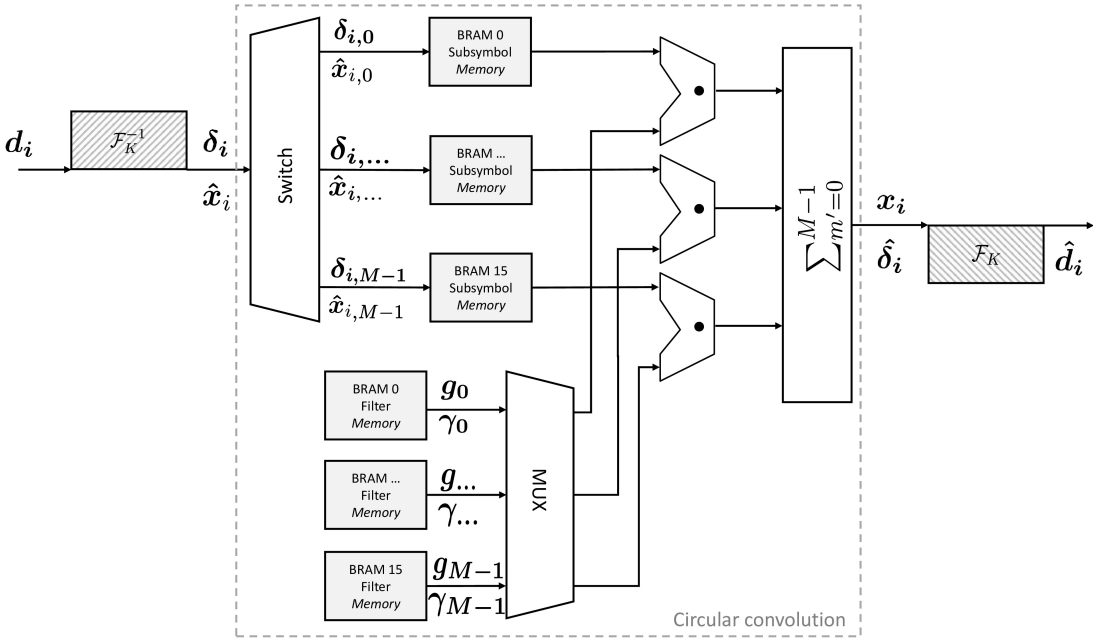


Figure 4.4: Block diagram of the modulator / demodulator. The symbols above the arrows mark the modulator, below the arrows the demodulator. The IFFT is used for the transmitter. The FFT on the receiver side.

Table 4.5 summarizes the characteristics of the Modem. At the receiver side the same processing blocks are utilized but in a reversed order. The transmit $\boldsymbol{g}$ and receive filter $\boldsymbol{\gamma}$ values can be freely chosen, where the amount of subsymbols M is limited to 16 by the current implementation. [54] shows that depending on the filter several waveform variants can be created by the GFDM framework. Table 4.10 shows that the other signal processing blocks need to be configured as well to realize this.

The modem has a wide parameter range, compared to a fixed OFDM variant and therefore

Parameter	Value/Range	Comment
K	2^n, $n = 3, ..., 9$	Number of subcarriers
M	$1, ..., 16$	Number of subsymbols
$\boldsymbol{g}$	Vector with $K \cdot M$ samples	Transmit filter
$\boldsymbol{\gamma}$	Vector with $K \cdot M$ samples	Receive filter
Latency t_{Modem}	$L_{\text{FFT}}(K) + N + 23$	in cycles
Clock Speed [MHz]	150	

FPGA Resources		
Register	8% 40550	
DSPs	17% 256	
Block Memory	18% 140	
Look-up-Tables	8% 20991	
Slices	18% 11403	

Table 4.5: Modem parameter

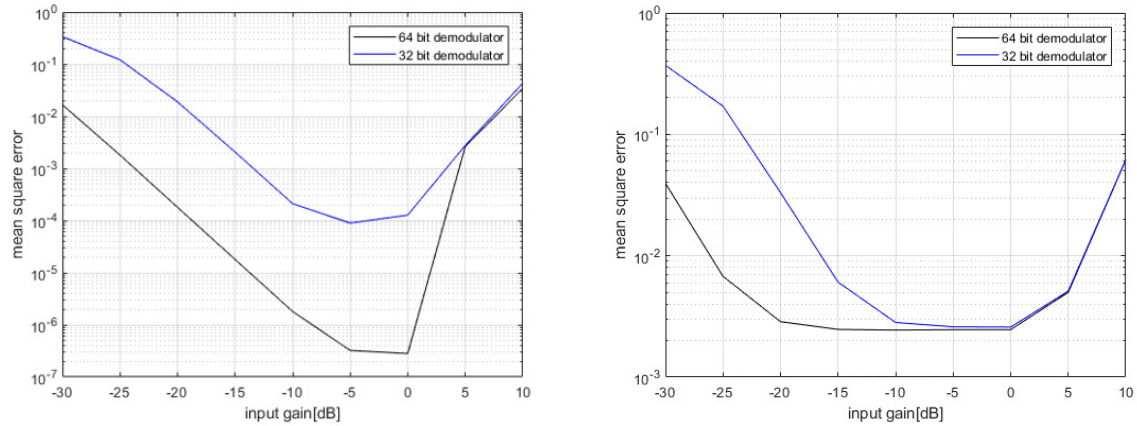
also increased requirements on the numerical stability. Therefore, test sequences are sent through the modulator running on the FPGA. Then a channel is applied to the modulated sequence, which is given to the demodulator thereafter. Finally the input and the output sequence are compared with each other. Fig. 4.5 shows the dynamic range of the modulator and demodulator implementations for 32 bit and 64 bit fixed point precision. The reason for the decision to take one of these two bit widths is because the memories are employed either way. The 64 bit bit signal processing achieves a mean squared error (MSE) of up to 10^{-6} as long as the data input format is scaled precisely with respect to the pulse shaping filter values. In case the input signals are affected by a 20 dB deep notch in the frequency response of the channel, the precision is reduced to a MSE of $2 \cdot 10^{-3}$ and is close to the 32 bit version. However, the dynamic range of the input signal is 10 dB wider and relaxes the requirements with respect to the correct scaling of the input signal, which is important on the receiver side, due to fading effects. Therefore, the implementation uses the 64 bit version. As indicated earlier, the following modules process the data using 32 bit, thus the 64 bit wide signal word has to be shifted and truncated by a scaling block. This scaling operation can be controlled from the host and realizes the scaling operations with respect to the IDFT, too.

4.1.4 Post modem processing

To prepare the data signals for the transmission process, a cyclic prefix (CP) with the length N_{CP} and, depending on the configuration, a cyclic suffix (CS) of length N_{CS} needs to be added for each transmitted data block $\boldsymbol{x}_i$.

$$\boldsymbol{x}_i^{\text{t}} = [\boldsymbol{x}_i^{\text{T}}(N - N_{\text{CP}} + 1 : N)\boldsymbol{x}_i^{\text{T}}\boldsymbol{x}_i^{\text{T}}(1 : N_{\text{CS}})] \tag{4.8}$$

The implementation, shown in Fig. 4.6 solves this using two independent memory banks, where the modulated symbol $\boldsymbol{x_i}$ is written in one of the two. Afterwards, this memory is



(a) Without channel impairments **(b)** 20 dB deep fading

Figure 4.5: Dynamic range of the 32 or 64 bit modulator and demodulator with and without 20 dB notch in the frequency response of the channel

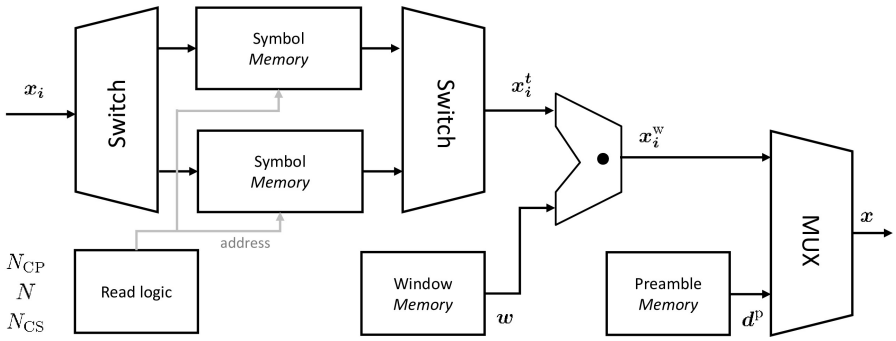


Figure 4.6: Block diagram of the CP/CS, the windowing and preamble insertion logic.

read out such way, that the samples of the CP are given to the output $\boldsymbol{x}_i^t$ first, followed by the data symbol and finally the samples of the CS conclude the frame.

As already indicated in the pre-standard definition of 802.11a [40], windowing can suppress the out-of-band (OOB) emissions significantly by multiplying an window $\boldsymbol{w}$ element-wise with the data $\boldsymbol{x}_i^{\mathrm{t}}$.

$$\boldsymbol{x}_i^{\mathrm{w}} = \boldsymbol{x}_i^t \odot \boldsymbol{w} \tag{4.9}$$

So, a windowing unit follows, where only the rising half is stored inside a memory. An integrated counter in the control logic counts up until N_{W} is reached to trigger the memory for the appropriate samples. During the main data block the unit is disabled, so that the data block will be multiplied with one. Finally, the same counter is decreased to create the falling part by reading out the window memory in a reversed order.

Finally, the preamble $\boldsymbol{d}^{\mathrm{p}}$ is prepended to the frame $\boldsymbol{x}_i^{\mathrm{w}}$.

$$\boldsymbol{x} = [\boldsymbol{d}^{\mathrm{pT}} \boldsymbol{x}_{\boldsymbol{0}}^{\mathrm{wT}} \cdots \boldsymbol{x}_i^{\mathrm{wT}}] \tag{4.10}$$

Different to the data, the complete preamble is precalculated on the host computer. Therefore, it will be written to a memory during the configuration phase of the transceiver.

Parameter	Value/Range	Comment
N	$1, ..., 2048$	Frame length
N_{CP}	$0, ..., 257$	CP length
N_{CS}	$0, ..., 257$	CS length
B	$1, ..., 255$	Number of blocks
Total frame length	$= B \cdot (N + N_{\mathrm{CP}} + N_{\mathrm{CS}})$	Limited to 32768 samples
$\boldsymbol{w}$	$1, ..., 2048$	Vector with the time window
Latency t_{TX}	$N + 7 + N_{\mathrm{CP}} + N_{\mathrm{CS}}$	in cycles
Clock Speed [MHz]	150	

FPGA Resources		
Register	4% 18722	
DSPs	0% 0	
Block Memory	11% 795	
Look-up-Tables	4% 11274	
Slices	8% 5147	

Table 4.6: Parameter for the post modem processing

Whenever the controller has finished reading the first data block of a frame into the data-first in, first out (FIFO), the preamble insertion unit is triggered to push the preamble samples to the final block. The resulting signal $\boldsymbol{x}$ is forwarded to the frame control logic.

4.1.5 Synchronization

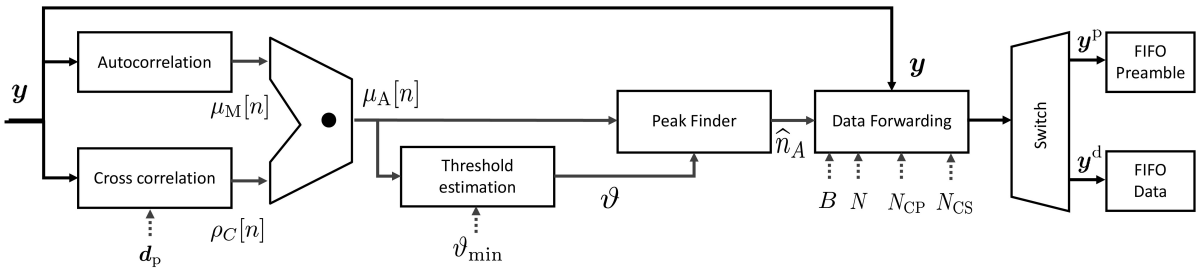


Figure 4.7: Block diagram of the signal processing modules used for the synchronization

The synchronization is a key module for the receiver, because it limits the sensitivity and robustness of the transceiver signal processing. Wireless networks utilizing standards such as WLAN or LTE use algorithms based on Schmidl-Cox [155] for the time and frequency synchronization of the data frame. The idea is to first utilize an auto-correlation of the received signal to coarsely detect the beginning of the preamble and then a cross-correlation with the known preamble to find the sample exact start [156]. This results in a threshold detection problem as discussed in [48]. A high dynamic range for the detection threshold is important. Especially, because the energy induced by burst transmission such as WLAN is low. In contrast to continuous communications like LTE, where a receiver has a constant data stream to adjust. The synchronization algorithm is

presented here briefly with the focus to describe the implementation. The thresholds to differentiate a successful detection have been determined by experimental analysis.

Before the synchronization, an automatic gain control (AGC) block ensures that the signal has the optimal gain without clipping. The calculated value is given to the RF attenuator after the low noise amplifier (LNA) to avoid high voltage levels at the input of the ADC. Though, in most cases the attenuation is set to zero, because the received RF signal is to weak. The AGC is most important in case the receive antenna is placed close to the transmitting one to prevent clipping.

In principle, arbitrary sequences would work for the auto- or cross-correlation with differing performance. LTE based cellular networks are employing Zadoff-Chu sequences for the purpose of having a good auto-correlation property and a low peak-to-average power ratio (PAPR) due to the constant-amplitude [157]. So, the implemented synchronization module uses two identical Zadoff-Chu sequences $d_{\mathrm{p,half}}$ as preamble d_{p}, that contains in total 128 samples. Further, a CP and CS is added to protect the preamble from impairments of the channel.

Fig. 4.7 outlines the implementation, where the receiver constantly correlates the incoming samples $\boldsymbol{y}$ with a time-shifted version to find the data frame, according to [155]:

$$\rho[n] = \sum_{k=n}^{n+127} y[k]^* y[k-64]. \tag{4.11}$$

Since a peak has to be detected reliably, the resulting metric is normalized using:

$$\mu_{\mathrm{S}}[n] = \frac{2|\rho[n]|^2}{\sum_{k=n}^{n+127} |y[k]|^2}. \tag{4.12}$$

To smooth the plateau effect of CP and CS the following expression integrates over the length of both [158] and determines the coarse start of the preamble

$$\mu_{\mathrm{M}}[n] = \frac{1}{L+1} \sum_{k=n-L}^{n} \mu_{\mathrm{S}}[k]. \tag{4.13}$$

The more exact start of the frame is found using the cross-correlation with the known transmitted preamble and is given by

$$\rho_C[n] = \frac{1}{64} \sum_{k=0}^{63} r'[n+k] d_{\mathrm{p,half}}[k], \tag{4.14}$$

where $d_{\mathrm{p,half}}[k]$ contains one halve of the original preamble. Since two concatenated sequences are sent, the two resulting peaks are added up together to get the main peak of the cross-correlation.

The metrics (4.13) and (4.14) are combined [159] to sharpen the peak indicating the start of the preamble

$$\mu_{\mathrm{A}}[n] = |\rho_C[n]| \cdot \mu_{\mathrm{M}}[n]. \tag{4.15}$$

Parameter	Value/Range	Comment
K	2^n, $n = 3, ..., 8$	
M	$1, ..., 16$	
$\boldsymbol{d}^{\mathrm{p}}$	64	Vector with preamble
Latency t_{Synch}	$3 + 482 + 18 + 9 + 1$	in cycles
Clock Speed [MHz]	200	

FPGA Resources		
Register	6% 30867	
DSPs	17% 269	
Block Memory	1% 10	
Look-up-Tables	12% 32569	
Slices	19% 12305	

Table 4.7: Parameter for the synchronization

Both metricizes combined result in several peaks, where the maximum $\widehat{n}_A$ has to be found within the peaks defined by the auto-correlation $\widehat{n}_m$:

$$\widehat{n}_m = \arg\max_n \mu_m[n] \tag{4.16}$$

$$\widehat{n}_A = \underset{\widehat{n}_m - N/2 \leq n \leq \widehat{n}_m + N/2}{\arg\max} \mu_A[n] \tag{4.17}$$

However, a threshold has to be reliably defined to differentiate peaks from the noise. Further, this threshold ϑ should be dynamically adapted for burst transmissions. In the realization the combined metric μ_{A} is summed up within a window of the size of one preamble half (64 samples) and compared with the previous definition of the threshold

$$\vartheta = \max\left(\vartheta, \frac{1}{4}\sum_{n=0}^{63}\mu_{\mathrm{A}}[n]\right). \tag{4.18}$$

A minimum threshold ϑ_{min} is predefined by the host configuration. Once the sum of the μ_{A} falls below this minimum threshold, ϑ is reset to the default value ϑ_{min}. This ensures that the threshold is not growing over time. So, whenever the synchronization starts to detect a preamble the threshold will rise and once the whole preamble sequence is processed, the threshold will be reset.

Finally the data forwarding module counts the sample positions and splits the received signal into two parts: the received preamble $\boldsymbol{y}^{\mathrm{p}}$ and the received data block $\boldsymbol{y}^{\mathrm{d}}$ without CP and CS and transfers them using FIFOs to the next processing steps.

4.1.6 Channel Estimation and Equalization

The impairments introduced by the wireless channel are estimated based on the received preamble. This approach is inspired by WLAN principles and the implementation depicted

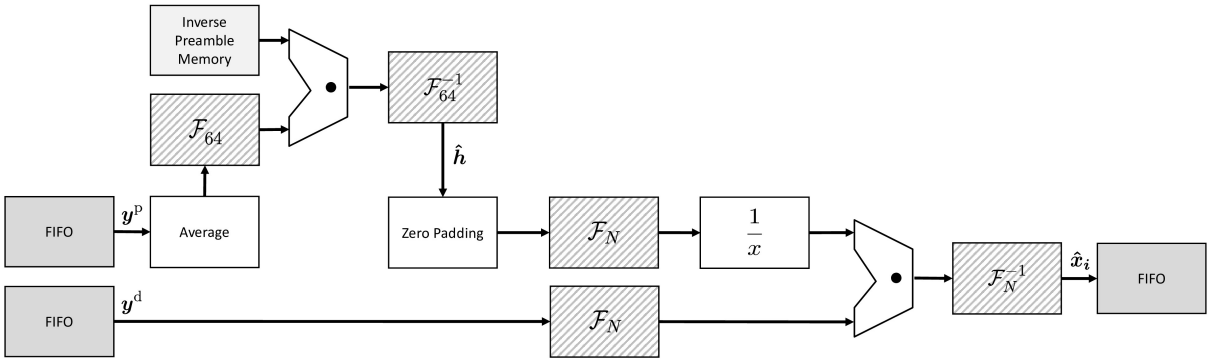


Figure 4.8: Overview over the channel estimation and equalization module

in Fig. 4.8. The channel estimation module will calculate the channel impulse response (CIR) according to

$$\hat{\boldsymbol{h}} = \mathcal{F}_{64}^{-1}\left(\mathcal{F}_{64}\left(\frac{1}{2}\sum_{i=1}^{2}\boldsymbol{y}^{\mathrm{p},\boldsymbol{i}}\right) \odot \frac{1}{\mathcal{F}_{64}(d_{\mathrm{p,half}})}\right). \tag{4.19}$$

Since, the half preamble is generally fixed to 64 samples, also the used FFT module is fixed to a length of 64. Before performing the discrete Fourier transform (DFT), both preamble halves are averaged $\frac{1}{2}\sum_{i=1}^{2}\boldsymbol{y}^{\mathrm{p},\boldsymbol{i}}$ to use both parts of the received preamble for the channel estimation. The inverse of the fourier transform (FT) of the known preamble $\frac{1}{\mathcal{F}_{64}(d_{\mathrm{p,half}})}$ is pre-calculated on the host computer and written to a memory on the FPGA, such that it can be multiplied sample by sample with received preamble in frequency domain. Finally, a inverse FT converts the estimate back into time-domain to get the CIR.

The length of the payload N can be varied. Therefore, the estimated channel frequency response needs to be interpolated, prior applying to the received data frame. In case, N is larger than 64 samples zeros will be appended to $\hat{h}$ and the term $\frac{1}{\mathcal{F}_N(\hat{\boldsymbol{h}})}$ converts and interpolates the data back into frequency domain. Now, the channel equalization module equalizes the received data $\boldsymbol{y}^{\mathrm{d}}$ using a zero-forcing (ZF) algorithm:

$$\hat{\boldsymbol{x}} = \mathcal{F}_N^{-1}\left(\mathcal{F}_N(\boldsymbol{y}^{\mathrm{d}}) \odot \frac{1}{\mathcal{F}_N(\hat{\boldsymbol{h}})}\right) \tag{4.20}$$

Here, the channel estimation results are stored inside a memory such that they can be repeated in case several data blocks B are following one preamble. To increase the parameter space both *Xilinx* FFT and DFT cores are used and named "Dual FFT" block in this context. The equalized data is passed to the demodulator. The Table 4.8 summarizes the parameter space and the necessary resources on the FPGA.

Following the WLAN standard it is possible to spread further training sequences as pilots using the resource mapper. So, this channel estimation and equalization module can be

Parameter	Value/Range	Comment
N	2^n, $n = 3, ..., 8$ and DFT core	Dual FFT
$\frac{1}{\mathcal{F}_{64}\boldsymbol{d}^{\mathrm{p}}}$	64	Vector with the preamble
Latency t_{est}	$68 + 214 + 9 + 214 + 2 + 3$	in cycles
Latency t_{equ}	$B \cdot (2 \cdot L_{\mathrm{FFT}}(N) + 55)$	in cycles
Clock Speed [MHz]	200	

FPGA Resources		
Register	8% 38935	
DSPs	5% 72	
Block Memory	4% 33	
Look-up-Tables	10% 25666	
Slices	15% 9248	

Table 4.8: Parameter for the channel estimation and equalization

used to initially correct the channel and then gradually track and correct the phase and amplitude for example as described in the following section. A LTE kind of channel correction is not foreseen in the implementation and subject to later work.

4.1.7 Demapping

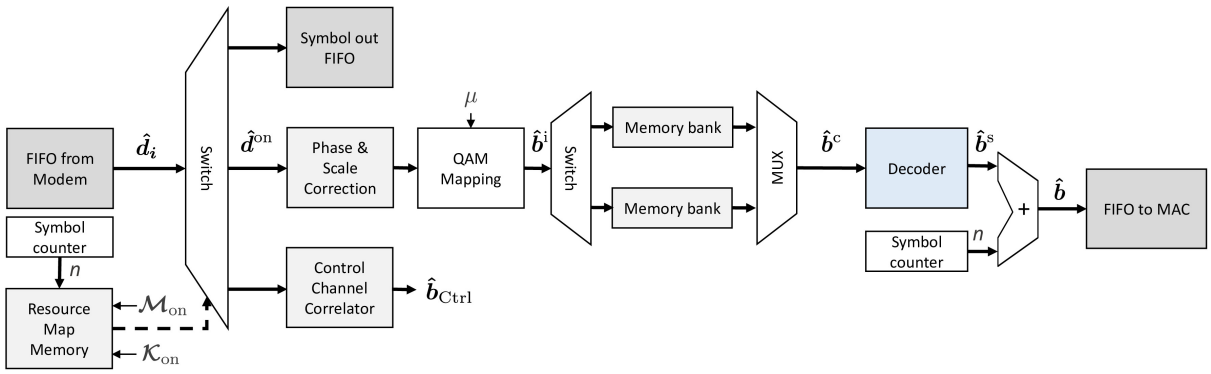


Figure 4.9: Overview over the demapping process to convert the demodulated IQ symbol stream to binary data.

The resource demapping is outlined in Fig. 4.9. The received data symbol $\hat{\boldsymbol{d}}_i$ is given from the demodulator to the resource de-mapper, which separates the different data streams, such as pilot or control channel data from the payload $\hat{\boldsymbol{d}}^{\mathrm{on}}$. Here the resource map is read-out from a memory and controls a switch to forward the data to the right output. In case a symbol is marked as a pilot, it is stored to adjust potential phase errors but, more importantly, to restore the correct scale of the data symbol too.

The QAM de-mapper assigns a matching bit pattern $\hat{\boldsymbol{b}}^{\mathrm{i}}$ to the incoming symbols based on the modulation scheme μ using hard-decision. The de-interleaver performs the similar

Parameter	Value/Range	Comment
K	2^n, $n = 3, ..., 8$	
M	$1, ..., 16$	
$\boldsymbol{g}$	Vector with $K \cdot M$ samples	
Latency t_{Demap}	$1 + 6 + 1 + 22 + 1 + 2 + 2 + 7 + 2 + 2 \cdot (61 + 8 \cdot \boldsymbol{b} + 3 + 3)$	

Table 4.9: Parameter for the demapping processes. The resource utilization is given in Table 4.4.

operation as the interleaver and is transposing the symbols for the decoder input $\hat{\boldsymbol{b}}^{\text{c}}$. The decoder performs a viterbi algorithm to decode the received byte stream $\hat{\boldsymbol{b}}^{\text{s}}$. Finally, the de-scrambler is subtracting the byte index n from the decoded bytes and creates the received packet $\hat{\boldsymbol{b}}$. Afterwards, the received packet is given to MAC layer to judge if the packet is acceptable or not.

4.1.8 Flexible Configuration

Using the described blocks, a multitude of waveforms for wireless communication systems can be emulated. A subset of existing possibilities is shown in Table 4.10. There, for different waveforms, the signal processing blocks are parametrized differently or even bypassed, such that in total the desired waveform is generated. For example, multicarrier waveforms such as OFDM or filter bank multicarrier (FBMC) employ the FFT operation at the modem to obtain a frequency-division multiplexing structure. On the other hand, the definition of the resource map strongly depends on the actual frame structure and no general statement can be made. Here, the flexibility of the resource mapper to support different standards shall be emphasized.

The circular convolution unit can not only be used for classical filtering, such as in GFDM, but it can also serve as the spreading operator for direct-sequence spread spectrum (DSSS) signaling. As an addition, some of the variants, such as FBMC, need to overlap several symbols together. This requires a change in the logic, that could be accomplished by replacing the MUX function with an add operation in the CP/CS processing block after the circular convolution. This overlap can be accomplished because several (sub-)symbols are stored inside the memory and the summation allows to combine them. Further, although DSSS signals can be transmitted, the receiver has to correlate the incoming sequence with the known ones. Here, the circular convolution block would have to sum up the individual contributions of each filter bank and then decide which one has the highest match. The implementation of the circular convolution presented in [148] includes this required summing function.

However, keeping the configuration and the data processing synchronous requires fine control. Fig. 4.10 shows the simulated execution of the signal processing modules in the FPGA for two different configurations. In the first case a configuration is depicted, where

Waveform	Resource Mapper	FFT	Circular Convolution	CP/CS	Window	Overlap	Preamble
OFDM	Frame-dependent	ON	OFF	ON	OFF	OFF	Frame-dependent
LTE	Pilot/Control	ON	OFF	ON	OFF	OFF	OFF
WiFi	Header/Payload	ON	OFF	ON	OFF	OFF	ON
Single-Carrier	Frame-dependent	OFF	OFF	ON	OFF	OFF	Frame-dependent
5G NR [160]	Pilot/Control	ON	OFF	ON	ON	ON	OFF
GFDM [161]	Frame-dependent	ON	GFDM pulse	ON	ON	OFF	Frame-dependent
FBMC [162]	Pilot/Control	ON	FBMC pulse	OFF	ON	ON	OFF
Spread-spectrum [163]	Frame-dependent	OFF	Spreading function	OFF	OFF	OFF	Frame-dependent
Chirp-Based [164]	Frame-dependent	ON	Chirp function	ON	OFF	OFF	Frame-dependent

Table 4.10: Configuration for waveform generation, taken from [46].

all baseband processing modules are enabled and set to the maximum parameter settings, in the other case the waveform parameters are set to the minimum values. The black color indicates that a certain module is active and it can be observed that for the maximum parameter settings a new packet start is triggered, when the previous packet is still written to the output. In case the baseband processor has to fill a continuous time grid, the signal processing cannot be stopped for a reconfiguration. In the minimum case each transmitted packet is processed much faster, such that most modules are idle, which is indicate by the white areas. Here, future research could show that modules can be deactivated to save energy, for instance.

In general, each FPGA processing module requires a defined amount of samples at the input to conduct the signal processing. If the sample count is misaligned the whole process will not continue. As indicated by the previous sections, many different configuration parameter have to be written to the FPGA transceiver for correct operation. This includes single values such as the amount of samples to be processed but also vectors and whole matrices in case of the GFDM pulse shaping filter. In addition, each module has a different amount of parameter inputs and several developers are working across the various modules. Therefore, a very defined set of data exchange interfaces is needed to organize the interactions with the host computer.

To deal with these requirements several options are possible. The first option is to build a global state machine which orchestrates all modules. This allows a very detailed control of all modules, but every time a module is changed, the state machine has to be adapted. More, the developer integrating all those modules needs to understand each module in detail. The next option is to keep all modules independent and each module has to be configured by the host computer separately. This method has the disadvantage that a timely coordinated operation is difficult. A limited amount of host to FPGA inputs and outputs are available which need to be shared with all FPGA modules. The third option is

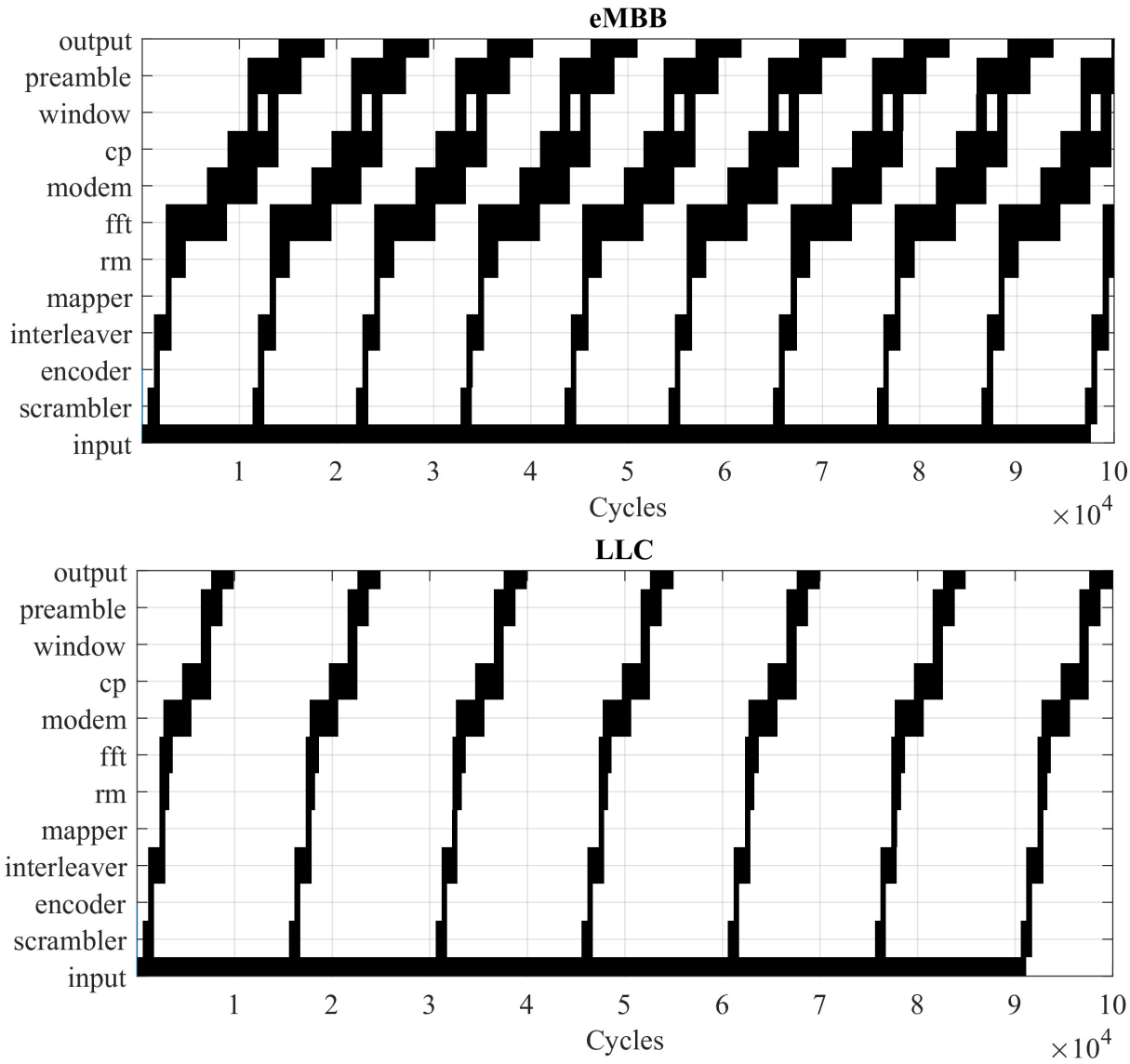


Figure 4.10: Simulation of the signal processing at the transmitter for the eMBB case and for the LLC case. The black areas mark the activity of a specific module, white inactivity. In this graphic, all modules operate with a processing speed of 150 MHz.

to introduce a bus system, such as the Serial Peripheral Interface (SPI) for digital circuits.

The idea is to daisy-chain all modules with a 64 bit wide bus as presented in Fig. 4.11. The host computer acts as the bus master and sends messages with individual addresses to the FPGA via a direct memory access (DMA) FIFO buffer. Then, the messages are passed from one block to another. In case the address ID fits a respective block, it is taken from the bus and processed in the block. Further, in case the same block has to deliver information back to the master, or to other modules, the same time slot is used. Fig. 4.12 sketches such a module. It is shown, that the data symbols are transferred independent of the control and debug messages over the bus to avoid congestion. Further it is shown

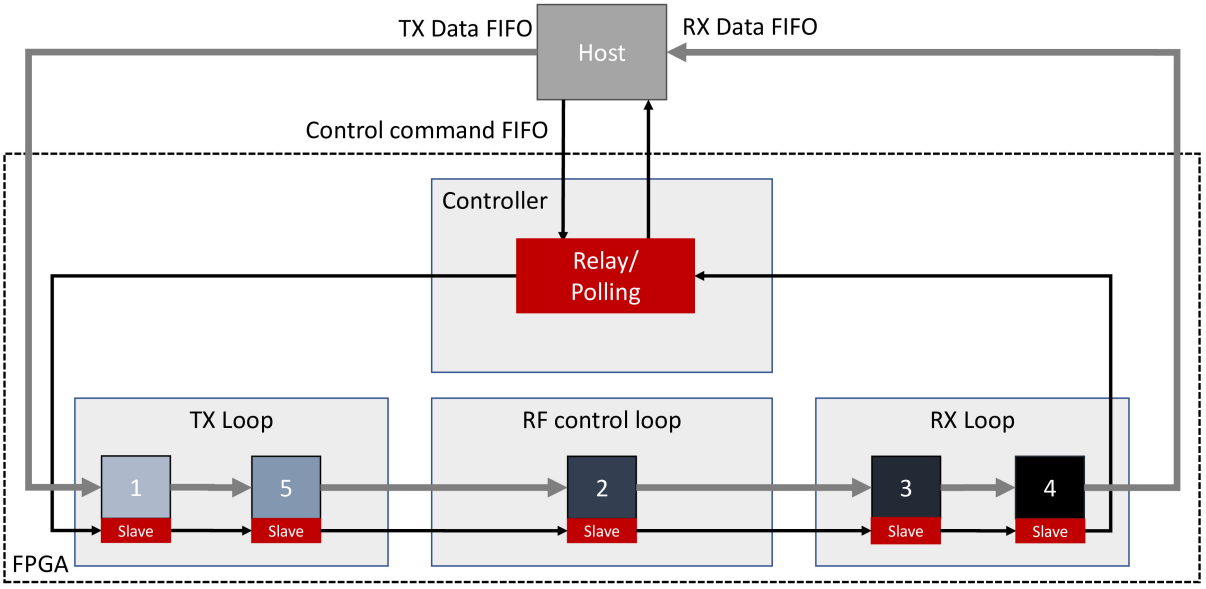


Figure 4.11: Diagram of the transceiver controlled by a Master/Slave Bus-system

that parameters are stored in double registers. This way new values can be programmed without disturbing the signal processing. In case the processing unit is ready to apply the values a dedicated flag toggles the second register to take over the new value. This flag can either be indicated via the bus or as part of the data path to ensure it is applied in the right moment between different frames. Vectors or Matrices are kept inside block random access memory (RAM) memories, where the new values are stored using an address offset and can therefore act as caches for several configurations. This way, also filter vectors can be exchanged quickly and synchronized with the other parameters by just changing the address offset.

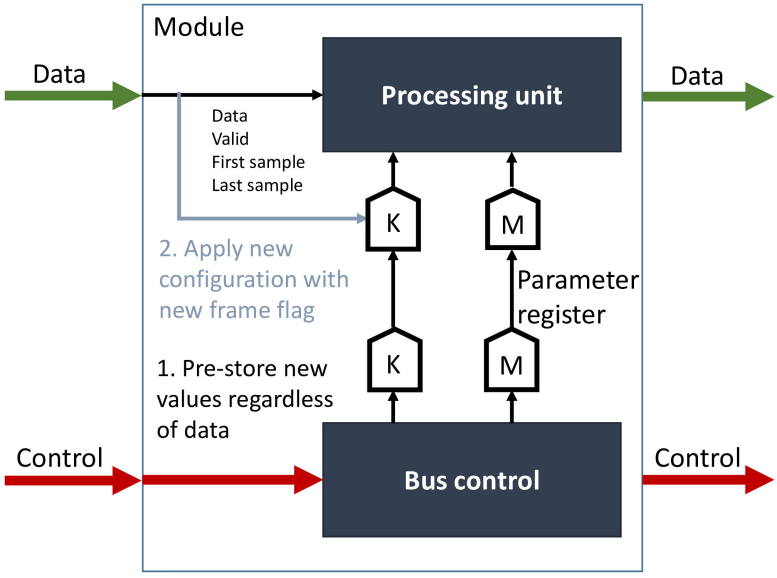


Figure 4.12: Diagram of a bus module

This idea also supports easier integration of new modules into the chain, because the interface is standardized for each module and the new configuration is handled by adding new messages to the bus input FIFO. This can be accomplished by reading them line by line from a text-file or reading them from network messages. The same technique can also be used to transfer debug information from a FPGA module to the host or even to the network. For instance, a complete CIR could be requested and transferred by the bus, without changing the other modules. This concept therefore allows to transform the presented implementations into a library of standard modules which can be re-assembled for other ideas. The data bus is always kept 32 bit wide, even in case values of lower precision such as bits or bytes are exchanged. The intention is to allow to bypass several processing modules such that even raw in-phase and quadrature (IQ) data can be streamed to RF.

Fig. 4.13 shows that the implemented signal processing modules at the transmitter support very fast parameter reconfigurations during run-time. It shows two different waveform configurations multiplexed in time, captured by another SDR device. The first, longer signal has the ”WiFi” configuration to deliver a video stream, whereas the second signal uses a shorten variant of the ”LLC” configuration for faster signal processing as required by control loop applications. In both cases, different preambles are prepended before the data frame to distinguish between the different services. For the demonstration, two receiving USRP SDR were used to receive either one or the other service. The distinction at the receiver is based on the different preambles.

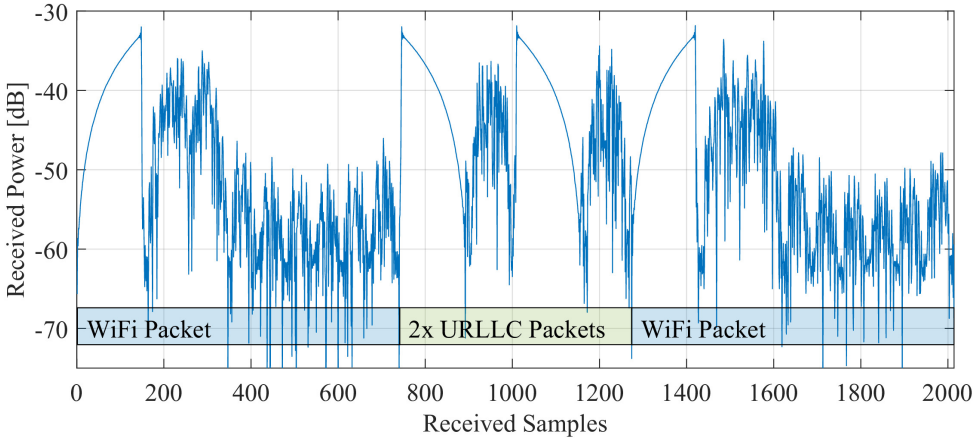


Figure 4.13: Captured RF signal of two different numerologies created by the flexible transmitter multiplexed in time. The signal shape of the preambles is modified for visual identification and will not be used in practical systems. Adapted from [165].

A fast reconfiguration at the receiver would require the help of additional control information to inform about the current waveform configuration. Therefore, a physical layer convergence procedure (PLCP) header modulated in a common waveform setting would have to follow the preamble, similar to the procedure in WLAN standards. After

this header is decoded, the receiver can be programmed with the necessary parameter settings for the remaining data frame. Demonstrating this capability is subject to further work.

4.2 Analysis

This section investigates the performance of the implemented physical layer. First, the fixed point precision of the FPGA implementation is compared to a floating point reference. Second, the latency is evaluated and lastly the resource consumption in comparison with the *NI* Application Frameworks (AFWs) is investigated. The AFWs act as reference, because they use the same hardware platform and represent implementations of the LTE and WLAN standards.

4.2.1 Numerical Precision

The performance of the implemented transceiver is evaluated by emulating additive white Gaussian noise (AWGN) conditions digitally. Fig. 4.14 shows the achieved result for different signal-to-noise ratios (SNRs), where the SNR is estimated based on the transmitted and received preamble. The waveform parameters are based on the "LLC" configuration. The simulation is using a complex floating point reference, while the FPGA implementation uses complex fixed-point with 32 bits precision for representing the transmitted signal. The performance is consistent with the simulations with a difference of up to 1 dB due to the reduced precision of the FPGA signal processing. So, it is assumed that the transceiver will achieve a sufficient performance while using the USRP X310 platform. Further, a traditional FPGA design flow entails the creation of fixed point models to simulate the behavior of the FPGA. The design flow applied here skips this step and still allows to study the principal performance using the base-band processor in the testbed.

4.2.2 Spectral analysis

The achieved spectrum of the implemented transceiver using the different waveform parameters and the *NI* LTE-AFW[2] are depicted in Fig. 4.15. The spectrum is measured with a *Rohde & Schwarz* spectrum analyzer which is attached to the transmit port of the *NI* USRP 2944. Fig. 4.15 shows that the transmit power varies with the signal, although the same transmit power is used. The bandwidth is normalized for each signal configuration, such that the usage of the spectrum can be evaluated.

[2] The *NI* LTE-AFW is a commercial implementation of the LTE 20 MHz PHY layer running on the same hardware platform as the implemented transceiver.

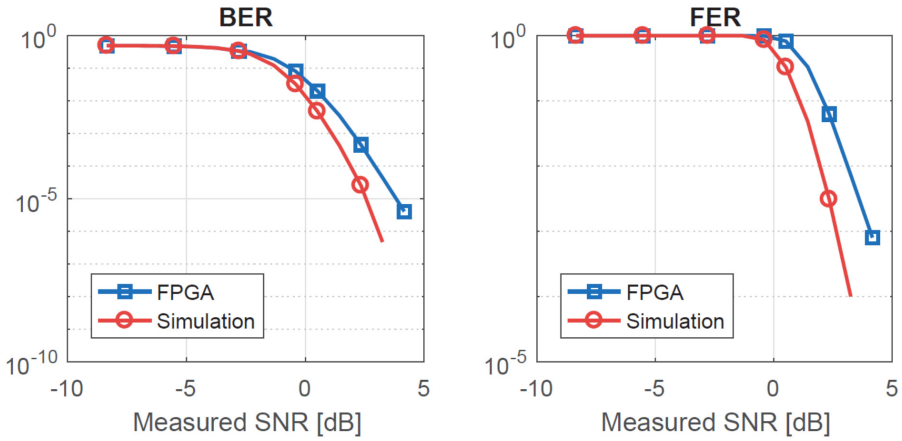


Figure 4.14: Bit and frame error rate of FPGA compared to floating point reference for the LLC parameter setting. Taken from [46].

The implemented transceiver using the "IoT" parameters has the highest output power because the complex IQ samples are transmitted without any additional filtering operation. Therefore, the signal energy can be normalized to use the full precision range of the DAC. The contrary case is the *NI* LTE-AFW which is designed to provide good signal quality. Thus, a large headroom is chosen to avoid clipping of signal peaks.

The curve marked with "LLC" is using the GFDM pulse shaping filter to reduce the OOB emissions compared to the "WiFi" setting, where Block-OFDM is used. The OOB performance can be compared by examining the power levels in the center of Fig. 4.15. However, the OOB differences (up to 13 dB) are not as good compared to the results achieved in Fig. 2.4 (up to 30 dB) where well-calibrated measurement equipment was used to create the signals. Therefore, it will be difficult to demonstrate the enhancements of pulse-shaping filters clearly. In this case, RF-frontends with better quality are needed, such as the *NI* 5791 adapter module. However, this module is more expensive, bulky and needs two FPGA cards[3] to run the transceiver, compared to the used USRP platform.

4.2.3 Latency

The signal processing latency on the FPGA is deterministic and depends on the parameter settings. Hence, the transmit latency can be expressed in FPGA clock cycles as

$$L_{\text{Transmitter}} = \frac{c_{\text{data}}}{c_{\text{Coding}}} \cdot t_{\text{Coding}} + \frac{c_{\text{data}}}{c_{\text{Mapping}}} \cdot t_{\text{Mapping}} + \frac{c_{\text{data}}}{c_{\text{Modem}}} \cdot t_{\text{Modem}} + \frac{c_{\text{data}}}{c_{\text{Post}}} \cdot (t_{\text{Post}} + N), \tag{4.21}$$

where the individual latencies are noted in the parameter tables of the individual modules. Further, it takes additional N clock cycles to write out all the samples after the first one. All signal processing modules can run with 200 MHz clock speed, except the decoder

[3] Although the same Kintex 7 FPGA is used, the FPGA driver environment allocates more resources.

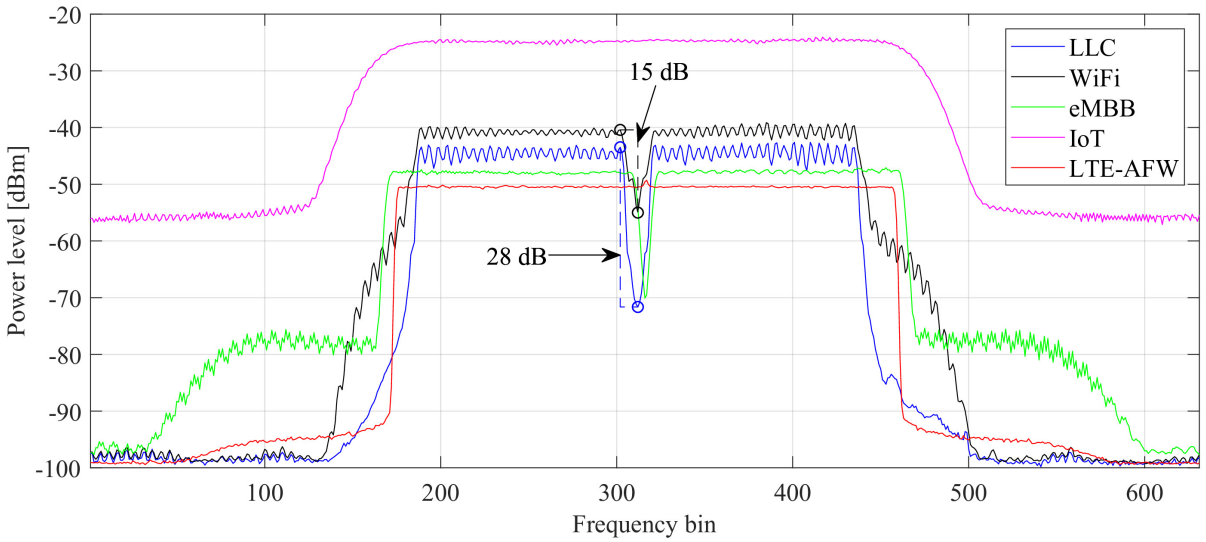


Figure 4.15: Spectrum of the transceiver and the *NI* LTE-AFW for the same transmit power setting. The bandwidth is adjusted for each signal individually and is doubled compared to the used channel bandwidth defined in Tab. 2.1.

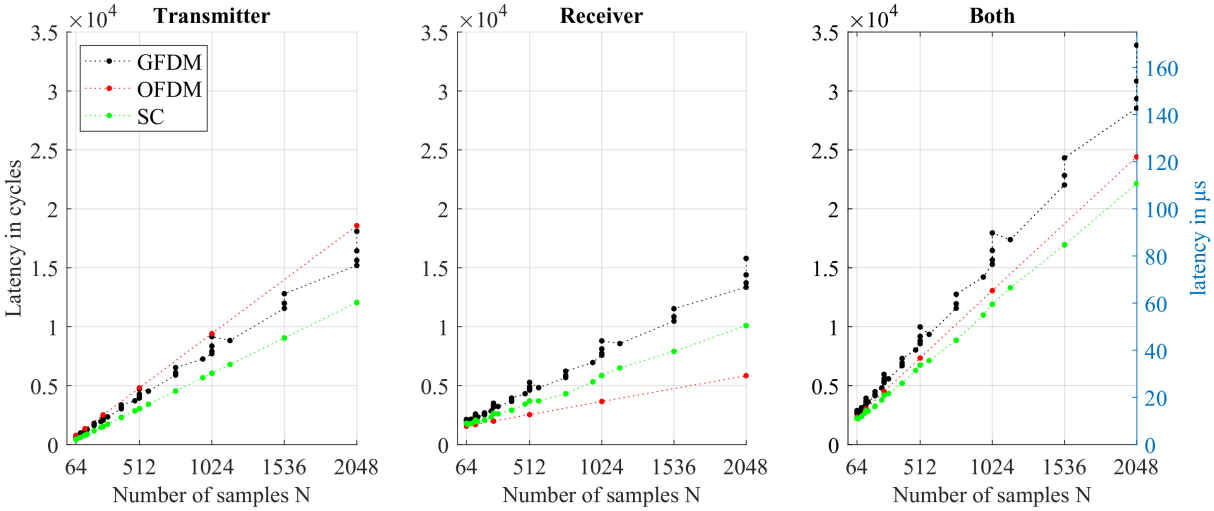


Figure 4.16: Latency of the FPGA signal processing, where $N_{\mathrm{CP}} = N_{\mathrm{CP}} = \frac{N}{4}$ and half of the N data symbols are used for payload.

which is limited to 100 MHz. In the used transceiver for the experiments the data clock is limited for most modules to leave more freedom for the placing stage of the FPGA compiler. Therefore, the processing speed varies, which is respected in the equations and the figures. In this section only the physical layer signal processing is considered, since the MAC can introduce ”arbitrary” latencies due to the configurations. The latency of FIFO crossing different clock domains is neglected, as long as they do not buffer a whole frame of N samples.[4]

[4] The contribution of a FIFO to the overall latency is not defined in the programming manuals. Experiments show only very few delays below 5 clock cycles.

The latency of the signal processing shown in Fig. 4.16 is linear with the frame size and mainly depended on the FFT latencies, which are noted in Tab. A.1, and the frame length N. Mainly because the interleaving function, the modem, and the CP prefix module need to delay the processing until the whole block is transferred to a memory. The different FFT processing delays can be seen as steps in the two graphics, because the total amount of samples do not change, but the amount of subcarriers and therefore the FFT size changes. The bypassing function of the modules allows to gain some latency savings in the case that features such as the circular convolution to process GFDM is disabled. This is visible in the single-carrier (SC) curve in the figures.

The receiver latency behaves similarly as

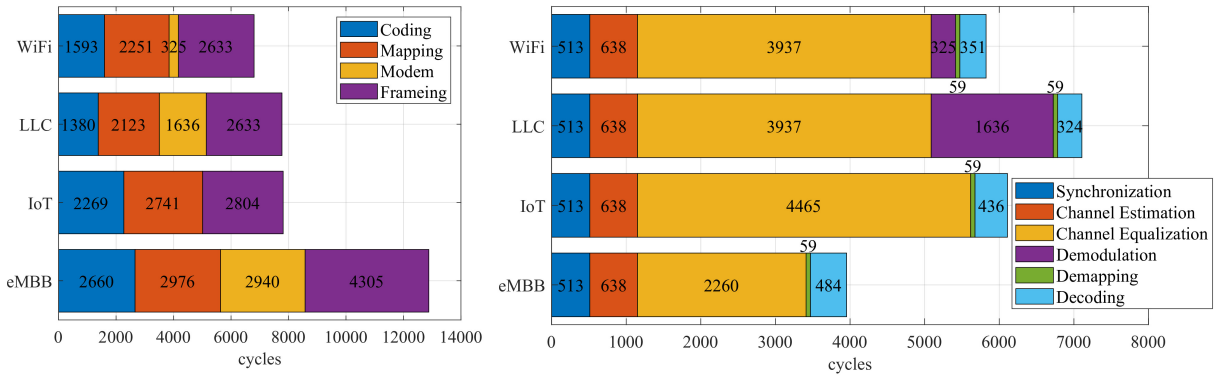
$$L_{\text{Receiver}} = t_{\text{Synch}} + t_{\text{Est}} + t_{\text{Equ}} + \frac{c_{\text{data}}}{c_{\text{Modem}}} \cdot t_{\text{Modem}} + \frac{c_{\text{data}}}{c_{\text{Mapping}}} \cdot t_{\text{Demapping}} + \frac{c_{\text{data}}}{c_{\text{Coding}}} \cdot t_{\text{Decoding}}. \tag{4.22}$$

The latency is reduced because the synchronization, channel estimation and channel equalization are running with a faster clock speed compared to most transmit signal processing blocks. The synchronization is continuously classifying the data symbols as preamble or data block. Prior to performing the channel equalization, the data block has to wait until the channel estimation is finished. Here, the amount of data blocks per preamble B has a influence which is not shown in the figures.

Fig. 4.17 shows the contribution of the individual sub modules onto the overall latency. Here the four different parameter settings of Table 2.1 are used. The module framing at the transmitter includes the preamble multiplexing, windowing and addition of CP and CS. Those steps take a longer time, because the whole frame has to be written to a memory before the first sample can be given to the output. Whereas the synchronization block at the receiver only needs to split up the frame. The modulation and de-modulation latency is the same, because the blocks perform the same operation. The mapping process takes longer time on the transmitter, because the resource mapper is programmed to wait for each sample individually from the different input sources to ensure that the total amount of samples is correct.

The overall latency of the implemented transceiver fulfills the requirements for the PHY layer of 5G networks, because the signal processing times are below the 100 µs limit. Fig. 4.18 shows the latency of transmitter, receiver and the time needed to transmit a frame with a given sample rate. Here the LLC configuration achieves the 100 µs requirement including the latency inducted by the over-the-air transmission.

Still, the latency could be improved by optimizing the clock speed of different modules using Equation 4.21 and 4.22. However, the compilation time of the FPGA design is up to 4 h, so the processes is time consuming. Further, it is difficult to achieve standard compliant response times of the 802.11a WLAN standard with the current implementation. This would require a tailored implementation, that is using FFT modules



(a) Transmitter **(b)** Receiver

Figure 4.17: Distribution of the latency of the FPGA signal processing depending on different configurations

fixed to a given size. Doing so, would optimize the latency of the FFT processing and allows to add the CP via the FFT core [166].[5]

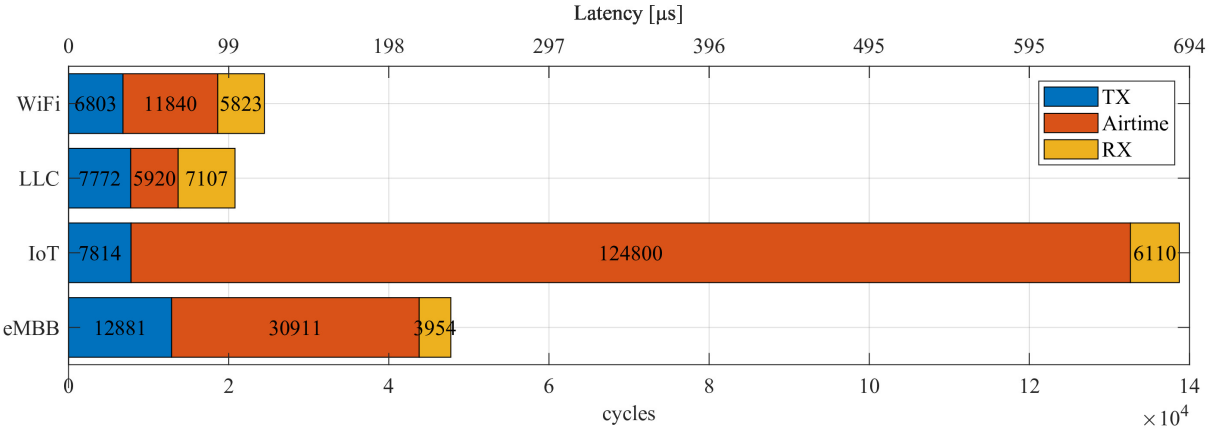


Figure 4.18: The overall latency of transmitter and receiver, including the transmission times using the respective sampling rate.

4.2.4 Resource Consumption

The estimated resource consumption in comparison with the commercial LTE and WIFI AFW is depicted in Fig. 4.19. The detailed resource usage of each signal processing module is shown in Fig. A.1. The resource utilization is estimated through compiling all components separately on the FPGA, where the compiler has more freedom to place the elements as in case of the full transceiver. Still, it allows an insight about the consumption of the various signal processing blocks, but the summation of all values might be different from the total slice count.

[5] This option is not available in case variable FFT lengths are configured.

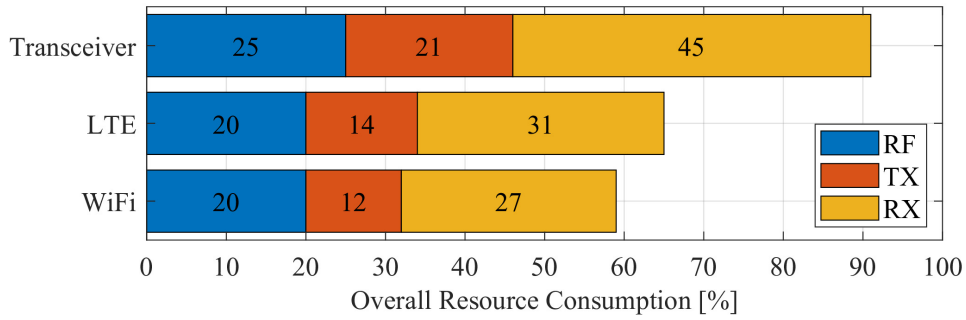


Figure 4.19: Comparison of the resource consumption of the transceiver with the LTE and WiFi application framework.

The overall slice count is an indicator how the FPGA resources are exploited, because one slice is the smallest entity in an FPGA [167] and consists of 4 look-up tables (LUTs), 8 flip-flops, multiplexers and carry logic. These elements provide basic logic, arithmetic and ROM functions based on the program. A *XILINX* FPGA arranges two slices in one configurable logic block that is connected via a switch matrix to the general routing matrix of the FPGA. In addition, specialized elements are part of the FPGA fabric, such as dedicated block memory, DSPs, registers and LUTs.

The slice count of the RF control/driver logic is slightly higher for the presented architecture, because it is kept untouched as much as possible with respect to the example programs. In contrast the commercial development, e.g. in the AFWs, is optimized for the specific use cases. Here, the intention is to ensure that the transition to newer driver versions is as smooth as possible because *NI* driver updates are seldom documented and introduce new behavior, which has to be reverse engineered to adapt the implementation. Further, buffering modules have been added as part of the MAC, to ensure that the frames can be transmitted in a given time grid and on request in continuous fashion.

The signal processing in the commercial AFWs is organized in different clock domains: One for the RF driver, one for the transmitter and one for the receiver, in contrast to the transceiver. Here each sub-module is part of a separate clock domain to allow individual optimizations, therefore, the resource consumption of the presented implementation is higher. In addition each sub-module is implemented using a flexible architecture, which depends on many programmable memories and flexible FFTs, in contrast to the AFWs where all parameters are tailored to a given waveform setting. So, both AFWs leave much more free space on the FPGA for further additions, e.g. such as better coding or modulation schemes. However, the structures inside both AFWs are tailored to the standards and are very restrictive in terms of timings for instance. Hence, it is difficult to exchange core modules such as the modem because of the tight coupling between the components [138]. Still, the present flexible architecture fits well into one Kintex 7 FPGA inside the USRP X310 platform.

4.3 Discussion

The implementation provides low latency PHY signal processing to evaluate different concepts for industrial networks, since all configurations are below the 1 ms time constraint, without considering higher network layers. Still, calculating an acknowledgment within the 10 us for standard-compliant WLAN is not possible with the current implementation and would require further enhancements, such as preparing the acknowledgment as soon as a new packet is received and disrupting the ongoing acknowledgment transmission in case the CRC fails.

In general, the amount of parameters introduced by this concept can also be a challenge, since many parameters depend on each other and require fine tuning. For instance, the FFT inside the modulator is based on radix-2, such that the sub-carriers K are limited to power of two values. The FFT for the channel equalizer in a GFDM setting needs to perform the equalization with respect to sample count of a whole frame N. So, the amount of sub-symbols M has to be chosen accordingly. Significant time has been spent on integrating all the modules and on the fine tuning of all the parameters.

4.3.1 Extension to MIMO

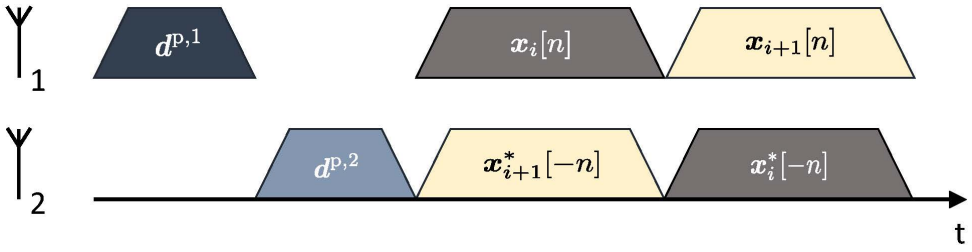


Figure 4.20: The TR-STC GFDM frame structure.

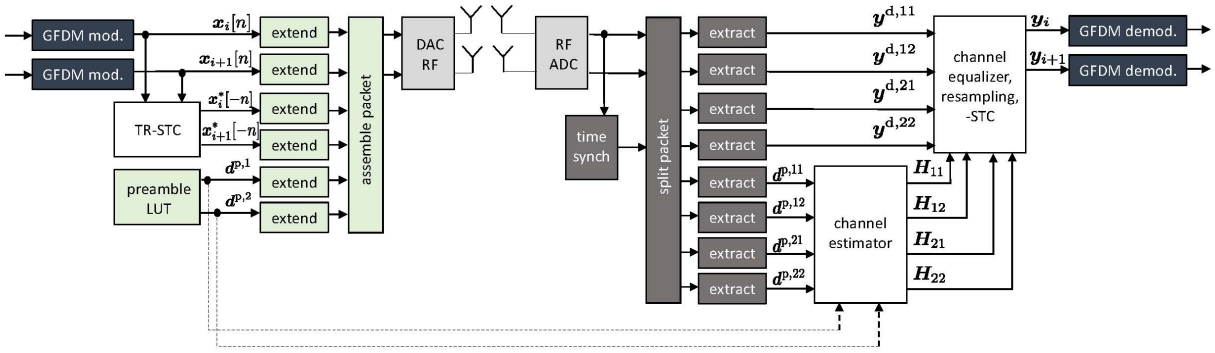


Figure 4.21: 2 by 2 TR-STC transceiver block diagram. The colored modules can be reused from the SISO transceiver with minor modifications. Adapted from [149].

The developed transceiver is limited to single-input and single-output (SISO) schemes only. However, many modern communication systems are using multiple input multiple

output (MIMO) techniques to increase the robustness and/or data rate. Thus, the thesis author outlines in [149] how the implementation can be extended with 2 by 2 MIMO capabilities based on Alamouti space time coding [168].[6]

Fig. 4.20 shows the frame configuration and drafts the working principle. At the beginning of the radio frame, two preambles $\boldsymbol{d}^{\mathrm{p},1}$ and $\boldsymbol{d}^{\mathrm{p},2}$ are sent out, one for every antenna. This allows the estimation of the four channel realizations, because only one preamble is sent out at a time whereas the other channel is kept silent. Two data frames $\boldsymbol{x}_i$ and $\boldsymbol{x}_{i+1}$ follow the preambles. The first transmitter port sends the frames in a regular order, but the second transmitter port reverses the order of the frames. Furthermore, the sample order in each frame is reversed and each sample is transmitted conjugate complex.

Fig. 4.21 indicates the increased effort for the implementation. At the transmitter side, the two data frames have to be created either consecutive or in parallel. In the following, a time-reverse space time code (TR-STC) module has to prepare the specific sequence for the second antenna port. This is a process similar to the interleaver, where the samples are written to a memory and read-out in a different order. A CP and/or CS needs to be applied to all sequences before the final transmit signal can be assembled, including the two different preambles.

The receiver synchronizes to the transmitted packets. The demonstrator in [149] is searching on the first receive antenna port for the first preamble $\boldsymbol{d}^{\mathrm{p},1}$ to conduct the detection process. After the two data streams have been split into the different sequences, the frequency response of the channel has to be estimated for each combination $\boldsymbol{H}_{11}$, $\boldsymbol{H}_{12}$, $\boldsymbol{H}_{21}$ and $\boldsymbol{H}_{22}$. Now, the received signals can be combined in frequency domain as follows to achieve full diversity gain:

$$\begin{aligned}\hat{X}_i[f] &= \boldsymbol{H}_{eq}^{-1}[f] \sum_{l=1}^{L} \boldsymbol{H}_{1,l}^{*}[f]\boldsymbol{Y}_{i,l}[f] + \boldsymbol{H}_{2,l}[f]\boldsymbol{Y}_{i+1,l}^{*}[f] \\ \hat{X}_{i+1}[f] &= \boldsymbol{H}_{eq}^{-1}[f] \sum_{l=1}^{L} \boldsymbol{H}_{1,l}^{*}[f]\boldsymbol{Y}_{i+1,l}[f] - \boldsymbol{H}_{2,l}[f]\boldsymbol{Y}_{i,l}^{*}[f],\end{aligned} \tag{4.23}$$

where

$$H_{eq}[f] = \sum_{j=1}^{2} \sum_{l=1}^{L} |H_{j,l}[f]|^2 \tag{4.24}$$

and L is the number of receiving antennas. After the conversion to time domain, the two estimated received data signals $\boldsymbol{y}_i$ and $\boldsymbol{y}_{i+1}$ are demodulated as described before.

The practicability of this approach has been demonstrated in [149] where the signal processing was carried out on a host computer without a realization on a FPGA. Further, this idea has been pursued into a real-time prototype as part of the 5GRange EU-Project. To achieve real-time operation and deal with the increased processing complexity, one USRP SDR platform is used as transmitter and one platform as receiver. The research

[6] The applicability of this principle to GFDM based signal processing was introduced in [169].

institute *INATEL* lead the implementation works and reused the presented modulator and demodulator implementation in a cognitive radio (CR) demonstrator utilizing TV white space (TVWS) in Brazil. The aim of this project was to provide internet connections for schools and farms in rural areas without dedicated frequency bands. Thus, gaps in the frequency bands allocated for digital television were used for the communication system. The flexible implementation using the GFDM framework successfully helped to deal with the different challenges:

- ▷ First, the television broadcast should not be harmed. Thus, the low OOB capabilities are enabled by using the pulse shaping filters.
- ▷ Secondly, a adaptable CP length deals with the long transmission ranges required to cover rural areas.
- ▷ Finally, Alamouti space time coding increased the robustness of the transmission.

However, MIMO techniques are not necessary to prototype low latency applications for industrial networks, since the data rates are well within the capabilities of the SISO chain and robust communications can be achieved by repeating the messages as well. Further, the increased effort to achieve MIMO real-time communications makes the integration with practical applications more difficult. For example, a mobile robot platform would have to provide energy for two SDRs. Still, the access points in the testbed should enable MIMO experiments.

4.4 Summary

This chapter presented a flexible transceiver implementation. The results can be summarized as follows.

1. The base-band processor design allows parameter reconfigurations to emulate several waveforms during run-time and not design-time.
2. The numerical precision using the proposed design flow matched the floating-point simulation, without the need for detailed fixed-point models.
3. The expected latency of the transceiver implementation is well below the 1 ms constraint defined for 5G networks.
4. The development fits into one XILINX Kintex 7 FPGA inside the USRP X310 SDR platform.

The flexible reconfigurations allow changing the number of sub-carriers and sub-symbols, the length of the CP and CS. In addition, arbitrary pulse shaping filters or receive filters, windowing functions and preambles are supported. With the proposed transceiver

architecture, different waveforms can be generated by simply reconfiguring the device without the need to provide a specialized implementation. Hence, PHY level real-time evaluations using different waveforms become feasible to understand their benefits and limitations. Further, the presented transceiver design allows to change its waveform configuration quickly and therefore supports research on mixed numerology applications.

To evaluate this concept in a more practical context, a testbed is needed to provide the respective environment, which is presented in the next chapter. The measurement results using the presented PHY implementation are discussed in the sixth chapter.

Chapter 5

Testbed

The performance of the presented transceiver structure was evaluated under laboratory conditions so far. However, to be able to draw conclusions with practical relevance, environments beyond ideal lab circumstances need to be considered. Therefore, a testbed is set up, with three aims:

- Real-time experiments
  - by providing access to the FPGAs inside the Software-defined radio (SDR) based nodes
  - with support through powerful server and cloud platforms.
- Experiments in (more relevant) environments
  - outside the typical laboratory conditions and use
  - indoor and outdoor locations on the campus of the university.
- The freedom of tools is supported
  - by providing customized firmware images for the SDR platforms as well as for the control servers
  - to provide a standard independent test ground.

In addition, three preconditions are driving the architecture of the testbed:

First, the German frequency regulator Bundesnetzagentur (BNetzA) is granting trial licenses on an annual base. For example, until 2016 the testbed license covered up to 100 W equivalent isotropically radiated power (EIRP) with a 20 MHz bandwidth around the center frequency of 1.99 GHz for the uplink and 2.18 GHz for the downlink. Later, the license covered 10 W EIRP with 100 MHz bandwidth at 3.75 GHz, which got reduced to 5 W since 2019. The transmission scheme is time division multiple access (TDMA). Further, the license has to be renewed every half year. As a consequence, the radio frequency (RF) front-end and potential power amplifier (PA) should be as broadband as possible. This way, a change in frequency only affects antennas or bandpass filters.

Otherwise, a major revision is needed to equip the front-ends with a different frequency range. Despite the tedious re-application process, the 3.75 GHz band is available in Germany for 5G campus networks and is close to the commercial 5G frequencies and therefore very attractive for experimental use cases. In addition, mobile nodes are allowed moving in a 10 km radius around the base station sites for transmission experiments.

The second precondition is that it is convenient to have the permanently mounted indoor and outdoor locations close to each other, despite having the option to operate in a large area. This way interactions between most of the nodes are possible and the maintenance overhead can be reduced. This allows readjusting or replacing components with reasonable effort. Especially in city scale testbeds, getting access or permission to install different devices on roof-tops or lantern posts can take awhile. Thus, a flexible structure is proposed that allows adjusting the testbed to the experiment while gaining understanding of novel techniques in a known environment. This way the testbed can be seen as a extension of laboratory trials. Further, the aim of the testbed is to act as a playground for experimentation of novel waveform concepts and to enable research on wireless networks ranging from the physical layer (PHY) layer to the application layer. Nonetheless it cannot replace extensive field trials. As a result, the intention is to keep the design of a node to be relatively compact, such that it can be taken out to relevant environments such as the portable testbed presented by IMEC.

Finally as part of the nature of prototyping new ideas, even the best prepared hardware or software infrastructure might be state of the art only for a short amount of time. For instance, the first military SDR solution ”speakeasy” was tied to a certain DSP processor with a certain C programming flavor. The processor got outdated after 2 years due to the next digital signal processor (DSP) generation, while the development process was still ongoing [170]. Thereafter, all code had to be reimplemented for the up-coming generation. Therefore, the management part should be as simple as possible to do not tie the experimentation to certain processes. The testbed functionality is therefore split into a testbed backend, providing resources such as machines and network connections and tools for experimentation. A simplified communication plugin for the automation of experiments connects all components loosely when required. Further, the SDR hardware is based on the USRP platform because it supports a broad range of software development tools as well as applications.

The contribution in this chapter is based on work published by the author [76] and describes the developed infrastructure, the used hardware and software components. Further, this infrastructure is funded by the Federal Ministry of Education and Research as FAST Testbed [171]. In addition a strategy is given to evaluate the performance of the platforms from a digital signal processing perspective.

5.1 Infrastructure

The cellular testbed [139] has two types of nodes mounted permanently:

1) Six indoor nodes mounted on the ceiling, referred here as access point (AP).

2) Four outdoor sites as base station (BS) mounted on the roof-top of the university buildings.

The indoor nodes can hold one set of SDR devices, whereas the BS can hold multiple. The geographical layout of the testbed at Barkhausen-Bau of TU Dresden is shown in Fig. 5.1. Besides the mounted nodes, two laboratory rooms are connected to the testbed to flexibly add further devices. Further, several bicycle rickshaws and one robot platform are available as mobile nodes for manual experimentation.

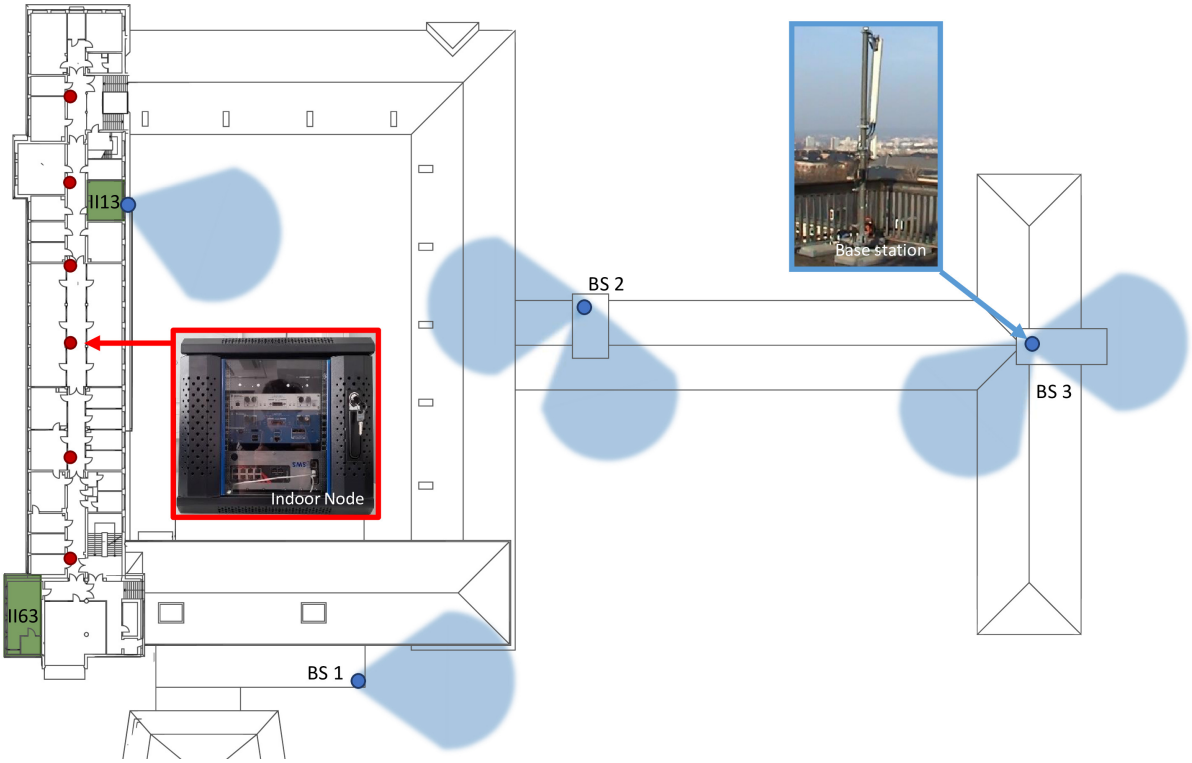


Figure 5.1: Layout of the proposed testbed, showing six access points and two laboratory rooms on the second floor. The four base stations are on roof-top level with their respective sectors highlighted in blue. The map of the building is taken from [172].

All nodes are connected to a central laboratory room, labeled as II63 in Fig. 5.1, via a set of fiber cables (4 for indoor, 8 for outdoor). The idea is to have at least two independent connections per node, for a possible split between data and control planes. A *CISCO* 9500-48Y4C switch interconnects all components to the server back-end as indicated by Fig. 5.2. It has 48 connectors for 1 / 10 / 25 GBit fiber modules. Further, it can aggregate the signals via 100 GBit uplink ports to a server room, with another switch to attach different cloud platforms for experiments. This server room provides two server

platforms, one acting as an experiment controller and storage. The other is intended as a computing platform. For instance, *HPE* edgeline m710 blades can be booked to enable edge cloud computing as part of an experiment. Further one set of SDR devices is kept in the laboratory, without any further RF components, with the aim to enable basic transmission experiments using cables between them to emulate additive white Gaussian noise (AWGN) conditions.

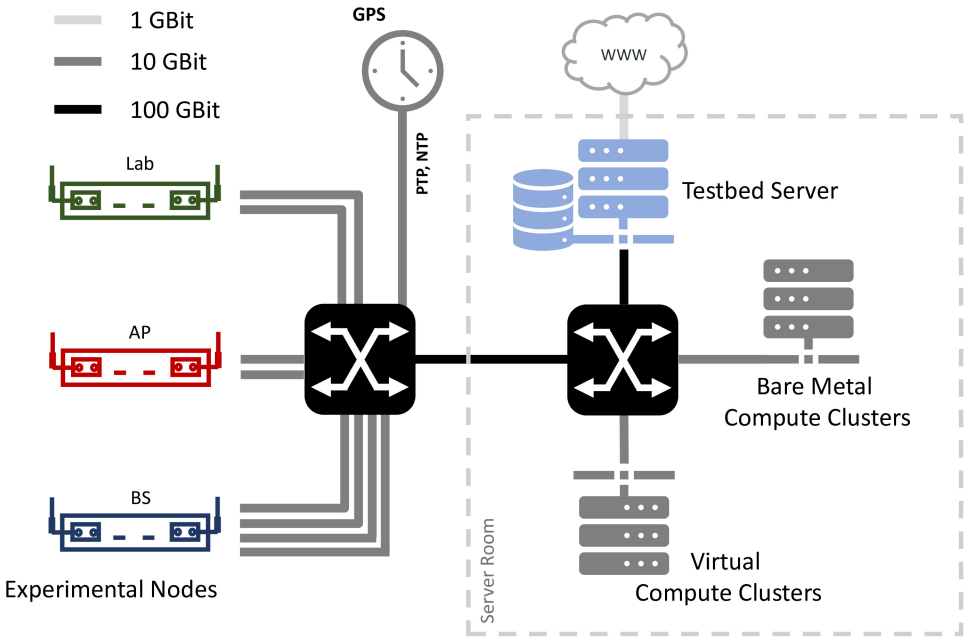


Figure 5.2: Overview of the network and server infrastructure

The software infrastructure is reduced to core components compared to extensive frameworks such as Orbit Management Framework (OMF) [129]. Fig. 5.3 depicts the overall concept. Each project gets a virtual machine (VM) to include all software, firmware, drivers or experimental data. Hence, each project is responsible on its own to organize the experiment. This method grants the freedom to use any available tools, even such as OMF. The testbed management consists of a booking platform and focuses only on powering on the booked resources, such as nodes or compute platforms and interconnecting them with the VM. A default version of the VM can act as a starting point for new experiments. It offers a mechanism based on Clonezilla [173] to copy stored hard disk images from the VM to the booked resources before and vice versa after a experiment. Clonezilla supports any operating system, because it copies the hard disk byte for byte without interpreting the content.[1]

This concept allows to freeze an experiment, with respect to the software components, to

[1] The boot process of Linux always detects hardware changes in case a harddisk image is copied to a different machine. In general Windows follows the same principle, but gets stuck if the harddisk controller is changed or secure boot options are enabled. However, Windows can be forced to forget all customized hardware settings during the shutdown. This way also Windows can cope with hardware changes.

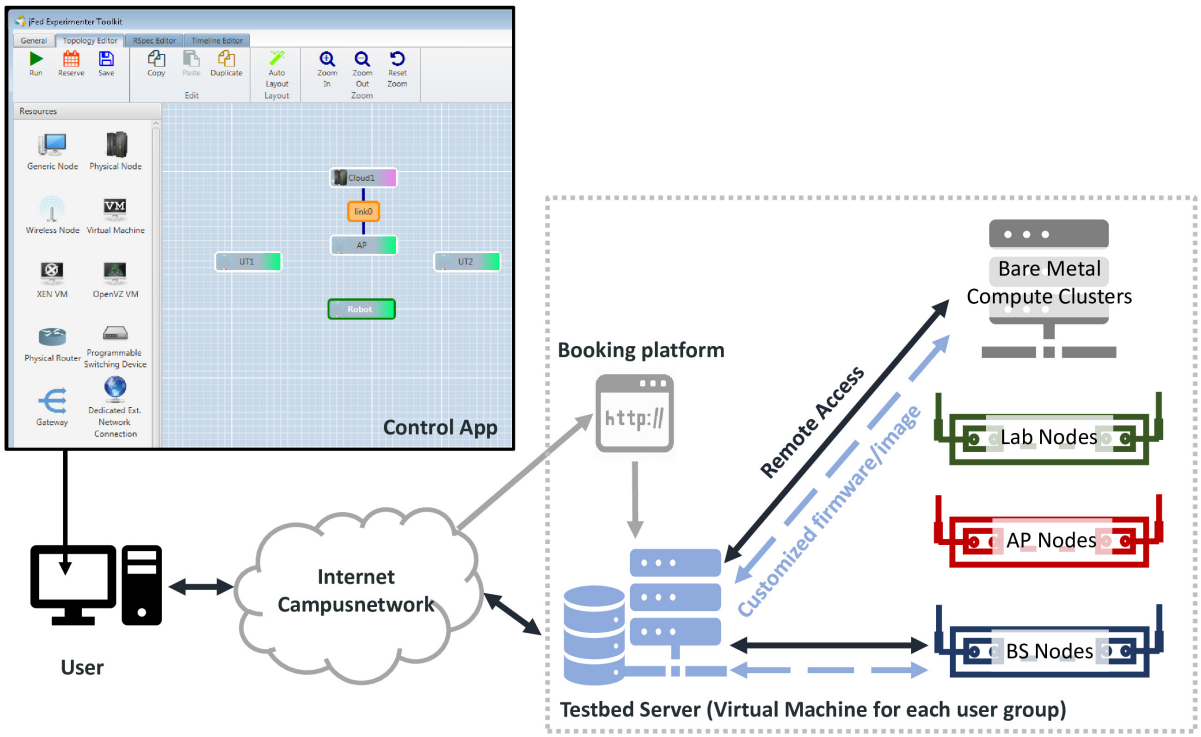


Figure 5.3: OWL control architecture

a given instance in time and to replicate it on request. Thus, experiments are repeatable, unless the hardware set-up is changed. Further it enables sharing the hardware between different use cases and user groups, because it is independent of the operating system (OS) or drivers etc.

5.2 Automation

Experiments need to be repeatable to verify the behavior of the components. The testbed control framework OMF provides an approach called Orbit Measurement Library (OML) that is using scripts to parametrize the individual components [174]. On each experimental node, a service daemon collects all information and sends them to the control network and to a SQL database, for example. The approach used here, separates and connects all components via a multicast user datagram protocol (UDP) called TestMan. It is using interprocess communication (IPC) principles [175] to set-up a micro service architecture [176] for experiments.

The first aim of IPC is to simplify and standardize the communication between different applications in a network [177]. For easier integration the protocol TestMan has been developed into a C# based Dynamic Link Library (DLL) as a plug-in for various applications. However, the intention is to still keep the protocol as simple as possible, such that it can be integrated easily in software development suits like LabVIEW by reimplementation, if it is necessary. In contrast, relying on Middle-ware, such as presented

in [178] for the CERN institute, might not be possible for small development groups, if in the worst case the protocol has to be reimplemented for a different software platform.

The second aim is to distribute various measurement task into different independent applications instead of monolithic solutions. This has the advantage, that the developed programs are functional standalone and can be recombined depending on the needs for the experiment. For instance, an application, informing the network with global positioning system (GPS) status messages, can be used for localization as well as for time tracking purposes. When monolithic software is to be used, then this feature would have to compiled every time into the overall program. Further, it simplifies the development process if an application focuses on an specific function instead of multiple, because a unit test would have to check all combinations. In addition, once an update is needed only one certain application is affected.

The own protocol used, called TestMan, transfers dictionaries with key – value string pairs, which can encode any data. Further it introduces addresses for applications, instead of Internet protocol (IP) addresses for computers. Several applications can be grouped to exchange commands or status information. TestMan uses UDP broadcast as transport layer to spread the messages through the network. This has the advantage, that every physically attached network device can listen to this messages, even if the IP address is not configured accordingly. However, UDP broadcasts can be dropped anytime by the network. Therefore two message types are foreseen, status messages and commands. Commands are using a four way handshaking to receive an acknowledgment for the reception and for the execution. So, the sender application knows if a command has been received and executed by the desired application. In contrast, status messages can be dropped by the network, because information such as provided by GPS will be repeated anyway. The main intention is to use TestMan to configure and control experiments, because the handshaking protocol is not designed for throughput or latency. To avoid those limitations, Point-To-Point links for high-throughput transfers can be set-up individually using the transmission control protocol (TCP) protocol. Here, TestMan will initiate the connection.

Fig. 5.4 gives an overview how a complex demonstrator system can be controlled, combining very different languages and operating systems. Here, every instance is a standalone operating program constantly exchanging run-time information. Section 6.2 provides further details.

5.3 Software Defined Radio Platform

The hardware in the testbed is selected to be as generic as possible, but to provide sufficient signal processing power. The USRP-RIO series from *National Instruments (NI)* accomplish that by connecting a *Xilinx* KINTEX 7 field programmable gate array (FPGA)

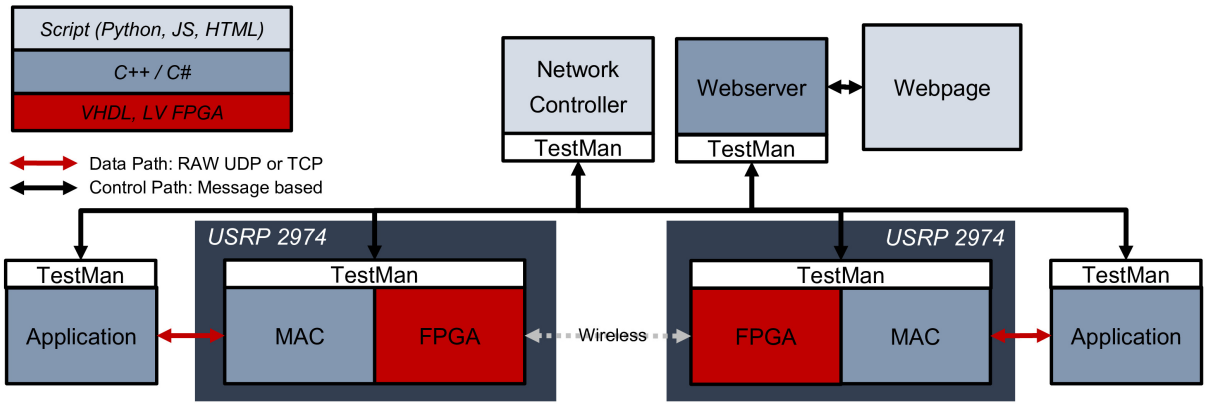


Figure 5.4: Testbed control using UDP broadcast as control path and one-to-one links for data transfers, where the TestMan plugin is used for control information.

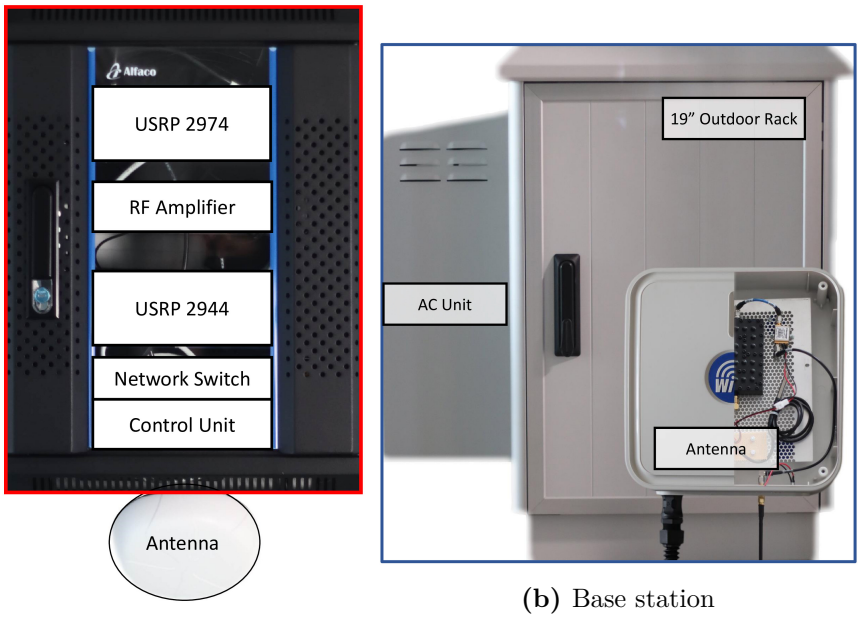
to two independent, broadband radios.[2] Therefore many open software frameworks are compatible [180] as well as the closed source development suite LabVIEW. A control PC can be connected by means of PCI Express extension or via the two 10 GBit Ethernet connectors. A new generation under the brand name USRP 2974 provided by NI includes an embedded PC attached to a USRP X310 in a standalone device.

As shown in Fig. 5.5, each node consists of one USRP-2974 and one USRP-RIO to provide two independent SDR platforms. Further, the USRP-2974 encompasses an Intel i7 host platform, where each USRP is connected via PCIe. To enhance the SDR platform with sensing capabilities and to monitor the usage of the spectrum, one additional USRP B205-mini is attached to the USRP 2974 via USB3. In addition, one 10 GBit switch is used to interconnect all components inside the node.

The selected USRPs provide a frequency range from 10 MHz to 6 GHz. Therefore, the aim in the next section is to provide each SDR with a broadband RF front-end, which is powerful enough for micro cell experiments. The covered bands should include ideally both WiFi-bands and the 5G band for local use cases, around 3.75 GHz. This enables wireless experiments in licensed as well as unlicensed bands and thus, allowing to use all major transmission schemes such as LTE, 5G, 3G, WiFi, Bluetooth or ZigBee for instance.

Frequency and time stability amongst the SDR can be a determining factor for experiments concerning coordinated multi point (CoMP) techniques [181]. The EASY-C project utilized *Meinberg* GPS receivers to convert the GPS signaling into a 10 MHz sine reference and a pulse-per-second signal (PPS). The 10 MHz sine signal is used to generate the center frequency, whereas the PPS is used as a time trigger to initiate a transmission process in a regulated time grid. Unlike EASY-C, the OWL testbed introduces indoor nodes, which do not have access to the GPS signals. As a consequence clock synchronization between the indoor and outdoor nodes is difficult.

[2] A *NI* branded USRP-RIO can be converted into an *Ettus* USRP X310 version and back [179], because only the firmware is changed.



(a) Indoor access point

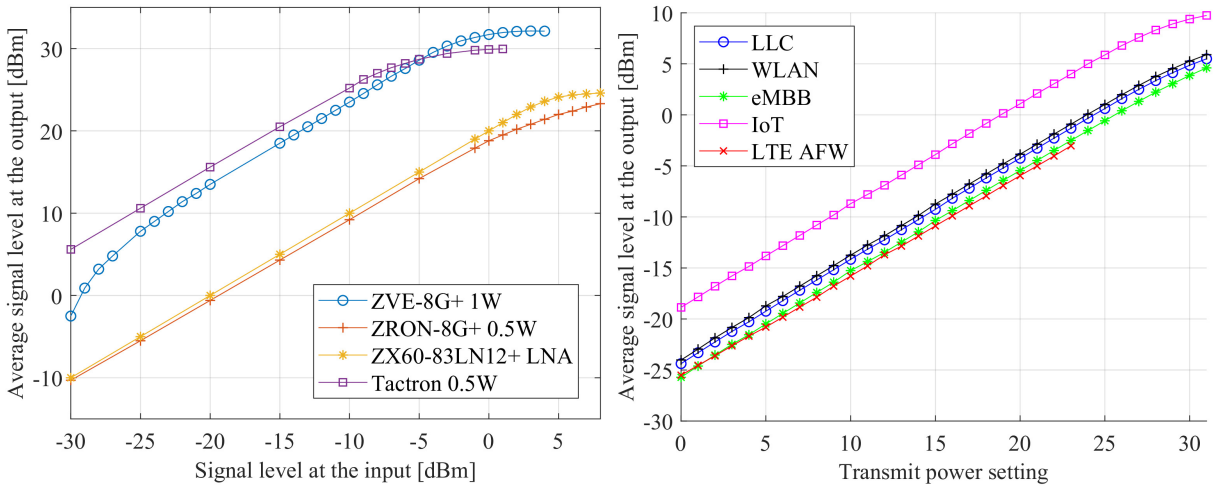
(b) Base station

Figure 5.5: Overview of the hardware components in the nodes

To overcome this limitation, the *FibroLAN* Falcon network switch series is used in the testbed. Here one μFalcon-MX/G switch acts as a precision time protocol (PTP) grandmaster clock mounted in one of the BSs that can receive GPS signals to stabilize its internal clock. With the help of the PTP network protocol, it is possible to synchronize other clocks in the network with a precision of several nanoseconds compared to the master clock. Thus, the μFalcon-MX/S switch is selected for all indoor nodes to act as a PTP client adapting to the grandmaster clock. In addition, it provides the 10 MHz and the PPS signal via hardware outputs to the USRPs. So, any SDR experiment can use these signals to experiment with CoMP or synchronous single-frequency network applications. Especially the built-in hardware PTP clocks allow a time synchronization as accurate as the time duration of the transmitted samples, without specialized software solutions to synchronize the FPGAs.

5.4 Radio Frequency Front-end

Since the power output of the USRP degrades depending on the waveform (Fig. 5.6b) due to the peak-to-average power ratio (PAPR) and frequencies, a matching power amplifier is needed to provide sufficient transmission power for the usage in the testbed. The goal is to provide 23 dBm average transmit power, close to the 5G uplink specification and to compensate potential cable and filter losses. Further, the selected components should be standalone useable as well.



(a) Amplifier characteristics based on sinusoidal input using a signal generator

(b) USRP transmit power using different waveform configurations

Figure 5.6: Measured average transmit power

5.4.1 Sub 6 GHz front-end

Fig. 5.6a compares different broadband power amplifiers. However, one drawback of using a transmitter (TX) power amplifier is the sensitivity of the receiver, as introduced in Fig. 2.1. The maximum allowable USRP X310 input power is 10 dBm and for the small USRP B205 mini 0 dBm. Thus, the isolation between transmit and receive chain must be at least 15 dB. To achieve this, several options are available:

- ▷ Typical frequency multiplexing system cover this using band-pass filters and a frequency gap between uplink (UL) and downlink (DL).
- ▷ In radar applications also RF circulators are used.
- ▷ Using a switch to select the operation mode is common in TDMA systems.

Several band-passes and switches were tested during the planning phase of the testbed. The common requirements are: 1) capability to handle 1 Watt of input power, 2) a low insertion loss of less than 3 dB, 3) enough isolation between the transmit and receive part and 4) being broadband for flexible usage. To provide very generic and reproducible results, commercial off-the-shelf (COTS) equipment is to be used rather than customized front-ends. Considering low-cost also for the selection of the components, the following options are viable:

- ▷ *G-Way Microwave* offers a custom filter design that meets the requirements with more than 50 dB isolation, however the front-end will be bandwidth limited and the UL and DL frequencies have to be fixed.
- ▷ Circulators are typically expensive and only military-grade components meet the requirements, most available products deliver an isolation between only 14 and 25

dB. A mismatch at the antenna reflects the signals back and degrades the isolation further.

- ▷ RF switches are available in large quantities, but much fewer can offer the specific requirements while keeping a sufficient switching speed.

The two candidate switches ZSW2-63DR+ and ZFSWA2R-63DR+ from *Mini-Circuits* seemed to be a viable option. The first one is a reflective switch, that appears as an open load to the amplifier and could therefore inflict damage. Whereas, the other can barely handle 1 Watt power, but provides much better isolation at the cost of transmission loss. In addition, this switch is much faster and can change the state within 35 ns. However, in both cases the FPGA has to control the switch for transmission or reception, because the host computer would not be fast enough. So, every software solution utilizing the platform would have to be adapted, which is infeasible.

A basic method to provide isolation between TX and receiver (RX) is to place the respective antennas with distance from each other. Omni-directional antennas, like the *MobileMark* MGRM-WHF, have their "blind" spots along the axis. Placing them vertical to each other provides an isolation of around 30 dB, depending on the environment. The indoor nodes use the four by four MIMO antenna LGMQM4-7-38-24-58 from *Panorama antennas* as shown in Fig. 5.7. It has four antenna elements intended for 1.7 to 3.8 GHz and four antenna elements for 2.4 and 4.9 to 6 GHz. The USRP transmitter with the *Tactron* PA is connected to the first set of elements and all receiver front-ends to the second set of antenna elements. The isolation between both is around 27 dB. Additionally, peaks are prevent from reaching the receiver by using a 10 dBm limiter.

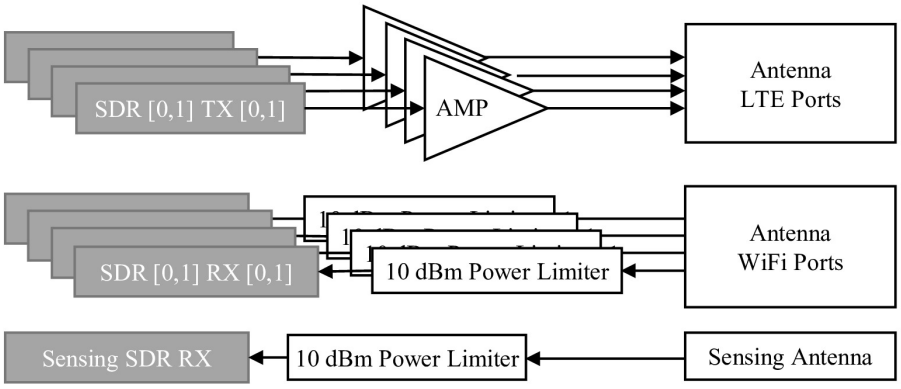


Figure 5.7: RF circuits used in the APs.

In principle, the set-up for the BSs does not differ. Except that the BSs have a set of sectorized antennas (*Wireless Instruments* WiBOX PA M3-14X). Fig. 5.8 visualizes the set-up. The *Tactron* PAs and additional customized bandpass filters are mounted inside the two by two MIMO antenna. A receive low noise amplifier (LNA) from *Mini-Circuits* is mounted in the receiving antenna. This way both antennas can be placed with a given

distance to provide the isolation. Further, both amplifiers are mounted inside the antenna to cope the losses through the RF supply cable and a lightning protector.

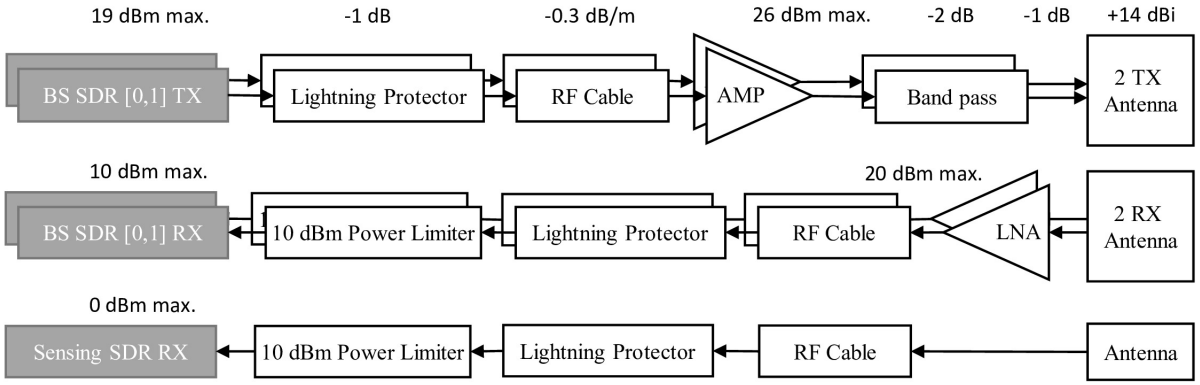


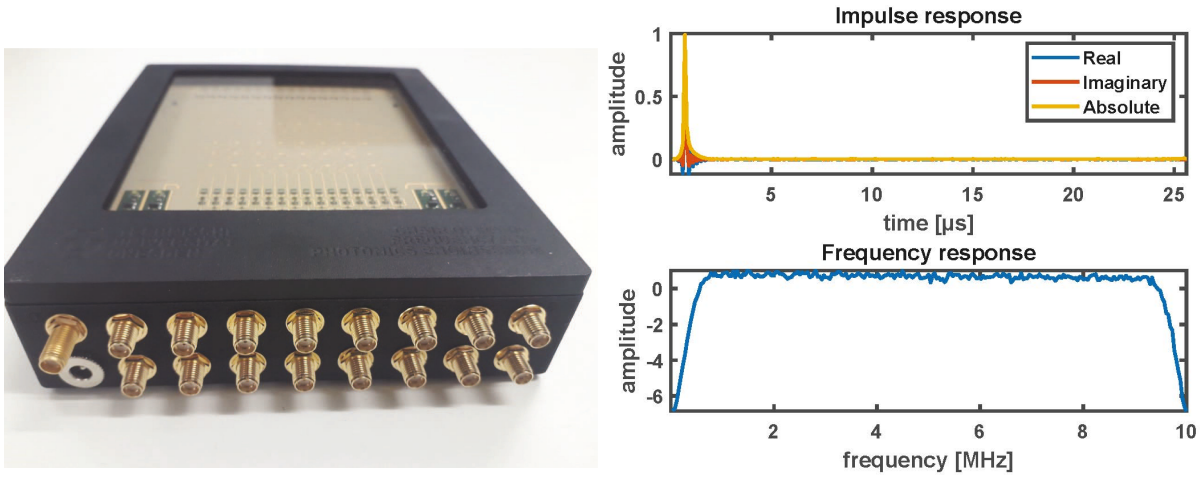
Figure 5.8: RF circuits used in the BSs, where the amount of TX and RX is doubled, due to the second USRP. The PA, band pass filter and LNA are mounted inside the antenna. The numbers indicate the power levels or gains according to the data-sheets.

5.4.2 26 GHz mmWave front-end

One novelty of future mobile communication systems is the employment of Millimeter-Wave (mmWave) bands. The advantages of this new technology are key to improve the network capacity in 5G networks, because they allow the use of larger bandwidths than the conventional sub-6 GHz systems.

Therefore, a hardware setup which combines conventional USRP platforms with mmWave frontends [182] is proposed to explore this novel frequency bands with experimental studies. The specialty of the frontend is, that it is possible to use the signal provided by the analog frontend of the USRP as intermediate frequency input for the mmWave system and vice versa for the receiving side. So any 2.4 GHz signal source and sink can be used for mmWave experimentation. Further, the frontend enables beamforming experiments without requiring additional control signals.

As described in [105], the 2.4 GHz signal is up-converted to 26 GHz before entering the passive Butler matrix beam-forming network. The Butler matrix routes the inputs similar to a Butterfly graph for a fast Fourier transform (FFT) algorithm through several passive delays and attenuators to form the individual beams. The PAs amplify the signal, before the quasi-yagi antennas radiate it. Although the Butler matrix can operate bidirectionally the amplifiers fix the frontend either to transmitter or receiver. Thus, the mmWave receiver frontend first amplifies the signal on each antenna via LNAs, and feed them into the Butler matrix and down-converts the output to 2.4 GHz. Fig. 5.9a illustrates the developed frontend. The operating frequency ranges from 26 GHz to 30 GHz. This covers the 5G NR frequencies for Europe (26 GHz) as well as for US and Asia (28 GHz).



(a) Picture of the hardware

(b) Measured channel impulse and frequency response

Figure 5.9: mmWave frontend developed by the Chair of Radio Frequency and Photonics Engineering of the TU Dresden. Based on [150].

5.5 Performance evaluation

Prior to judging RF front-ends, PAs or antennas, a methodology is needed to understand the expected performance from the perspective of the signal processing in the digital domain. The reason is that RF measurement equipment is normally expensive and accurate and as a consequence it is showing the best case performance, without reflecting the capabilities of the SDR front-end, which digitizes the analog signal. Therefore, a channel sounder based on a chirp-based sequence is implemented due to its advantageous auto-correlation property, serving as an SDR based network analyzer to measure the equivalent channel seen by the digital signal processing. The channel sounder used here is based on [76] and described in Section A.3. Later, it was extended into a real-time system which is detailed in [183].

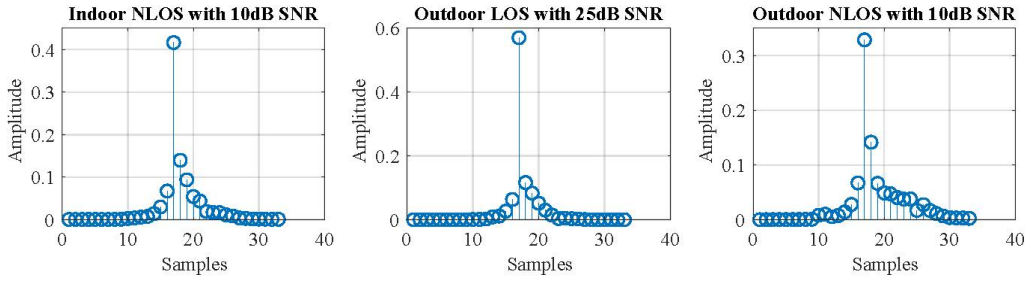


Figure 5.10: Average equivalent channel impulse responses for the different categories and the averaged SNR, measured with a bandwidth of 20 MHz, based on [76].

Thus, prior to setting up the whole testbed, the expected performance is estimated for the different locations using the developed channel sounder. The small sized USRP B205 mini

are utilized because they are battery powered via USB3 from a notebook and hence do not need a finished infrastructure. GNU Radio [133] is used to create the chirp sequence on base-station one. The user-terminal (UT) records all measurements for offline processing to define the signal-to-noise ratio (SNR) and channel impulse response (CIR).

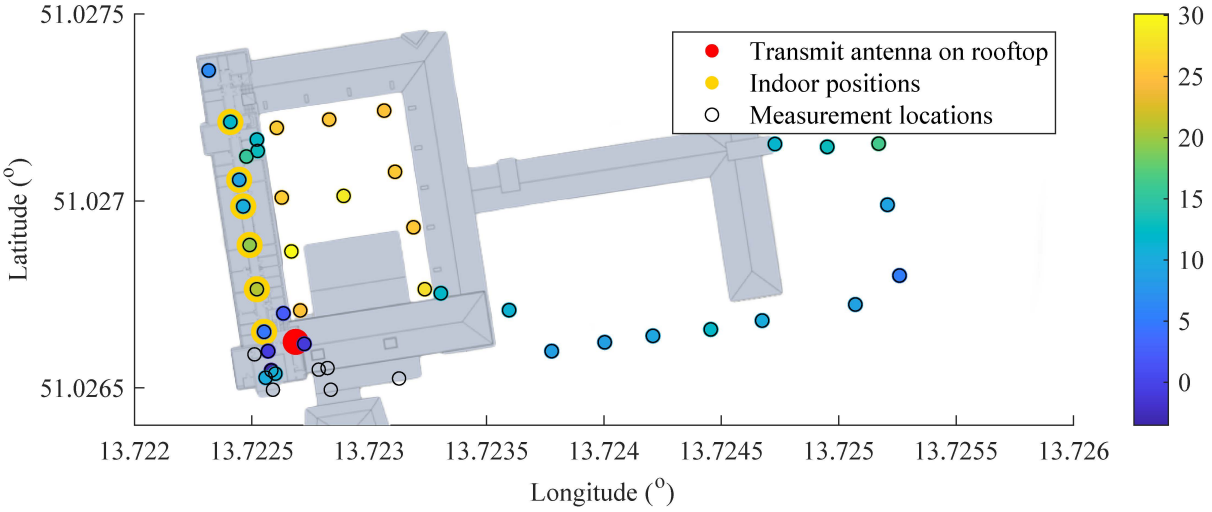


Figure 5.11: Link quality measurement, using the map of the Barkhausen building from [172] and results from based on [76].

The captured channels are depicted in Figure 5.10. The channels are averaged for indoor & outdoor, and for the non- & line-of-sight (LOS) case including the average SNR. In the outdoor LOS case, a single strong tap appears, as well as in the indoor case, because the weak paths are blocked by obstacles like windows or walls. Whereas in the outdoor non-LOS case, a strong tap, a weaker second tap, and a few local scattering taps are present. The measured SNR values are shown in Fig. 5.11. The average SNR for the indoor nodes for the given BS is around 9 dB, where already the windows have an attenuation larger than 25 dB. The yard of the institute building has a good coverage of 25 dB SNR for mobile experiments and the coverage around the building is sufficient for experiments with 10 dB SNR.

The high bandwidth of the USRP platforms of up to 160 MHz also increases the possibilities to capture interference which saturate the receiver. Fig. 5.12 shows the case where the presented transceiver is operating at 3.75 GHz, while a Long Term Evolution (LTE) transceiver is located at 3.7 GHz center frequency. Although, both communication systems have a guard band of over 25 MHz, the LTE harms the 3.75GHz such that no successful decoding of the data packets is possible. Consequently, performing frequency-division duplexing (FDD) as well as cognitive radio (CR) experiments is an issue with the current setup and needs filters or larger guard bands to ensure sufficient isolation. If a bandpass filter is used, like in most common communications hardware, it is possible to decode the packets again. This signifies that although novel waveform candidates like generalized frequency division multiplexing (GFDM) offer low out-of-band (OOB)

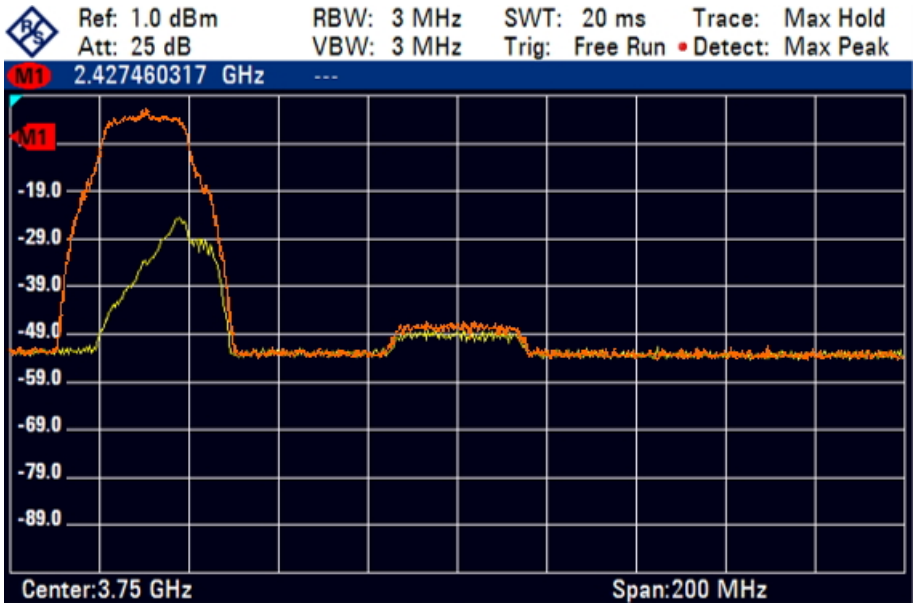


Figure 5.12: Interference with (yellow) and without (orange) bandpass filter.

features, but it is difficult to utilize them in CR or fragmented spectrum applications just by software solutions. The hardware has to be adapted in form of very sharp filters as well.

The performance of the mmWave frontend is evaluated too. For this reason, a *Rohde & Schwarz (R&S)* signal generator is connected at the transmitter instead of the USRP and and a *R&S* 40 GHz spectrum analyzer with an attached horn antenna is used as the receiver. This way the transmit power is calibrated to ensure that the power levels of a *R&S* signal generator, the channel sounder and the GFDM transceiver later are aligned. The maximum allowed input power for the mmWave frontend is 0 dBm. To leave enough head room, the input power at the input of the switch is selected to be 0 dBm, resulting in a power level of −6 dBm at the input of the front-end, due to the losses inside the RF switch. The distance between transmitter and receiver is kept at 2.5 m. Using the channel sounder, the CIRs are measured and depicted in Fig. 5.9b. It is observed that although the mmWave frontend is attached, the resulting channel seen by the SDR platform is reasonably flat, such that the developed PHY can be used without additional preparation.

5.6 Summary

The motivation for this testbed is to make real-time experimentation as simple as simulations. The following summary shows that:

1. The testbed can serve as a permanent experimental environment to test novel ideas continuously without extensive preparations beforehand.

2. An automatic mechanism allows repeatable experiments.

3. Remote access allows to include experiments from external partners without being physically present.

4. The layout of the presented testbed covers different aspects of wireless communication networks due to its: 1) Outdoor, 2) Indoor and 3) a Laboratory part.

This way, the Online Wireless Lab (OWL) testbed allows studies for very different application use cases:

- ▷ The outdoor scenario allows to perform cellular network experiments. For instance, to evaluate the suitability of different waveforms, e.g. OFDM, GFDM, Chirps. The back-end infrastructure enables experiments regarding Cloud RAN techniques such as CoMP. Especially since different split options are available, processing strategies can be evaluated.
- ▷ The indoor scenario enables experiments with respect to WiFi and mmWave networks, which are relevant for real-time control loop applications in industrial scenarios. Relevant for the same use case are hand-over experiments, either where the data is rerouted or where even the application instance moves. The x86 central processing unit (CPU) in each indoor access point can be used as a edge cloud, where the control loop application runs for example in a docker container. This docker container will be transferred to another access point in case the mobile robot moves out of range of the previous access point.
- ▷ The mixture of the indoor and outdoor scenario permits to study and compare device to device communications with cellular techniques. Further, each indoor node can be seen as an experimental small cell with different backhaul solutions, for instance via wireless (cmWave or mmWave) or via cable. Multi-RAT studies extend this scenario where each client has several wireless network links to increase the robustness of the communication. In addition, higher network layer experiments for scheduling and resource allocation or interferences between several transmission techniques can be probed. Network slicing is another possible experimental use case of the fully programmable testbed, consisting of software-defined networking (SDN) switches, SDR and cloud platforms. In addition, the presented channel sounding implementation can also be used to analyze the channel behavior over time or for localization studies.
- ▷ More basic experiments are possible using the laboratory set-up where most of the devices are interconnected via cables. This enables observation of wireless algorithms under AWGN conditions in the lab. Since powerful network devices are part of the testbed, trials regarding the general network performance are achievable as well.

This chapter described the OWL testbed facilities, which are used in the experiments presented in the following chapter. The aim is to evaluate the previously introduced transceiver. A list of the used RF components is given in Tab. A.3 and the used hardware components are listed in Tab. A.2

Chapter 6

Experiments

The aim of the transceiver and testbed presented in the previous chapters is to evaluate novel ideas for wireless networks in real-time. This chapter discusses selected experiments carried out using these structures from an application perspective. This implies that the transmission statistics are evaluated by using an test application, rather than evaluating them at the physical layer (PHY) implementation running in the Software-defined radio (SDR). The requirements of the applications on the network were introduced in the first chapter and comprise latency, data-rate and packet error rate (PER). Firstly, the performance of the communication between two nodes is evaluated. Secondly, the capabilities regarding multiple nodes are presented. Finally, an experiment concerning Millimeter-Wave (mmWave) frequencies is introduced. The contributions are based on the publications [184] and [150].

The latency is defined as the round-trip taken by a packet to travel through the network and back to the source [185]. As a consequence, the latency evaluation carried out in this work is close to the IEEE RFC 2544 specification [185]. However the parameters are adapted to the values given in Tab. 1.1 and Tab. 2.1, because RFC 2544 is defined mainly for cabled networks. There are newer standards for network performance testing, such as for example ITU-T Y.156sam [186], which are introduced with the goal of being closer to reality, especially in multi-service, multi-user environments such as operator networks. However testing under those conditions is out of scope for this work.

To create the test data, an application was implemented using C# and the .NET framework because it is platform independent. The application generates user datagram protocol (UDP) packets with a given rate, length and for a given duration. Each packet contains a sequence number, a timestamp and a cyclic redundancy check (CRC). This content is duplicated until the required packet length is reached. All packets have to be routed back to the test application to evaluate the latency and to not rely on time synchronization between several machines. This is accomplished by running another instance of this application on the client side, which mirrors the data back to the source. The source records all received timestamps to calculate the latency and to create a latency

histogram. Further, the same UDP packets are also utilized to measure the PER and throughput achieved during the experiment.

Of course, there are other packet generation options available such as iPerf, MoonGen [187] or using the *FibroLAN* switches of the testbed and various methods to optimize the packet rate [188]. The common issue to all is the requirement of a fully fledged network interface for the device under test, which is difficult to be implemented using LabVIEW. The prototype implementation supports only the reception and transmission of UDP data. Current off the shelf libraries in LabVIEW forward the content of the UDP packets without the Ethernet header to the field programmable gate array (FPGA).[1]

6.1 Single Link

In case of single-link analysis, the experiments are evaluated in different categories as depicted in Fig. 6.1. The first category covers the creation of the payload data and looping it back to the same test application introduced earlier. The second category passes the payload to the USRP SDR, however the payload is sent back without the signal processing on the FPGA. The third category includes the signal processing and the transmission process under additive white Gaussian noise (AWGN) conditions. The fourth category comprises a full end-to-end experimental set-up indoor and outdoor. The experiments in the following sections are organized after this scheme. In general, a *HPE* Edgeline 1000 controller with a m710 compute blade is used for the experiments, that encompasses a *Intel* Xeon E5-2618L v3 central processing unit (CPU) with 64 GB RAM. Further, Windows Server 2012 is used as host operating system with LabVIEW Communications 3.1. Depending on the experiment either a USRP X310 (*NI* part number 2944) or a standalone USRP 2974 is used as SDR.

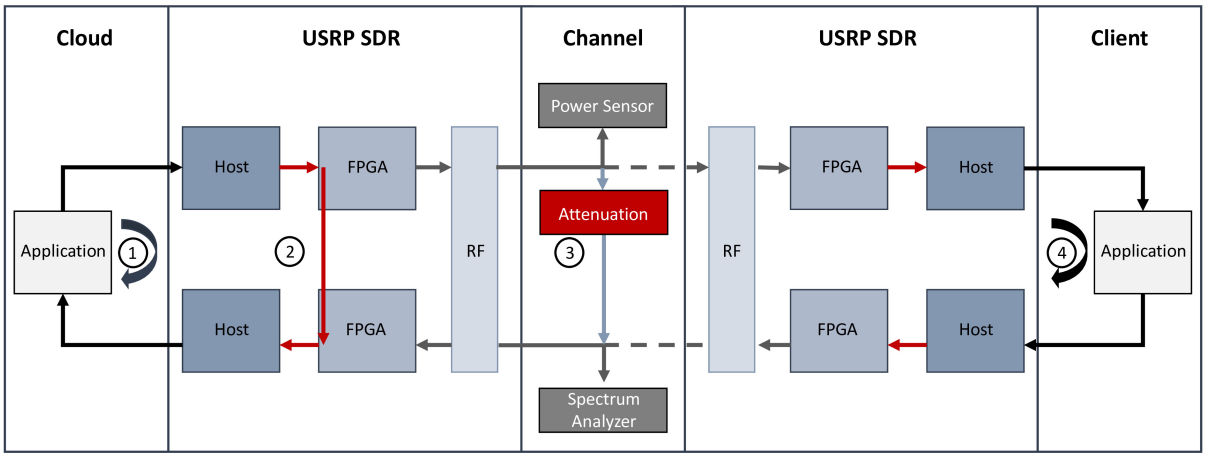


Figure 6.1: Set-up for the measurements

[1] OpenVPN could serve as a Middleware to set-up a virtual private network (VPN) tunnel between source and sink that encapsulates IP packets into UDP packets.

6.1.1 Infrastructure

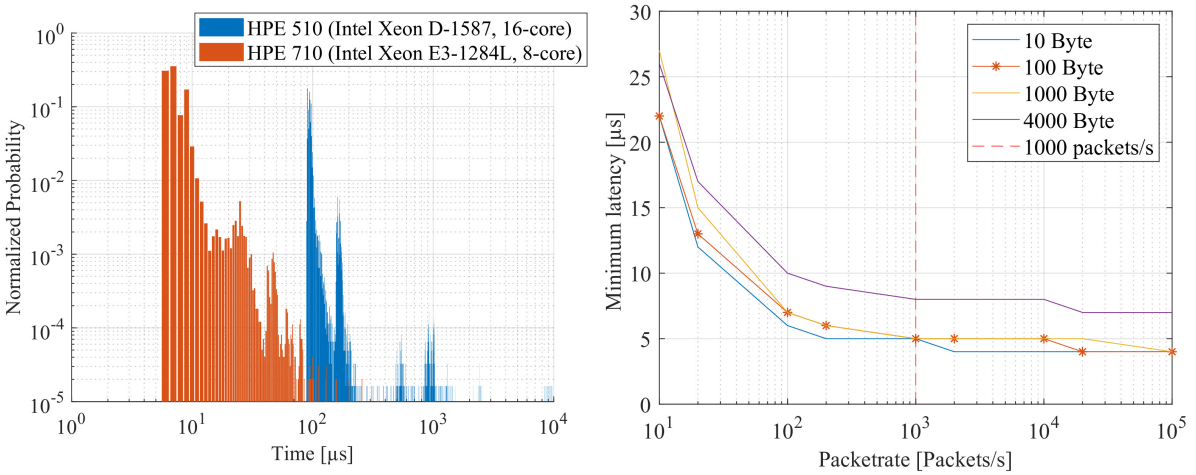
First, the latency of the application creating the payload is evaluated, which refers to the first path in Fig. 6.1. The two sub-plots in Fig. 6.2 show, that the latency measurements are influenced by different factors. Fig. 6.2(a) outlines that the experiment is influenced by jitter. Here the values are normalized by dividing each bar through the total number of all elements. Although most of the orange marked values are close to the peak of around 7 µs, there are several outliers up to 1 ms. The explanation is that the creation of the packets is dependent on the task scheduling of the operating system (OS). The thread creating the transmit packets is subject to the arbitrariness of the OS, although it is scheduled to create new packets in a given time fashion. So, the time between the timestamp is read out and the transmit packet is created is varying, because the timestamp is provided by a OS function.[2] Thus, the minimum latency is taken here as the latency of the communication network under test, without influence of the OS jitter. Still, it makes sense to take a look at the average values and distributions, because the final application, and especially the cloud version, utilizing those networks will exhibit these influences. Further, Fig. 6.2(a) indicates that the CPU platform has a impact too, where the blue marked server blade has a average latency of 107 µs.

The second sub-plot Fig. 6.2(b) shows that the latency is influenced by the packet rate and slightly by the length of the packet as well. A reason is that the scheduler of the OS prioritizes the higher loads and therefore higher packet rates, such as one packet per millisecond, differently than just a few packets created every second. To measure the latency, the packet rate has to be sufficiently high enough, without jamming the network. In the following experiments 1000 packets/s are taken and marked with a red dashed line. Higher packet rates tend to flood the network and the devices under tests.

The plot in Fig. 6.3 shows the second path, where the packets are routed from the packet source to an LabVIEW application, which guides them through the FPGA and sends them back to the source without any signal processing. Here, the minimum latency increases from around 5 µs to roughly 70 µs, when LabVIEW and the source application are running on the same computer platform. In a separate experiment, the round trip time (RTT) between the LabVIEW host application and the FPGA was measured, where the data exchange is using DMA FIFOs over the PCIe connection to the USRP. It always takes around 30 us to write samples of a vector of any length to the FPGA and read it back. In addition, the time increases with the length of the vector and it depends on the clock speed of the FPGA to pass each sample between the data input and output first in, first out (FIFO) buffer.

The intention is, however, to use the FPGA prototype stand-alone like a networked device which is attached to the cloud or client application. Therefore, Fig. 6.4 compares

[2] The timestamp utilizes the high precision timer which is bound to a counter based on CPU ticks with a typical resolution in nanoseconds. However, the OS can also decide to provide the system timer as input with a resolution in milliseconds.



(a) Distribution for 1000 packets/s and 71 Bytes payload length for two recent server platforms

(b) The minimum latency versus packet rate for different packet lengths

Figure 6.2: The test application is looping the packets back to itself on the same computer. This corresponds to the first path in Fig. 6.1.

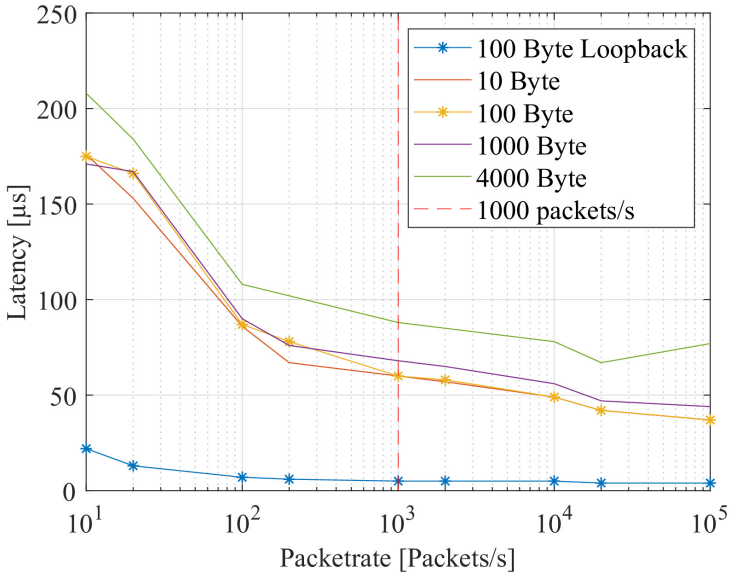


Figure 6.3: The minimum latency of the application and the LabVIEW host and FPGA application, without any data processing. This corresponds to the red line in Fig. 6.1.

the influence of the packet size on the latency, when the data is routed over a wired network. For comparison the same influence is shown in case the UDP data stays on the same computing platform, referred as "localhost". The minimum RTT latency for a wired network is in this case around 80 µs and increases with bigger packet lengths.

As a conclusion, the (minimum) latency until the payload arrives at the signal processing modules in the FPGA is already in the same order of magnitude than the signal processing latency values reported in Fig. 4.18. Further, this latency is subject to severe jitter. To

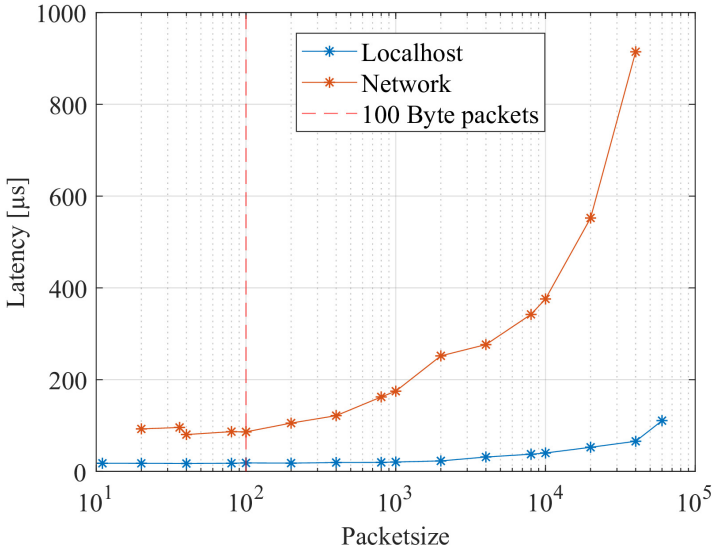


Figure 6.4: The latency introduced by adding a gigabit network between the cloud and client application.

reduce it, the UDP reception could be moved into the FPGA, because the USRP platform has built-in Ethernet receiver options. However, this only removes parts (30 µs) of the total latency associated with DMA FIFO data exchange. UDP is already the fastest network protocol, because it has a minimal overhead.[3]

6.1.2 Single Link Experiments

The capabilities of a single link transmission are studied in this section. In Fig. 6.1 this is indicated as the third path and a light blue line. However, the radio frequency (RF) transceiver in the USRPs are subject to cross-talk inside the SDR device. So, this experiment regarding AWGN conditions requires to use one USRP as transmitter and a separate one as receiver attached to the same host computer. The set-up wires the transmitter via a programmable attenuator with the receiver. This attenuator has a minimum attenuation of 75 dB and a maximum of 105 dB and can be switched in 10 dB wide steps. A further programmable attenuator is built inside the transmitter RF frontend, which can switch in 0.5 dB steps in a range of 31 dB. This way a combined attenuation range of 75 dB to 136 dB can be used for the experiment.

Further, at the transmitter side the signal is split into 4 portions. The first portion is intended to measure the average transmit power using a *Rohde & Schwarz* NRP-Z51 Thermal Power Sensor. The second portion is connected back to the same transmit USRP to act as reference. The third portion is attached to a spectrum analyzer and last portion is connected to the programmable attenuator. After the attenuator another splitter is used

[3] If it is necessary to provide the payload packets more reliable, then transmission control protocol (TCP) would have to be used which results in a higher latency.

to connect the receiving USRP and an additional spectrum analyzer. The loss of the two splitters is respected in the minimum attenuation. The reason for this set-up is, that the *Rohde & Schwarz* power sensor is not sensitive enough to measure the average received signal strength at the USRP. Therefore, the transmitted average power is measured and the adjusted attenuation is subtracted to get the receive power level. The power sensor can only measure the average signal strength, so all waveform configurations have to send frames continuously.

Using this set-up, the Long Term Evolution (LTE) Application Framework (AFW) and the implemented transceiver in different waveform configurations are compared with each other, using a fixed modulation coding scheme (MCS) setting with code rate ½ and quadrature phase shift keying (QPSK). Although both implementations are conceptually different from each other, they are both transparent for a cloud application connected with a client, except for the achievable link performance, such as PER and latency. Further, both implementations are running on the same hardware platform and are experiencing the same capabilities of the SDR platform.

National Instruments (NI) offers also the 802.11a AFW, which would match the generalized frequency division multiplexing (GFDM) transceiver in the given "WiFi" configuration. However, this framework did not operate reliably enough in different experiments, especially over the air transmissions, such that only the latency results are kept here.

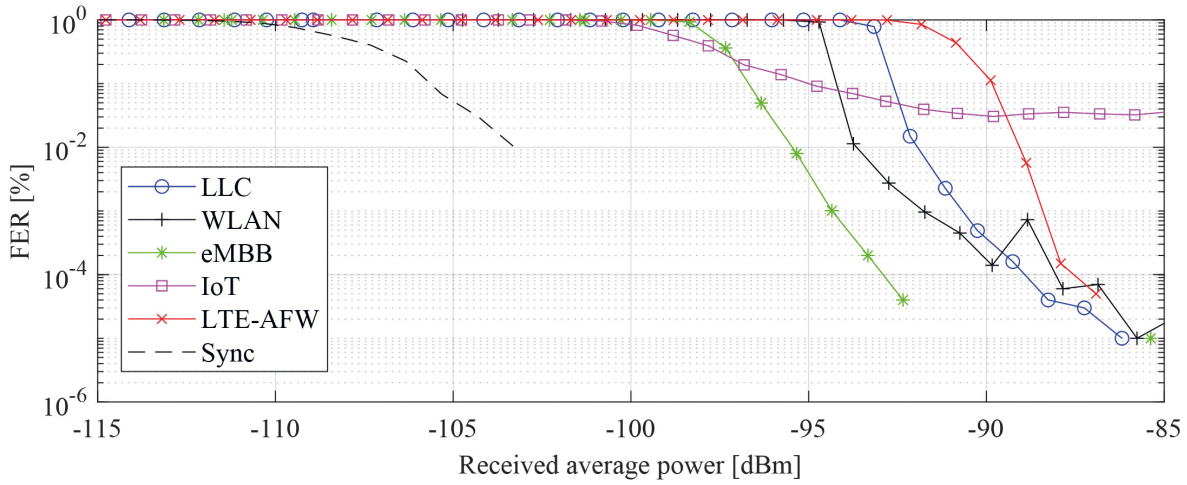


Figure 6.5: Frame error rate of the transceiver using different waveform configurations and the LTE Application Framework in AWGN conditions using a programmable attenuator. This corresponds to the third path indicated as a light blue line in Fig. 6.1. The FER performance is measured using UDP packets. The curve marked as "sync" indicates the ratio of transmitted to detected frames.

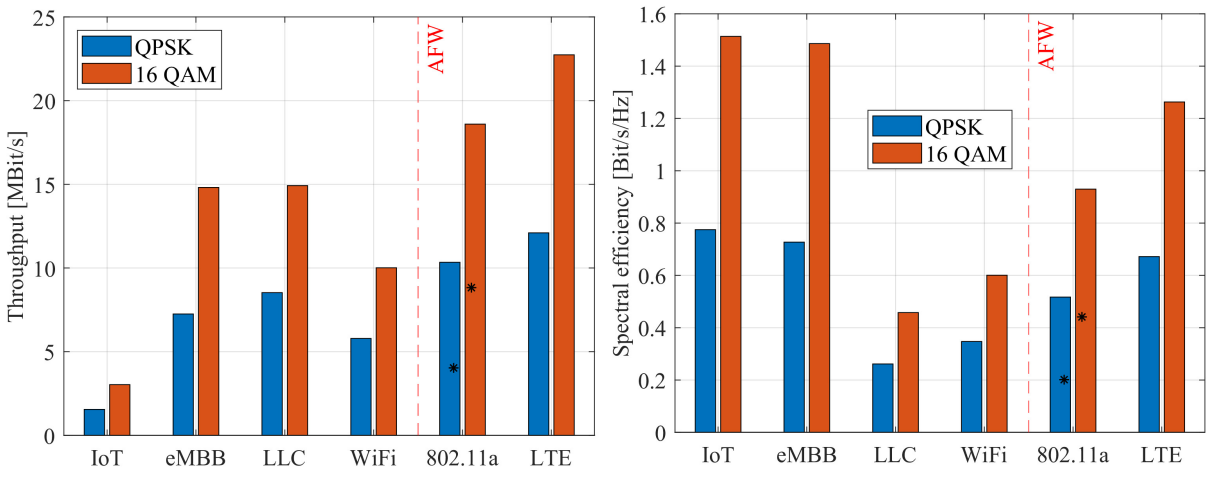
The PER for the presented transceiver and the LTE AFW are shown in Fig. 6.5 with respect to the average receive power. The LTE AFW has the worst PER because the

transmit signal does not utilizes the full dynamic range of the DAC. Hence, the transmit power is less than the different waveforms created by the presented transceiver, as shown in Fig. 5.6b. The transceiver in the "LLC" configuration is using 40 MHz of bandwidth, thus, more noise is captured in comparison to the other parameter settings. The "WiFi" curve uses 20 MHz of bandwidth while other waveform parameters are similar to "LLC". As a result, the PER performance is around 2 dB better. The bandwidth is further halved for the "eMBB" case. Furthermore, the "eMBB" parameter settings are adjusted to resemble LTE. So, the communication system is avoiding RF impairments at the edges of the usable bandwidth and the cyclic prefix (CP) length is increased. This results in a 3 dB difference. Only the "IoT" configuration, that uses a single-carrier (SC) waveform, can still transfer data for lower power levels, because it can utilize the dynamic range of the ADC/DAC better due to the relaxed peak-to-average power ratio (PAPR) compared to the multicarrier (MC) waveforms. However, each IQ sample uses the complete bandwidth of the SDR including all impairments. The implemented synchronization still detects the radio frames up to −105 dBm average receive power, such that a better PER performance might be feasible in case advanced receiver designs are used.

The achievable throughput is evaluated in Fig. 6.6, where the transmit and receive USRP are connected via a fixed 30 dB attenuator to ensure stable AWGN conditions. The transceiver in the "IoT" configuration has the lowest overall throughput because of the lower sampling rate. The "eMBB" configuration is adapted to a LTE 10 MHz system configuration and fits the SISO LTE data rates given in [189]. The transceiver in "LLC" configuration achieves the highest data rates of all configurations because it uses the highest (40 MHz) sample rate. The transceiver in "WiFi" configurations has similar results to the commercial available 802.11a AFW in case the packet sizes are aligned.

Fig. 6.7 presents the latency results. Here, the 802.11a AFW and the transceiver achieve similar results, because both have simple medium access control (MAC) mechanisms compared to the LTE AFW. The 802.11a AFW is slightly faster than the transceiver in the "WiFi" configuration because it operates using a fixed orthogonal frequency division multiplexing (OFDM) waveform setting, whereas the transceiver applies more signal processing steps to create the Block OFDM waveform. However, this only holds if the listen before talking (LBT) mechanism defines the channel as free and does not delay the transmission.

In contrast, the LTE AFW follows the LTE standard and has several caching mechanisms due to the protocol. Already on the host side, the packets are buffered first before adding a respective MAC header and they are written to the FPGA. Further, LTE operates in a very strict time grid, such that the data is buffered until the start of a new subframe every 1 ms. Then, the channel encoder consumes already 100 000 cycles or 0.5 ms, prior passing the data to the remaining processing modules. A similar behavior happens at the receiver. So, the data packets have a much higher initial latency of 4.7 ms and experience a much higher jitter, as shown in the next section in Fig. 6.8.



(a) Achieved UDP throughput.

(b) Achieved spectral efficiency

Figure 6.6: Throughput of the implemented transceiver in different parameter settings in comparison with the 802.11a and LTE AFW from the application perspective using UDP. The stars marked for the 802.11a AFW are indicating the throughput achieved, if the 802.11a AFW packet size is the same as that for the "WiFi" configuration. The maximum rates for the 802.11a AFW are achieved using 1024 Byte packets. The results are taken by following the light blue line in Fig. 6.1 with the attenuation fixed to 30 dB.

6.1.3 End-to-End

Finally, the End-to-End performance including all elements is studied, such as depicted in Fig. 6.1 marked with a black arrow. That means the packets are sent to the client application and looped back to the cloud application using the complete network, so they will pass twice the different links. Here, the USRP based software solutions are also compared with different commercially available WLAN and LTE solutions.

The latency histogram of both LTE AFW and GFDM transceiver are given in Fig. 6.8. It shows that the round trip latencies of the packets using the GFDM based link are confined around 1 ms in comparison with the LTE AFW that has a higher average latency and jitter. The same Fig. 6.8 links the results to other networking solution, too. Using just two 10 Gigabit switches between cloud and client, will have the lowest jitter and an average value of 150 µs. The two wireless local area network (WLAN) USB sticks in ad hoc mode are providing similar results compared to the transceiver. However, the latency values seem to be quantized compared to the other diagrams, due to buffering effects and time triggers inside the chip-sets or the OS driver, that only allow to pass data in given time intervals. The results presented as LTE networks are gained by using two LTE routers in-between the cloud and client application.

The capabilities of the USRP based transceiver and LTE AFW solutions are also compared using the testbed in an outdoor experiment. Here, the antenna is located 9 m over the ground, facing a straight sidewalk where the user-terminal (UT) is located with different

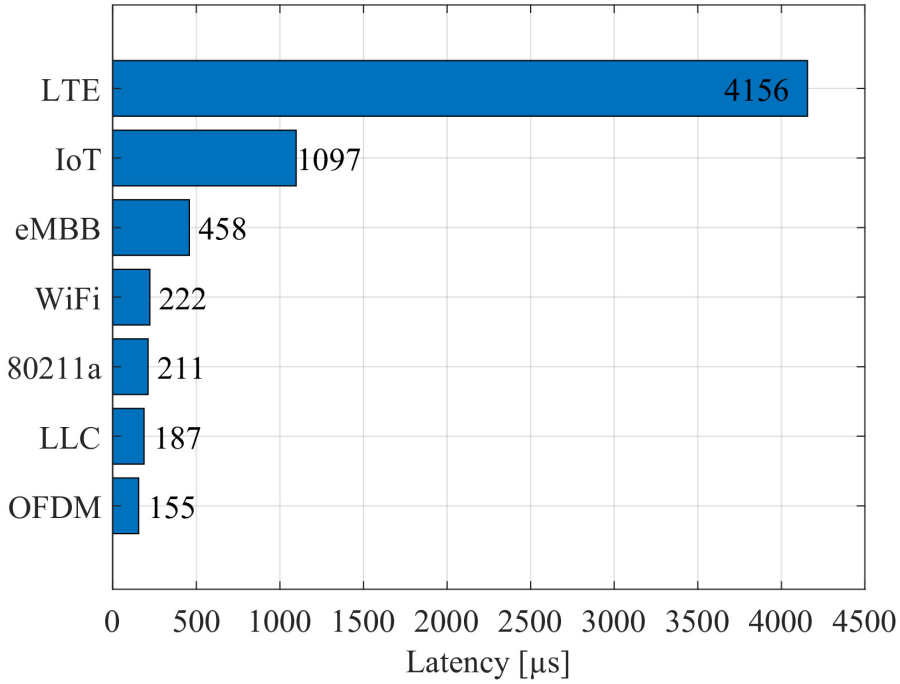


Figure 6.7: The minimal measured latency of the transceiver in comparison with the *NI* WLAN and LTE Application Framework in AWGN conditions. This corresponds to the light blue line in Fig. 6.1. The parameter setting "OFDM" corresponds to the "WiFi" configuration, however using 80 MHz of sample rate.

distances to the base station (BS). All positions fulfill line-of-sight conditions towards the BS. The maximum length is 140 m. For several measurement positions the experiment is conducted for 30 s for both GFDM transceiver and the LTE-AFW.

The set-up is composed of one USRP 2974 used as a BS and is located on the second floor of the building. It uses a sectorized antenna of the type *Ubiquiti* AM-3G18-120 and the 1 Watt power amplifier ZVE-8G+ from *Mini-Circuits* for the transmit path. The transmission license for this experiment covers 10 W equivalent isotropically radiated power (EIRP). Therefore, the maximum transmit power is chosen to reach it.

To provide sufficient isolation in the experiment, the receiver port is connected to the horizontal polarization of the antenna, whereas the transmitter port is connected to the vertical one. This ensures 28 dB isolation between both paths. Moreover, a custom tuned bandpass filter is used to protect the receive frontend.

Although, the implemented transceiver supports time-division duplexing (TDD) operation, the LTE AFW in the used software version 2 does not. So, the downlink (DL) frequency for the experiment is set to 3.5 GHz and the uplink (UL) frequency to 3.75 GHz for a frequency division duplexing (FDD) set-up. Further, the transmission power levels have to be calibrated using *Rohde & Schwarz* NRP-Z51 Thermal Power Sensor, because

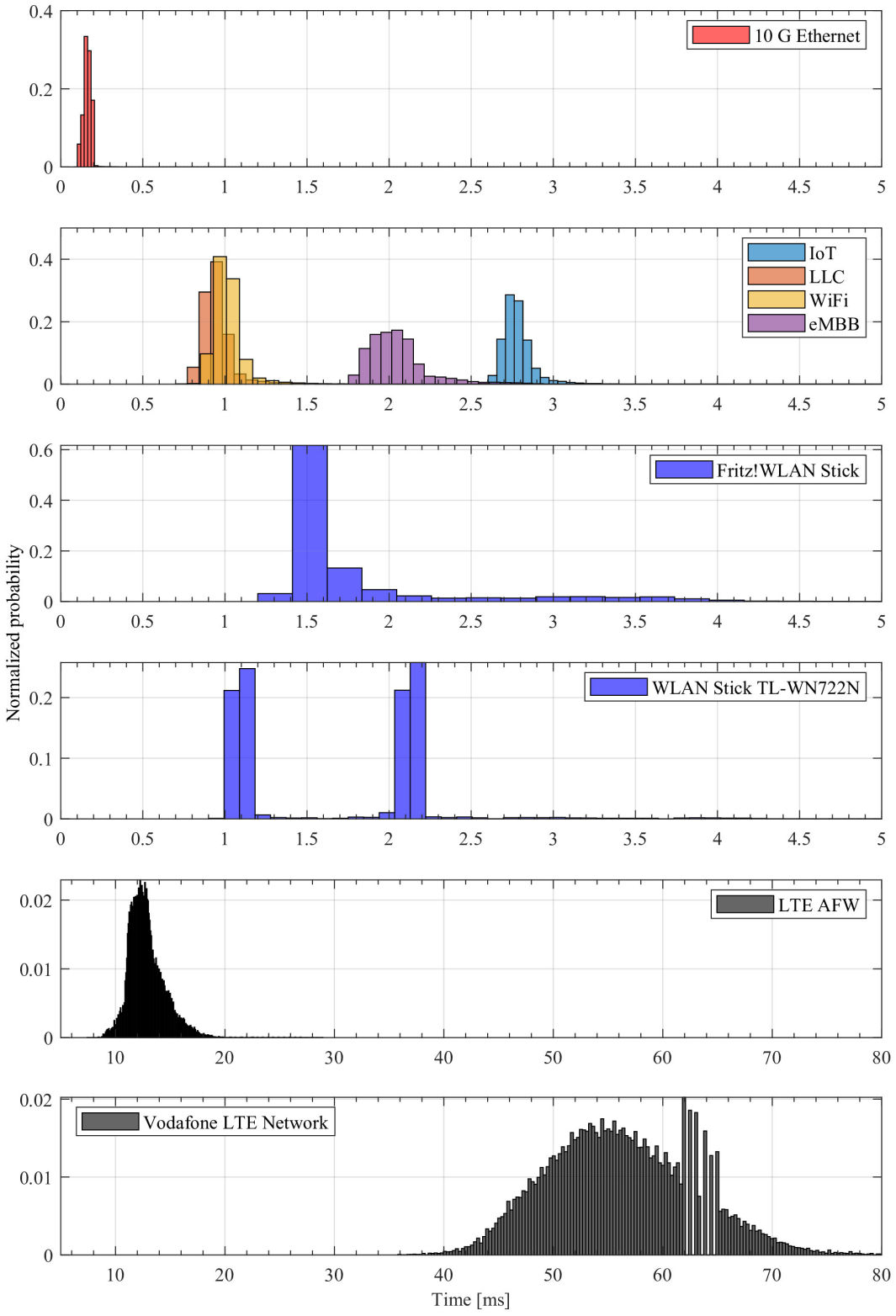


Figure 6.8: Distribution of the RTTs using various network connections between cloud and client application.

the LTE AFW has a fewer output power level then the transceiver, according to Fig. 5.6b.[4] Fig. 6.9 shows that the PER are similar for both implementations. The transceiver has an advantage and successfully transmits all packets in 6 out of 9 locations.

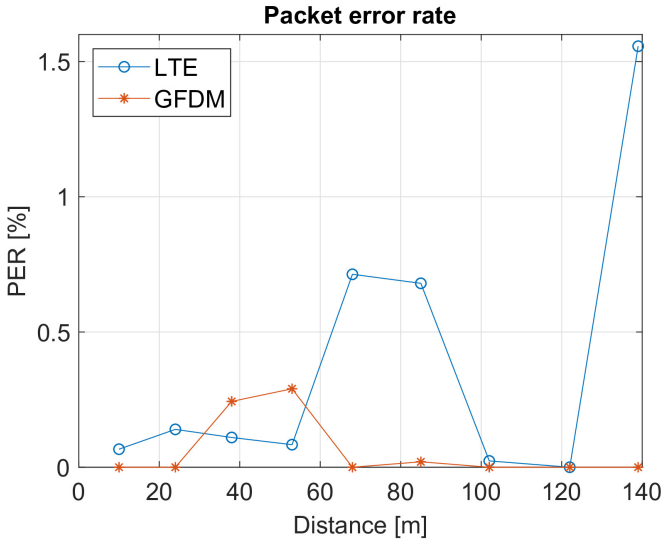


Figure 6.9: PER of the transceiver in "LLC" configuration and the LTE Application Framework in a outdoor line-of-sight (LOS) scenario, based on [184].

6.2 Multi-User

The transceiver can also be used beyond single user experiments using the standalone USRP 2974. Fig. 6.10 shows a set-up comprising several robotic applications connected and controlled by an access point (AP) with an attached edge cloud. The USRPs with the identification (ID) numbers 1 and 2 represent a sensor/actuator system. The camera-sensor captures the status of the air-hockey table and process the data using a attached computer [190]. The data is then transferred via the AP to a cloud processor which calculates the strategy and the resulting movements of the actor to defend the goal. This motor control data is transmitted to the actor via the attached USRP. A second set of two USRPs is linked to two robot arms. The robot arm with the USRP ID number 3 is the master and controlled manually. The robot arm marked with ID 4 acts as a slave and copies every movement of the master robot. Further, it sends force feedback information back to the master, such that the operator can feel in case an obstacle has been hit. The last element in the demo is an automated guided vehicle (AGV) that is controlled by a cloud application.

Although, the MAC layer is simplified, it still allows to define one transceiver as AP that serves several clients such as described in Subsection 4.1.1. The waveform configuration

[4] Despite the parameter setting in the graphical user interface (GUI).

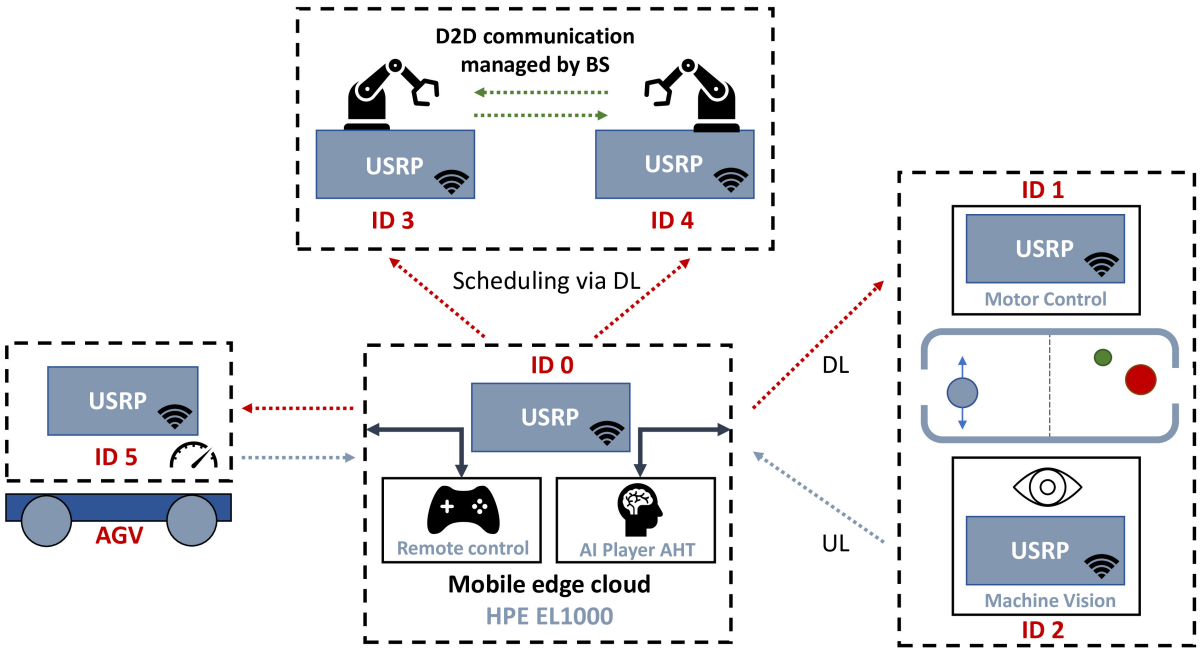


Figure 6.10: Demonstration set-up with one access point and several clients

for the overall demonstrator is set to ”LLC”. Tab. 6.1 summarizes the latency and PER results for the presented set-up. The minimum latency values are in line with the previous results for a single link in Fig. 6.7, just doubled because now the packets are on a round-trip between cloud and client application.

The latency results can be summarized in Fig. 6.11 and show the influences of the components into the overall RTT. The interconnection of all components using Ethernet switches has the biggest contribution, followed by the LabVIEW run-time environment before the payload is finally passed to the FPGA. This influence could be reduced by utilizing the Ethernet transceivers in the USRP platform. As indicated in the fourth chapter, the signal processing in the FPGA is predictable and deterministic, but has a minor impact as well. So, that it is not necessary to perfectly optimize the FPGA implementation for higher clock-speeds and lesser delays, except the aim is to reach WLAN standard compliance.

Station	1	2	3	4	5	Mean
Minimum latency	475 µs	487 µs	475 µs	484 µs	499 µs	484 µs
Average latency	1051 µs	973 µs	980 µs	974 µs	938 µs	983 µs
Jitter	486 µs	307 µs	319 µs	302 µs	231 µs	329 µs
Maximum latency	8233 µs	8558 µs	8287 µs	6269 µs	5863 µs	7442 µs
PER	1,697e-05	6,796e-05	6,793e-05	3,455e-05	4,381e-05	4.6244e-05

Table 6.1: Achieved performance of the transceiver in the ”LLC” configuration with 5 users. The performance is measured between the cloud computer and the individual clients.

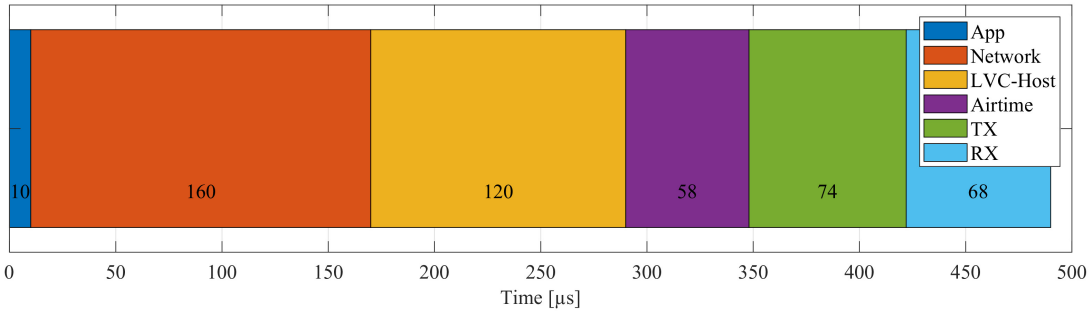


Figure 6.11: Estimated contribution of the components to the overall round trip time in µs, based on twice the values in Fig. 6.4, Fig. 6.3 and Fig. 4.18 for the transceiver in the ”LLC”’ configuration.

6.3 26 GHz mmWave experimentation

The presented architecture can also be used to study algorithms aiming at MAC level functions and beyond the frequency range of the USRP. From an experimental viewpoint, hardware platforms for mmWave are usually sophisticated and expensive, because they are tuned to provide high data rates of 1 GBits and higher [26], which certainly increases the overall complexity of the transceiver systems rather than providing a basic platform for the practical evaluation of beam steering algorithms. Here, it is highlighted that for the sake of investigation of intelligent beam steering algorithms, the data rates are not decisive. This also applies to further mmWave experiments for instance in multi-connectivity demonstrators, where the focus is to distribute messages over several radio access technologies.

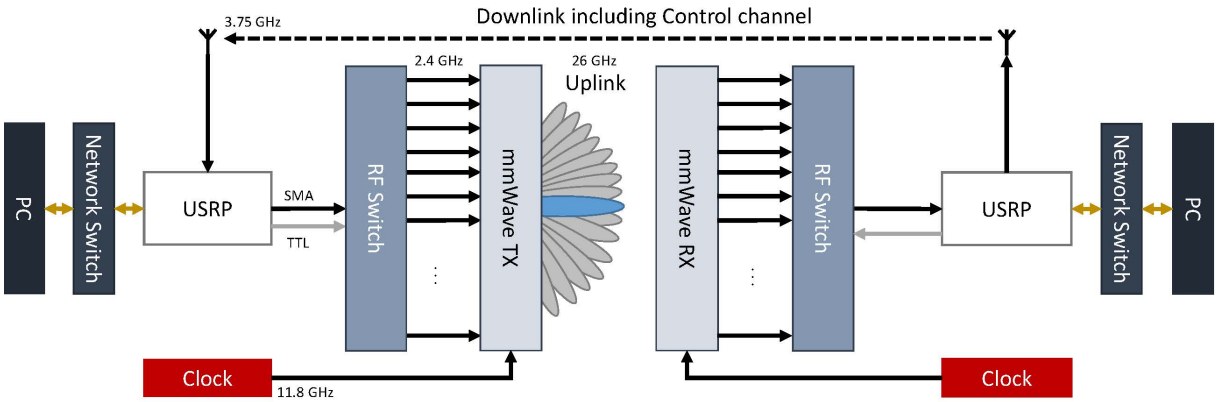


Figure 6.12: Overall hardware set-up used for the mmWave experiment, based on [150].

The mmWave frontend, shown in Fig. 5.9a, has 16 inputs which are corresponding to 16 independent beams into different directions. Since only a single beam in a single direction is going to be radiated, a 1 to 16 RF switch (*Mini Circuits* USB-1SP16T-83H) connects the USRP TX output to the input of the mmWave frontend. The FPGA then selects the respective port and therefore the beam via a separate TTL signal. The same applies to the

receiver as well, where the receiving port has to be selected. The issue for the operation of a stable network is to find the best beam combination initially and to continuously track it to keep the link alive. The background is that mmWave frequencies are impacted by a strong path-loss attenuation and to mitigate this, narrow beams are used to focus the signal energy into the direction of the receiver. This requires constant tracking of the receiver to keep a favorable LOS condition, despite using narrow beams.

This experiment explores the capabilities of the beam-steering algorithm similar to [26] under two different walking-speeds. The experiment is using an X310 based set-up with a control PC running Windows against the USRP 2974 running the *NI* Real-Time Linux as shown in Fig. 6.12. The PHY parameters for the GFDM transceiver implementation are set to the "LLC" configuration. The data-rate is measured with UDP packets between the Transmitter and Receiver. Fig. 6.13 shows the achieved data rates with the set-up. The user equipment (UE) moves parallel to the receiver, i.e. the angle changes, with moving speeds around 1.1 km/h and 2.2 km/h starting from a 0° angle until the receiver cannot detect any data.

The experiment shows, that compared to the static case at 0° the data rate drops with the increasing speed. Although most of the PHY signal processing is moved to the FPGA, the operating system has an influence on the performance of the system again, because the beam steering algorithm transfers a few control signals via the host computer. Here, the control PC running Windows drops many UDP packets compared to the Linux Real-Time system. As already indicated in the beginning of this chapter, the CPU platform as well as the OS can severely perturb the experiment.

Using the *NI* Linux Real-Time system, the set-up can successfully apply beam tracking and achieves a throughput of 2.9 Mbits/s with light mobility. This data rate is sufficient to enable wireless control loop experiments utilizing mmWave frequencies.

6.4 Summary

This chapter shows the performance of the proposed transceiver architecture in different experiments. The results enclose:

1. The implementation can achieve the 1 ms latency constraint defined for 5G low-latency networks.

2. The transceiver exhibits similar PERs in comparison to the commercially developed software framework, such as the *NI* LTE AFW using the same SDR platform.

3. The architecture offers a higher flexibility than conventional implementations to support various application scenarios.

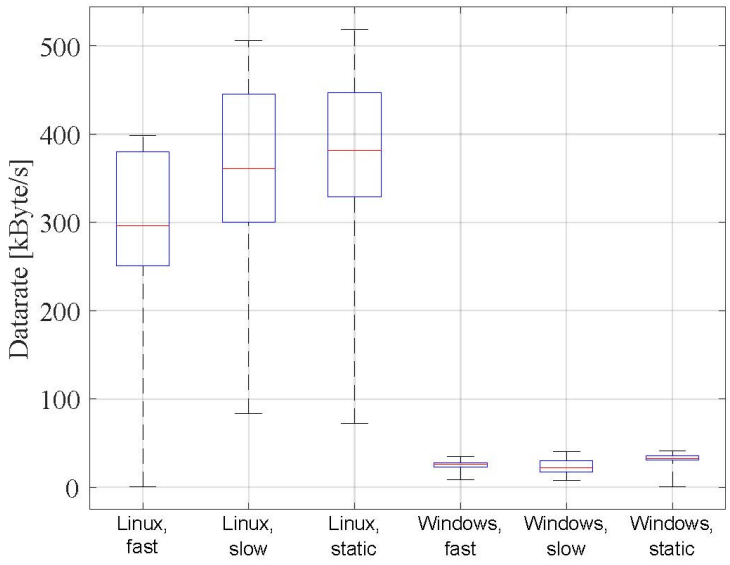


Figure 6.13: Achieved data-rate with enabled beam-tracking to follow a moving client. The evaluation is conducted for every second during the experiment. Based on [150].

4. Finally, the demonstration of a tactile internet prototype utilizing robotic arms in a multi-user environment could successfully be achieved.

Although the transceiver is implemented on a FPGA with deterministic execution characteristics, the end-to-end latencies are affected by jitter due to other network elements and the OS. The explanation for the jitter due to the OS is that it has to manage and prioritize very different processes which leads to unpredictable delays. The usage of real-time kernels could mitigate the influences of jitter, in case the application can fit into these OS concepts. However, programming such a wireless control loop especially on a cloud computing platform using real-time OS is not trivial. Especially, since modern concepts tend to move functionalities into cloud mechanisms such as virtual machines (VMs), which would add additional sources of jitter as shown in Fig. 7.4. Those concepts tend to use large frameworks to abstract complexity for the targeted application and introduce further ambiguities. One example are ”sampled” latency histograms of the WLAN adapters, where the (OS-) driver, the chip-set, the USB-Bus or all elements together could be the source. Compared to LTE, the implemented transceiver is more deterministic.

Further, the Ethernet network connecting all elements adds more delays for the application and the delays are in the range of around 100 µs per switch. The experiments here and in [124] show that changing from one gigabit to ten gigabit Ethernet changes the latency negligible.[5] Potential jitter could be handled by utilizing time-sensitive network techniques, but the constant overhead would still remain. Moreover to keep the latency, the current approach requires to create constant packet flows with dummy data to ensure

[5] Although the study is not comprehensive and does not include different switches from different companies.

that the OS is keeping the high priority, even if the application is just a wireless emergency stop, which does not need to transfer huge amounts of data, but has to react in a short and deterministic time.

The ideal low latency edge cloud platform would therefore consist of one powerful CPU platform, where the wireless network is attached via a PCIe-Bus, instead of using networking components to connect the wireless AP with the computing elements. In addition, all software components would have to be implemented respecting the prerequisites of real-time OS architectures, avoiding virtualization frameworks. From a hardware perspective, the *HPE* Edgeline 1000/4000 series allows to extend an edge cloud platform with FPGA based hardware accelerators or USRP SDR via the PXI interface [191].

NI is using a similar way for the ns-3 simulator. The idea is to combine the extensive higher network layer simulator ns-3 with real-world experimentation such as the USRP SDR platform and the LTE AFW. This allows to emulate the core network and MAC protocols without re-implementing everything in the LabVIEW environment. The final realization uses linux pipes [192] as the data exchange protocol instead of the UDP and is running on the same computing platform, because of the above mentioned reasons.

Finally, experimentation proves as an essential tool to study the behaviors of networking elements. It is difficult for simulators to cover all aspects, ranging from RF impairments to jitter caused by the OS system. Thus, this chapter provides a tool to measure the overall and consecutive PER and the latency including the histogram to capture the performance of different network elements from a application perspective. The results could be kept in databases, such as *IEEE* DataPort, to provide simulations with performance models of different networking components.

Chapter 7

Key lessons

During the development of the transceiver, the testbed and the demonstrators presented in this work the following key lessons have been identified.

- ▷ The complexity of the overall system is very large.
- ▷ There are many (external) dependencies out of reach.
- ▷ There is a uncountable amount of possible realizations, but it is difficult and time consuming to judge the best one.

A general observation is that the demonstrators and prototypes got more complex over the past years, beyond the boundaries of the communication research area. Many years a clean in-phase and quadrature (IQ) diagram was sufficient to exhibit successful transmissions over a wireless link. Since the proposal of the Tactile Internet, however, showcasing a wireless network prototype needs to include for instance robotics, drones or fancy graphical user interfaces. Mainly to attract interest and to advertise the concept, but more importantly to satisfy the demands of the funding sources.

Tactile Internet demos and research prototypes require expertise from many disciplines, for instance with the real-time demo presented in Section 6.2. Employing the robot arms required the help of robotics engineers from Technische Universität München to program the arm with its 7 joints properly. The operation of such an arm wirelessly requires stable network conditions, which has a packet delivery error of less than twenty consecutive packets, otherwise the arm will make an emergency break. This means that the normally defined packet error rate (PER) is not a correct metric to judge a wireless system. Further, the commercially available consumer wireless access points *AVM* Fritz!Box 4040 for instance crashes in the moment the robot arms start to create their packet flows, whereas the presented transceiver implementation operates the arm stably. Here, not the amount of data throughput was the issue, but the amount of packets created.

Also the amount of components in the system set-up increases with demonstrator complexity. The final demonstrator in Fig. 6.10 consist of 6 USRPs with one control

notebook each. One server as the edge cloud, 4 computers to control the air-hockey table and the two robot arms. 3 Robot control platforms and another server to operate the automated guided vehicle (AGV). Further, to visualize the results a wireless local area network (WLAN) and several computers were set-up. Each device needs a IP address, each component is programmed in a different language (HTML, Javascript, Python, C++, LabVIEW, VHDL and C#) running on various operating system (OS) (Windows, Linux, Real-Time Linux) and is developed by other team members across three institutions. Setting such a demonstrator up, in the the development environment but also in a showcasing location, requires discipline but also an intelligent back end. Here techniques such as Message Queuing Telemetry Transport (MQTT) or as presented in the previous chapter TestMan are very relevant to orchestrate the components separated from the actual data flows. Besides the time consuming trend to provide demonstrations including aspects outside of the research areas, there are other limitations too.

7.1 Limitations Experienced During Development

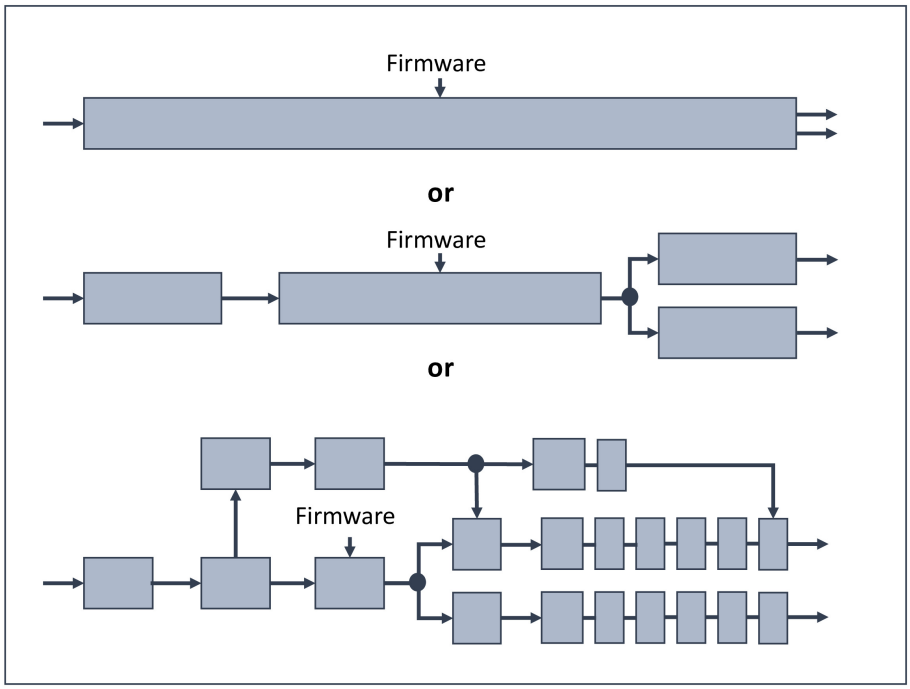


Figure 7.1: Adapted from [98]: Should a complex demonstrator be bought as a whole or in smaller pieces?

As indicated in [98], selecting the right hardware set-up is an art in itself. The issue is that modern hardware and especially Software-defined radio (SDR) contain a huge amount of software solutions, which can be released in an immature state with the intention that updates allow to fix bugs and add missing features later [98]. For example, a USRP-RIO has several ways to connect the field programmable gate array (FPGA) to the host

computer: via two 10 GBit Ethernet or one PCI-Express connector. The LabVIEW version however is supporting only the PCI-Express connector. Hence, Ethernet packets can only be routed via the LabVIEW driver and PCI-Express to the implemented transceiver. Especially, low latency applications would benefit from a direct access via the network interfaces, instead of complex routing via the host computer which introduces jitter from the operation system. The developer needs to plan ahead and plan for the possibility that drivers and firmware might be adapted either far into the future or not at all[1] or change the development method and rewrite the transceiver chain. Even worse, manufacturers can dismiss features. During the implementation of the presented transceiver using LabVIEW Communications System Design Suite each of those cases happened, which lengthened the development severely.

In addition, universities do not have many senior code-developers such that the implementation works has to be carried out by students or junior researchers, whereas the senior researcher focus on academic research and teachings. Thus, strict coding guidelines and constantly enforcing them could be missing in those environments. This leads to unstructured and undocumented code, which is rather re-implemented then re-used, because the following generations do not understand the previous developments. A first FPGA implementation [153], could not be reused because the version of the IDE was changed to support the modern hardware and the code could not be converted automatically. Further, the code was developed by a single developer who had left. Tools such as Doxygen [193] could be used to develop the documentation alongside with the source code, for text based programming languages.

As in any wireless system, power control is needed to adjust the input power level. The driver environment for the USRP implements this as a non-disclosed function, where the desired values have to be written over a slow connection via the host computer. This is acceptable for systems with continuous data streams such as LTE, but it is difficult for WLAN systems with burst transmission, where the power control has to act fast. Hence, the implementation required time-consuming reverse engineering, because the driver mechanisms are not documented. Open source drivers, such as the UHD for the USRP series, are not automatically a better solution just because the source code is open, as long as a (hand) written manual describing the firmware is missing as visible in [194].

In our case, the testing strategy changed during the implementation phase due to removed features in the software development environment. The initial idea was to use the feature of LabVIEW to interpret MATLAB scripts because the transceiver has so many parameter settings where a script could generate the reference signals more efficiently than creating test-vectors for every possible case. Further, also the transceiver configuration code was developed using MATLAB scripts because the interpreter runs under Windows as well as Linux. However, the interpreter was removed in the recent versions, such that the very important reference and configuration code had to be reimplemented. Keeping the

[1] In this specific case it took four years for this driver update.

older LabVIEW version was not an option either, because missing features, for instance the Ethernet driver, were added in the newer version and the stability of the software increased significantly.

Cooperation with other institutes could help by sharing the implementation efforts and split the overall work into smaller parts to divide them via a larger team. Here complications may occur when different aims are pursued, for instance developing a salable product rather than a open source prototyping platform.

7.2 Prototyping Future

The biggest challenge regarding the development of real-time prototypes is the time spent on the implementation and optimization of code for a specialized hardware platform such as a FPGA or digital signal processor (DSP). Further, in most cases this implementation is not flexible enough and needs to be re-adapted to support more features, which prolongs the development cycle. In addition, if very modern or even prototype development workflows are used, then they are tied to specialized hardware / software platforms that might be discontinued in the future. Here, central processing unit (CPU) based implementations provide a easier development and test environment.

So far, architectures purely based on CPU could not keep up with the real-time demands of a radio. OpenAirInterface is known to support Long Term Evolution (LTE) in a 5 MHz mode using 4 CPU cores [117], because LTE has relaxed requirements regarding the acknowledgments (ACKs). However, modern CPU with many core-architectures, such as *AMD's* Ryzen 64-core and 128 threads CPU, are able to replace a graphics processing unit (GPU) for computer games [195]. Further, some mainboards allow to place two of those CPUs in one system.[2] Thus, they could become a viable option to perform signal processing tasks on a standard x86 platform. Further, USRP or more interestingly smaller form-factor SDR cards such as [196] can be added via PCI-Express, such that similar priced system, as the USRP 2974R, can be build, without the need for specialized FPGA modules. The idea of simply using many x86 cores instead of specialized processors architectures is also pursued by *Intel* with the Larrabee project [197]. Here the aim is to build a x86 based many core graphic card.

The authors in [198] study how server and network structures such as presented earlier can process radio frequency (RF) signals with a bandwidth of 3 GHz and 10 Gbps using standard CPUs. This could be a way towards processing data rates for mmwave systems, without having complex platforms [104] which are used in [26].

However, the USRP X310 system is still not obsolete, since the PCI-Express connection offers a lot of combination possibilities. As indicated in the beginning it can be used as

[2] Linux fully supports this architecture and can use the multi-threading capabilities, and so outperforms *Intel* alternatives [195].

a massive channel emulator, massive MIMO demonstrator and as presented in the course of this work as an industrial access point with mmwave capabilities. So the extension towards a massive MIMO demonstrator using mmwave is not too far away. For instance, it is planned to extend the single-user setup with a multiple antenna system by attaching multiple USRPs instead of a single one and the 1 to 16 RF switch as shown in Fig. 7.2. Because in theory, by using a passive Butler matrix, each input and the respective beam pair is independent from each other, multi-user spatial multiplexing can be explored. Here, the *National Instruments (NI)* Massive MIMO software could serve as a starting point.

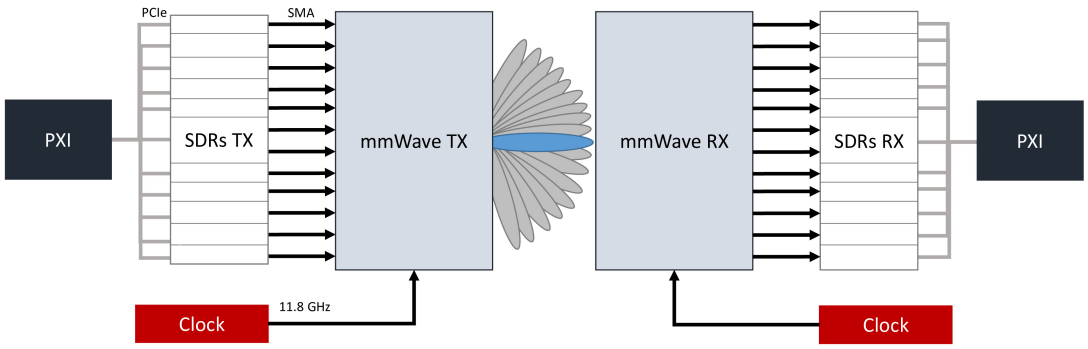


Figure 7.2: Hardware set-up using the *NI* Massive MIMO Framework. The clock signal for the mmWave Frontend can either be shared as in the figure or separated like in the experiment. Adapted from [150].

To process wide bandwidth data for a Terahertz demonstrator an interconnection of multiple USRPs for multi gigabit data rates is imaginable, because each USRP could process a given part of the spectrum. For instance, one subcarrier could be 160 MHz wide and each USRP would process it as a single carrier waveform such as in the LTE uplink. Such, that the whole demonstrator of multiple USRPs can be seen as a multi-carrier waveform receiver.

In addition, the cloud RAN concept of Fig. 7.3 with functional splits offers a good compromise between pure specialized implementations and general purpose software, where partial functions run on the FPGA and other on the CPU. For instance, the functional splits could be applied to individual blocks in Fig. 4.1 as well. Especially for the evaluation of advanced modulation or decoding techniques it makes sense to run the synchronization and channel estimation algorithm on the FPGA, rather than in a CPU. This reduces the data rates, because not all IQ data has to be transferred.

Despite all promises of virtualization, a mixture of different signal processing platforms is important. For instance, the testbed of chapter 5 offers bare metal servers as well as virtual machine (VM) as part of the testbed, because although the server hardware for the virtualization environment might be more powerful than the compute cluster, the real-time performance suffers due to the resource scheduling for the virtual machines. Fig. 7.4 shows the jitter of the processing latency of a VM for the case that the server is

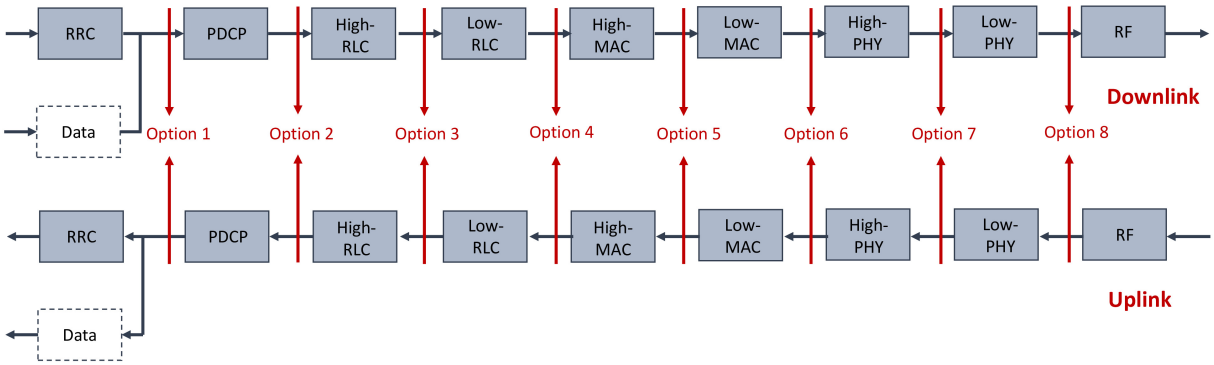


Figure 7.3: Adapted from [199]: The functional split options between the processing units considered during standardization for cloud RANs [199], where a subset of option 1, 2 and 7 was chosen.

idle or loaded with other VM, measured using [200]. In both cases, the maximum latency values are in the range of milliseconds which is already difficult for GNU Radio. Therefore, bare metal servers are still needed to do relevant processing tasks.[3]

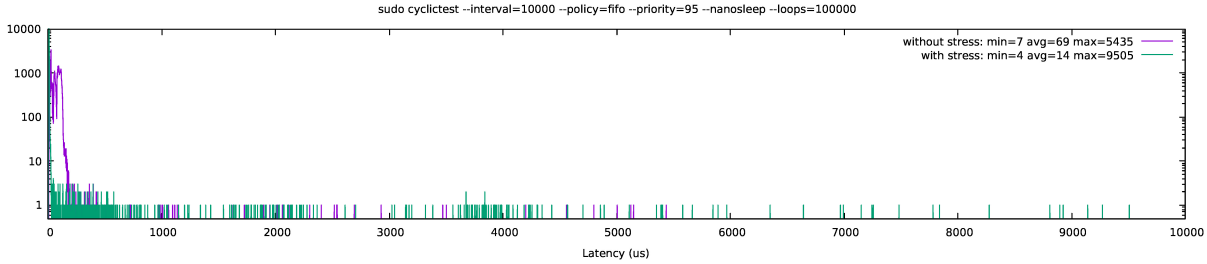


Figure 7.4: Jitter introduced through the virtualization environment.

7.3 Open points

A drawback of the current solution in chapter 4 is that the transfer from a prototype to a more mature state is difficult, because the development cannot easily be exported into reusable components, such as software written in VHDL or Verilog. So, most of the software would have to be reimplemented. Further, it makes it inconvenient to achieve standard compliance to increase the relevance of the experiments. Especially, because higher layer network functions, such as the core network, are missing in the current approach. Higher layers have a big relevance for cellular experiments, because each device has to interact with the core network before it can start transferring data.

The implemented FPGA base-band processor does not consider energy efficiency, because the prototyping platforms and the developments did not aim for this topic. The results of

[3] Docker could help to still have a VM kind of environment, but providing lower latencies.

this work are therefore more relevant for base stations or access points than for clients, since a base station typically can consist of more complex hardware than a (hand held) device. Especially to test the presented transceiver, a prototype client would have to carry either high capacity batteries or a permanent power supply. This could be hindrances with respect to more realistic experiments for instance in industrial scenarios.

Of course many development steps are necessary to transfer an academic project to industry, but a bit further evolved stage could allow interesting experiments with modern software-defined networking techniques, which are seen as key steps for more flexible networks. One idea could be to study network slicing, where different services, e.g. to control a robot, would get different guaranteed parts or slices of the network to respect the quality of service requirements. Here, an implemented full standard compliance Ethernet module is missing in the current implementation.

This work witnessed general limitations as presented in [98] too, with the distinct difference that the SDR platforms provide a better performance and the testbed includes more components. However, splitting up a complex demonstrator into many smaller pieces as suggested in [98] was not successful for the 60 GHz mmWave prototype described earlier either. The MiWaveS project [77] split up the work for developing the 60 GHz frontend across the project partners, and failed providing a compatible and stable working solution, such that in the end a proprietary antenna frontend from *SiBeams* was used.

7.4 Workflow

The introduction and the third chapter presented a work-flow for upcoming experimental platforms, where the transceiver and this testbed can be seen as a playground or extension of the laboratory for basic studies. The aim is to establish a consequent test strategy which is inspirited by RF design patterns. Instead of trying a new idea complete with all challenges, split it into small modules which are tested individually before combining. Further, RF components such as amplifiers for instance are available in many variations from different vendors. However, many data-sheets are hiding the specific issues of a given amplifier [201], such that in the end most relevant amplifiers have to be tested individually to compile a database with known devices. This database can be created continuously as part of the laboratory work, without a specific target. Afterwards, useful components can be picked, for example to be used in a back-plane of a phased array [202].

The presented testbed lays a foundation to create experimental data sets, such as channel impulse responses or captured IQ data, that can be accessed internally as well as externally. For instance, artificial intelligence methods rely on large available training sets, before they can achieve interesting performance. A database, such as [203], with data ordered in meaningful categories can be similarly helpful as existing image databases used to train image recognition [204]. A well-thought data management is essential and

an important topic to follow on. Many methods such as OMF propose SQL databases, which are too static and too customized because of the SQL syntax, other approaches such as MongoDB are more open for flexible data sets.

Despite the problems, the programming flow established in the LabVIEW environment is an example how future software development will look like. Especially, dealing with a huge amount of different components benefits from graphical resprentation to keep track of the overall software. Other work flows, such as GNU Radio or Node-RED, present similar concepts to abstract complexity of algorithms by adapting similar strategies. Although not finalized, the multi-rate diagram [205] in early LabVIEW Communcations versions gave a glimpse on graphical high-level synthesis tool flows.

7.5 Summary

The summary of this chapter can be presented as follows.

1. Prototyping itself is generally considered to be more an engineering task than an academic activity. However, prototyping provides a base for academic research by giving realistic inputs, e.g. to simulation assumptions. Further, it allows to give insights into the overall complexity of a (cloud) communication system, rather than a simulation of a detail of such a system.

2. The overall complexity of a (future) cloud-based communication systems requires a well aligned teamwork of engineers and researchers, which is a challenge on its own.

Prior to developing a prototype, academia needs to provide inputs of recent research questions to understand the issues that need to be addressed. Hence, the challenge is the inter-working between two teams, a prototyping and an algorithm development team. The amount of hardware and software components further reinforces joint works to cope with the various challenges as in the cloud driven multi-robot example presented in Section 6.2.

Thus, academic prototyping should take note of trends such as *Googles* "Site Reliability Engineering" or according to [206] called as "DevOps done right", where normally separated work-domains, such as in this case IT administrators and code developers, have to work in the same team to provide a certain (cloud) function reliable. This principle could be applied to the complexity of practical academic research as well. This means that a prototype development team is needed to further extend and drive academic work with demonstrators. This team needs to consist of multiple members, to deal with hardware (FPGAs), software (CPUs), graphical user interface (GUI) and IT challenges. It should exist continuously to keep the knowledge of developing prototypes with the aim of maintaining a library of building blocks that can be "puzzled" together to form various demonstrators as well as testbeds or experiments. The authors in [207] show how those "DevOps" principles are applied to the Colosseum testbed used by the DARPA spectrum challenge.

Chapter 8

Conclusions

The demand to achieve higher data rates for the Enhanced Mobile Broadband scenario and novel fifth generation use cases like Ultra-Reliable Low-Latency and Massive Machine-type Communications drive researchers and engineers to consider new concepts and technologies for future wireless communication systems. The goal is to identify promising candidate technologies among a vast number of new ideas and to decide, which are suitable for implementation in future products. Thus, this work examines how novel ideas for future wireless networks can be evaluated. Fig. 8.1 outlines the discussed topics and indicates the interactions necessary to form a prototype.

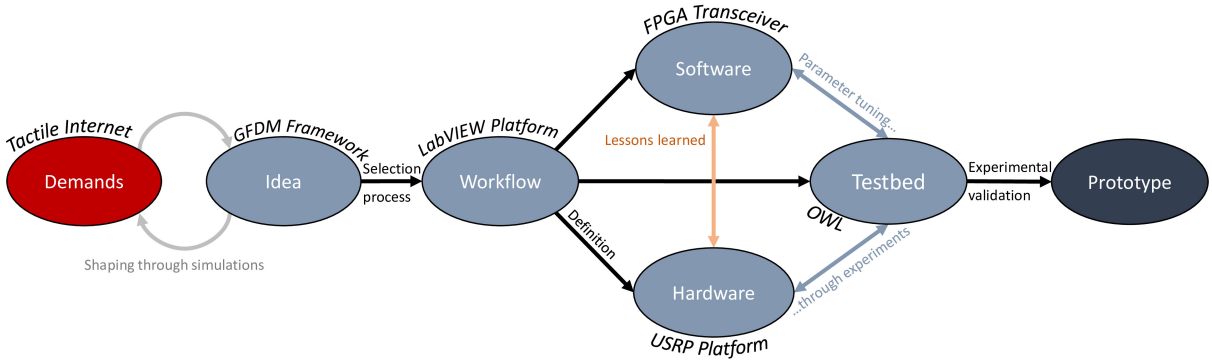


Figure 8.1: From theory to prototype

The first chapter introduces the requirements and demands of novel tactile internet applications on wireless networks. Further, it reveals that there is a large interest and request for prototyping solutions targeting upcoming network generations, especially because new applications have different networking requirements than classical use cases, such as internet access or video streaming. For instance, machine-to-machine communications utilized by wireless robotic applications increases the amount of short messages exchanged per second and have different dependencies on the packet error rate. Thus, the capabilities of the networks have to be adapted for these use cases.

Before a suitable workflow is proposed, the generalized frequency division multiplexing (GFDM) signal processing framework is introduced in the second chapter. This concept has been previously shaped through theoretical work and validated by simulations. Hence, the GFDM idea can bring in more flexibility to wireless communications. The aim is to support very flexible configurations to adapt wireless links to the scenario as well as to emulate other modulation schemes for legacy purposes [54]. As a conclusion, four parameter settings ”LLC”, ”WiFi”, ”eMBB” and ”IoT” were chosen to evaluate the implemented GFDM based transceiver presented in this work.

The third chapter explores suitable workflows and their consequences prior designing a prototype. Here, the technology readiness level (TRL) classification allows to assess the expectations versus the efforts required for the demonstrator, because the preconditions determine remarkably the requirements for the hardware/software platform and the respective trade-offs. Finally, the development expenses are driven by the selected platform, where the selection process is complex due to the amount of available options and their respective (hidden) implications. Every prototype needs to be thoroughly tested to verify its usefulness. So, a testbed encompasses all efforts to demonstrate and evaluate the capabilities of the proposed concept. Additionally, it solicits attention due to the necessary infrastructure and maintenance overhead.

The heart of every networking component is a base-band transceiver software that translates digital information to the analog world and vice versa. According to the varying demands, a flexible base-band design is presented in the fourth chapter. It allows parameter reconfigurations to emulate several waveforms during run-time and not just design-time. The present work uses a design flow where the numerical precision matches the floating-point simulations, without the need for detailed fixed-point models. This time consuming step is usually needed to estimate and adjust the performance of the processor precisely. Further, the expected latency of the transceiver implementation is well below the 1 ms constraint defined for 5G networks. This way the design can be used to evaluate tactile internet applications. The implementation can be run on a single USRP X310 Software-defined radio (SDR) platform to simplify the evaluation of future wireless applications.

Until this point, all experiments were carried out in a laboratory under predefined and predictable conditions. However, new wireless networking solutions need to be tested in more realistic scenarios. Thus, the Online Wireless Lab (OWL) testbed was proposed and built that serves as a permanent experimental environment. The aim is to test novel ideas continuously without extensive preparations beforehand and to use automatic mechanisms for repeatable experiments. The layout of the presented testbed covers different aspects of wireless communication networks due to its: 1) Outdoor, 2) Indoor and 3) Laboratory part. Further, the remote access allows internal as well as external partners to conduct experiments without being physically present. So far, six different universities used the OWL facilities.

The sixth chapter describes the testing of the developed prototyping solutions using the previously defined four parameter settings. Experiments proved that the implementation can achieve the 1 ms latency constraint defined for 5G low-latency networks. Further, the transceiver exhibits similar packet error rates (PERs) in comparison to a commercially developed software framework, such as the *National Instruments (NI)* Long Term Evolution (LTE) Application Framework (AFW) using the same SDR platform. As already defined as a design goal, the architecture offers a higher flexibility than conventional implementations to support various application scenarios. So, the demonstration of a tactile internet prototype utilizing robotic arms in a multi-user or multi-service environment could successfully be achieved.

During the course of this work, several implications occurred, which are discussed in the seventh chapter: "Key Lessons". In principal, prototyping itself is considered to be more an engineering task than an academic activity. However, prototyping provides a base for academic research by giving realistic inputs, e.g. to simulation assumptions. Further, it allows to give insights into the overall complexity of a (cloud) communication system, rather than a simulation of a detail of such a system. Especially, future tactile internet applications increase the complexity of the overall prototyping system, such that a well aligned teamwork of engineers and researchers is required, which is a challenge on its own.

Finally, the developed GFDM based transceiver prototype has been demonstrated in many events with a broad audience, such as CeBIT 2018, Mobile World Congress 2017/2018, Vodafone Innovation Days 2018, 5G Summit Dresden 2017/2018/2019 or connect-ec 2019 Dresden. It took part in various technical conferences, e.g. EuCNC 2017/2018, ICC 2017, Globecom 2017. Further, the robotic demonstrator with the transceiver traveled as part of the Vodafone Enterprise Roadshow 2018 through many locations in Germany and was shown in television programs, such as in ZDF, MDR or in Japanese TV.

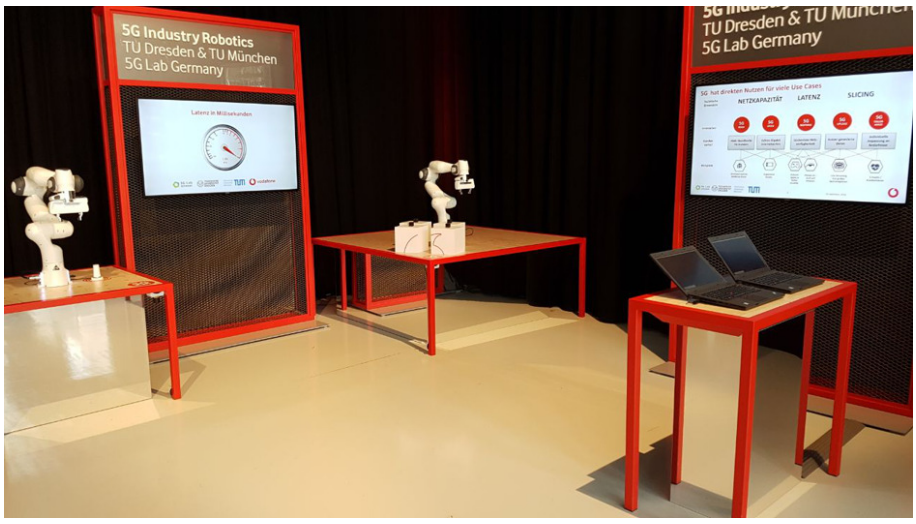


Figure 8.2: Demonstration set-up used during the Vodafone Enterprise Roadshow for high-level representatives.

8.1 Future Work

Fig. 8.1 summarizes the different aspects presented in this work. Although there are interactions amongst them, each aspect can be extended by future work individually as follows in the next sections.

8.1.1 Prototyping Workflow

This work utilized the LabVIEW tool flow developed by *NI*. The aim of LabVIEW is to provide a fully graphical development environment that allows to program complex software by interconnecting symbols with wires, similar to designing circuits using Electronic Design Automation tools. The advantage of this tool flow is that the programs can be executed on field programmable gate arrays (FPGAs) as well as central processing units (CPUs). Further, this way FPGA designs can be implemented much faster than any other workflow available, according to the knowledge of the author. The disadvantage is that the LabVIEW environment is very strict with respect to extensions not foreseen by *NI* and the software is locked to the *NI* hardware platforms. So, this method can be used for prototyping but it is very difficult to create (standalone) products.

Another prototyping workflow provided by *XILINX* is called PYNQ (Python Productivity for ZYNQ) [208]. The idea is to bring Python to the ZYNQ platform. The ZYNQ chip-sets consist of an *ARM* processor and an FPGA placed on the same die. The on-chip interconnections between *ARM* CPU and FPGA allow low latency data transfers between both. One aim is to execute signal processing operations on the FPGA, which are called like a library from code running on the CPU. PYNQ wants to utilize this possibility for the popular Python programming language. Python programs are typically run interactive like scripts and therefore are handy for performing simulations. Normally this reduces the execution speed of the program. Thus, powerful libraries have been added to Python to overcome this limitation. PYNQ extends this concept by offloading signal processing operations, e.g. fast Fourier transform (FFT) to the FPGA. Prototyping novel waveform concept or advanced receiver designs could benefit because the flexibility of CPU programming is kept because most of the program code is running on the *ARM* processor core and thus is avoiding long development cycles. However, selected operations can be accelerated on the FPGA to improve the overall execution speed.

8.1.2 Flexible Transceiver Core

The FPGA implementation presented in chapter 4 can be further optimized in terms of resource usage and flexibility. Fig. 4.10 indicates that most of the signal processing modules are waiting for data input and are therefore idle, except for the maximum case like LTE. Further, most signal processing operations, e.g. Modem, Synchronization etc.,

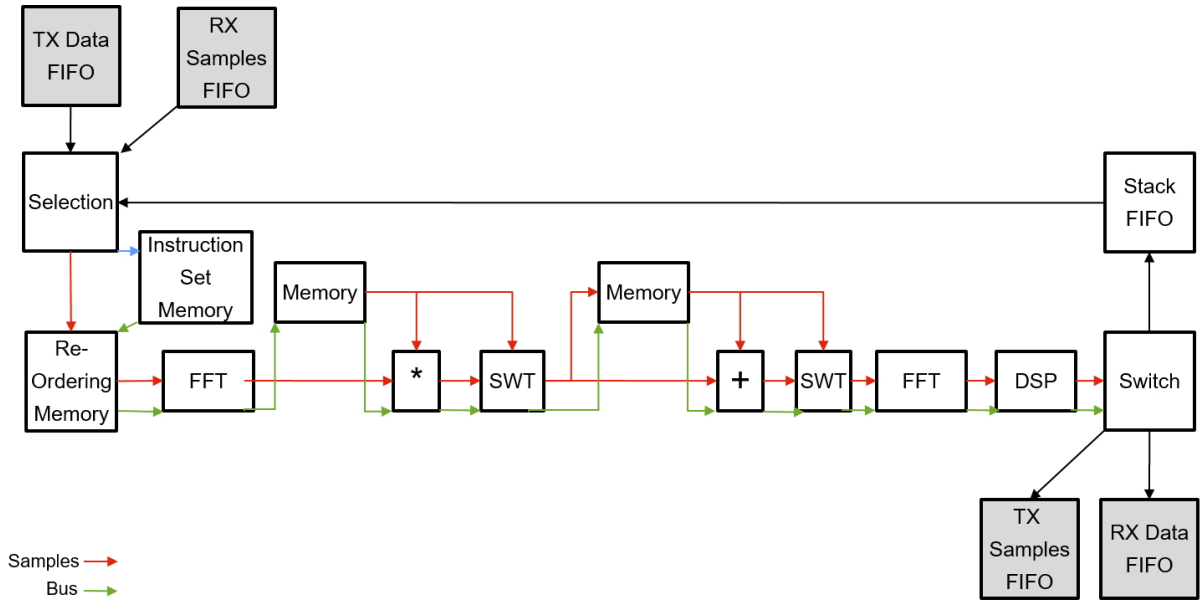


Figure 8.3: Waveform Core

are either performing a fourier transform (FT), multiplying or summing the input with a vector given in a memory or adjusting the scale of the numbers. After those individual processing steps, memory operations take place to rearrange the symbols, e.g. resource mapper or demapper.

The waveform core architecture in Fig. 8.3 outlines a signal processing concept to include those main functionalities. Here, the waveform processing would be event driven, where data is either created by the medium access control (MAC) layer running or by the synchronization algorithm which anyway needs to listen to the radio frequency (RF) signals continuously. Each frame would have to be loop several times to process all features and the interim result would have to be saved after each step in first in, first out (FIFO) buffers used as a stack. The bus architecture would program all components by reading out the commands from a instruction set memory.

A waveform such as GFDM would have to run at least three times through the processing chain, to perform the resource mapping, the modulation process as in [138] and finally to apply cyclic prefix (CP), cyclic suffix (CS) and the window on the transmit signal. Receiving such a waveform would require 4 steps. First the channel estimation has to be performed, then the estimate has to applied onto the received data, the demodulation is the third step and the last step would be the resource de-mapping.

Realizing such a concept would allow to reduce the FPGA footprint of the flexible transceiver, while keeping the functionality in exchange for latency. So, smaller-sized USRPs such as the E3XX series could be supported by this implementation. The advantage is that those USRP have built-in batteries and a hand-held size. This would extend the capabilities of the testbed to more mobile experiments, whereas the current solution is limited to bigger and heavier client prototypes due to the weight and size of

the battery. Furthermore, this flexible concept could inter-work with the PYNQ tool-flow presented in the previous section to demonstrate very advanced transceiver algorithms.

8.1.3 Experimental Data-sets

Especially, the popularity of artificial intelligence triggers the interest in (public) available data-sets. The experiments of chapter 6 can be extended to capture the channel impulse responses experienced during the transmissions, as shown in Fig. 5.9b. Those recorded channel conditions would allow to evaluate which transceiver parameter settings would be more suitable for a given situation. A future advanced transceiver could adapt the parameters in real-time based on the on-going situation to enable a reliable exchange of messages for example. Fig. 6.5 indicates that a single-carrier (SC) waveform is more robust in situations with high attenuation.

The performance was evaluated in static scenarios only. Future work could cover more dynamic scenarios such as a robot platform moving across the testbed. This experiment would allow an more realistic assessment of the capabilities of the implementation. Further, hand-over or coordinated multi point (CoMP) mechanisms could be studied in this scenario. The outdoor part or a portable testbed installation could extend the work with vehicular experiments additionally.

The developed measurement software may be used to study state-of-the-art technologies and products. The latency evaluation concerning the switches and operating systems could be continued to judge their capabilities. For instance, the *AVM* Fritz!Box 4040 was not able to handle the amount of packets generated by the *Franka Emika* robotic arms. Furthermore, the behavior of (real-time) applications running in virtualized environments could be of interest for modern communication systems.

8.1.4 Evolved Access Point Prototype For Industrial Networks

The idea of a very flexible FPGA transceiver can be further pursued to develop a flexible access point application. Especially, 5G NR and beyond technologies introduce more flexibility and adaptivity to communication networks than previous standards. An evolved access-point could make use of the flexibility, because 5G flexible numerologies allows to adapt the properties of the physical layer to the applications and/or wireless scenarios as presented in [165]. This includes for example the subcarrier spacing, bandwidth and the resource allocation. An overview of the radio-slicing concept for cellular communications is given in [209], where the underlying technique is a 5G NR cellular network and its respective orthogonal frequency division multiplexing (OFDM) waveform.

In contrast to cellular networks, access points in industrial scenarios should support network slicing to enable very different communication types with different radio access techniques, particularly including a change of the physical layer waveform itself, such as

indicated in Fig. 1.1. The reason is that various types of wireless technologies have to be connected, because diverse applications need access to cloud platforms. Harmonizing all those applications to use strictly one technology is a non-trivial challenge. For example, legacy systems cannot be updated to newer standards and embedded sensors have to deal with low energy consumption and therefore cannot use complex communication systems. Typically, several and independent access points (APs) are used. Especially their non-synchronized operation leads to interference that results in reduced network performance as experimental studies in the EU project eWINE show [65].

Decoupling User Plane and Control Plane and controlling radio channels with software-defined networking (SDN) could solve this issue of radio access network (RAN) slicing, if all involved network elements at the AP support multiple technologies in a harmonized approach. The idea is to use one flexible physical layer (PHY) chipset, instead of multiple PHY chipsets. This has the advantage that only one chipset with a higher quality RF chain can be used rather than several independent ones. More important, also the timings between the different PHY-services can be aligned. Thus, the inter-service interferences can be reduced. This allows using the frequency bands more efficiently. However, since only one chipset is going to be utilized the parameters of the signal processing must be reconfigured quickly.

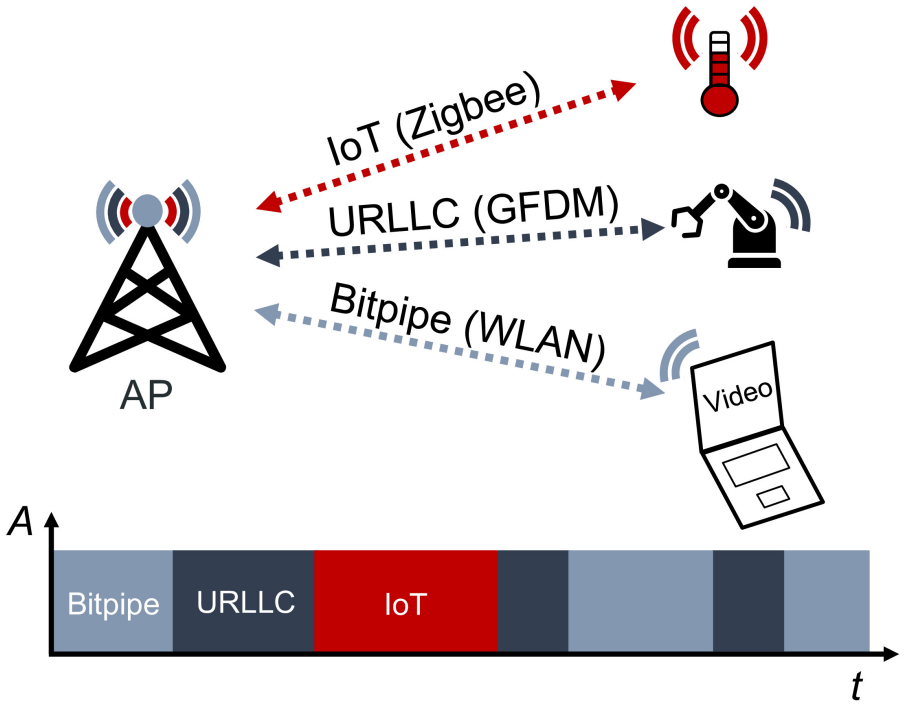


Figure 8.4: Evolved access point with SDN functionalities offers various services using a reconfigurable and flexible physical layer chipset, taken from [165].

Fig. 8.4 presents the envisioned communication system where one AP provides different clients with different services using different numerologies. In this case, IoT sensors are served with ZigBee communications for legacy reasons, whereas the robot receives control

information provided by a cloud-based controller with an Ultra-Reliable and Low-Latency Communication (URLLC) service. Since industrial, scientific and medical bands (ISMs) are not restricted to certain physical layers, a modified 5G NR or other PHY proposals could offer the URLLC service. Finally, a wireless local area network (WLAN) is provided for non-critical but high-throughput clients, all operating in the 2.4 GHz band.

The goal is that one AP operates on a given frequency band for its services, such that the other bands can be used by neighboring APs, without overlapping to prevent interference. The multi-standard capabilities of one AP enable serving the clients in a time-aligned approach to prevent any inter-service interference within the range of the AP, instead of relying on carrier sense multiple access (CSMA) to negotiate which service gets access to the medium. Here, a scheduled airtime is given to a service where all other services are "turned off".

8.1.5 Testbed Standardization

Fig. 8.1 shows that a testbed combines the various efforts for demonstrating the feasibility of an idea. The first and third chapter introduce very different examples of testbeds. However, the authors in [30] state that there is a lack of common definitions and collaboration with respect to test infrastructures, because different initiatives across the globe are creating testbed islands. In many cases, those testbed islands are nontransparent for the outside community due to a missing common understanding and because only very few testbeds are fully public. Further, the often proprietary interfaces make them non connectable to other initiatives. The seventh chapter indicates that prototyping of future tactile internet applications is complex due to the vast amount of inter-connected software and hardware components. Thus, the emerging challenges can only be handled in teams with various specialized members.

The IEEE Future Networks Testbed Working Group wants to tackle those challenges by [30]:

- ▷ "Defining common semantics and/or vocabularies (including metrics)",
- ▷ "Common and federated Authentication, Authorization and Accounting mechanisms and services",
- ▷ "Introduction and use of common control, management, and orchestration mechanisms and APIs",
- ▷ "Introduction of standards for data-sharing",
- ▷ and "Testbed operations monitoring".

The aim defined in [30] is to improve the learning curves for experimenters through providing a federation of testbeds with harmonized access mechanisms and policies. Further, this federation will take advantage of the flexible underlying technologies

to maximize the resource efficiency and lessens the manual involvement. Finally, the simplifications of the testbed usage will reduce the lead cycle from prototypes to deployment and will allow to develop novel standards and ideas in an unified and collaborative way.

Appendix A

Additional Resources

A.1 Fourier Transform Blocks

To facilitate many different waveforms a configurable FT block was used on the FPGA. This block consisted of the *Xilinx* FFT 9.0 and DFT 4.1 IP core. A wrapper around both cores allowed to switch between both to meet the necessary FT size requirement. Depending on the selection different processing delays can be observed in Tab. A.1.

A.2 Resource Consumption

The resource utilization is estimated through compiling all components separately on the FPGA, where the compiler has more freedom to place the elements as in case of the full transceiver. Still, it allows a insight about the consumption of the various signal processing blocks. So, the summation of all values might be different from the total amount. The overall slice count is an indicator how the FPGA resources are exploited, because one slice is the smallest entity in an FPGA [167] and consists of 4 look-up tables (LUTs), 8 flip-flops, multiplexers and carry logic. These elements provide basic logic, arithmetic and ROM functions based on the program. A *XILINX* FPGA arranges two slices in one configurable logic block that is connected via a switch matrix to the general routing matrix of the FPGA. In addition, specialized elements are part of the FPGA fabric, such as dedicated block memory, digital signal processors (DSPs), registers and LUTs.

For the implemented transceiver the individual signal processing modules shall be discussed. The modules named as ”Mapping” and described in subsection 4.1.2 contain the encoder, decoder, scrambling, interleaving and resource mapping functions. Thus, they are utilizing mainly slices, LUTs and memories to scramble and reorder the data. Both modems (subsection 4.1.3) consume DSPs and memory elements, due to the FFTs and the circular convolutions. There the pulse shaping filter and the data are stored in parallel memory banks and are multiplied with each other. The final frame is assembled

Size	Latency	Size	Latency	Size	Latency
8	74	180	592	600	1962
12	75	192	565	648	2225
16	143	216	732	720	2265
24	122	240	736	768	2214
32	190	256	905	864	2855
36	152	288	918	900	2952
48	176	300	955	960	2901
60	227	324	1074	972	3359
64	308	360	1191	1024	3229
72	271	384	1158	1080	3716
96	325	432	1362	1152	3671
108	373	480	1509	1200	3792
120	418	512	1672	1296	4331
128	499	540	1773	1536	4727
144	457	576	1734	2048	6300

Table A.1: Latency in FPGA clock cycles of the *Xilinx* FFT and DFT core

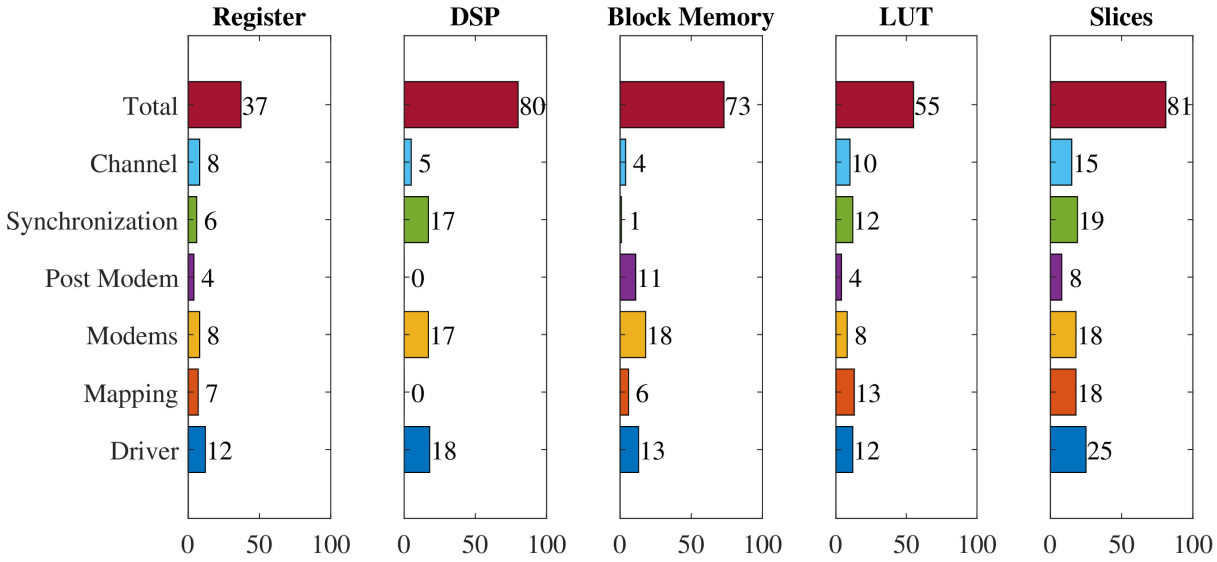


Figure A.1: Estimated resource consumption of the individual signal processing blocks in percent of the available resources.

in the ”Post Modem” blocks (subsection 4.1.4). This highly depends on the availability of memories, such that the CP, CS can be created, the window can be applied and the preamble is multiplexed in front of the transmit sequence. On the receiver side, the

synchronization follows first and performs correlation operations that are realized in this implementation based on DSPs. Although, the channel estimation and equalization blocks contribute with highest latency at the receiver, they are consuming a moderate amount of resources, compared to the other signal processing modules, where DSPs, as well as LUTs are used in several FFT modules.

A.3 Channel Sounding using Chirp sequences

The channel sounding presented in the following section is based on [76]:

The discrete-time reference down-chirp is defined by the Polyphase code [210] and is arranged in the following column vector

$$\mathbf{g}_{\mathrm{d}} = \left(\exp\left(-j\pi\frac{n^2}{N}\right)\right)_{n=0,1,\cdots,N-1}. \tag{A.1}$$

From (A.1), we construct 5 subsequent down-chirps sequences as $\tilde{\mathbf{g}}_{\mathrm{d}} = \left(\mathbf{g}_{\mathrm{d}}^{\mathrm{T}}, \mathbf{g}_{\mathrm{d}}^{\mathrm{T}}, \mathbf{g}_{\mathrm{d}}^{\mathrm{T}}, \mathbf{g}_{\mathrm{d}}^{\mathrm{T}}, \mathbf{g}_{\mathrm{d}}^{\mathrm{T}}\right)^{\mathrm{T}}$, where $(\cdot)^{\mathrm{T}}$ stands for the transposition operator. Then, we define a sequence of 5 up-chirps as $\tilde{\mathbf{g}}_{\mathrm{u}} = \left(\mathbf{g}_{\mathrm{d}}^{\mathrm{H}}, \mathbf{g}_{\mathrm{d}}^{\mathrm{H}}, \mathbf{g}_{\mathrm{d}}^{\mathrm{H}}, \mathbf{g}_{\mathrm{d}}^{\mathrm{H}}, \mathbf{g}_{\mathrm{d}}^{\mathrm{H}}\right)^{\mathrm{T}}$, where $(\cdot)^{\mathrm{H}}$ stands for the conjugate transposition. Notice that the conjugate of $\mathbf{g}_{\mathrm{d}}$ in (A.1) changes the sign inside the exponential argument, transforming the signal into an up-chirp. The transmitted sequence is

$$\mathbf{x} = \left(\tilde{\mathbf{g}}_{\mathrm{d}}^{\mathrm{T}}, \tilde{\mathbf{g}}_{\mathrm{u}}^{\mathrm{T}}, \mathbf{0}_{5N}^{\mathrm{T}}\right)^{\mathrm{T}}, \tag{A.2}$$

which is composed of the 5 down-chirps, 5 up-chirps and a vector of $5N$ zeros which is given by $\mathbf{0}_{5N}$. The zero sequence is included to allow us to estimate the noise power and consequently the SNR. Additionally, we highlight that the combination of up and down-chirps can be used to estimate integer carrier frequency offset (CFO)[1], as shown in [211]. At transmitter, we simply transmit the signal of (A.2) repeatedly.

A.3.1 SNR Estimation

In order to estimate the SNR, we need to synchronize the received sequence in time-domain. First, we notice that the maximum CFO normalized to subcarrier spacing of the chirp $\mathbf{g}_{\mathrm{d}}$ is very small. In particular, it is calculated as $\phi_{\max} = \frac{f_{\mathrm{acc}} \times f_{\mathrm{c}}}{B/N} = 0.048$, where $f_{\mathrm{acc}} = 4 \times 10^{-6}$ is the frequency accuracy of the USRP B205mini's oscillator, $f_{\mathrm{c}} = 3.75$ MHz is the carrier frequency, $B = 20$ MHz is the bandwidth (or sampling rate) and $N = 64$ is the length of $\mathbf{g}_{\mathrm{d}}$. Because $\phi_{\max}$ is small, we can synchronize the sequence in

[1] In this work, it was not necessary to estimate the CFO due to small CFO range. This sequence is designed for more general case.

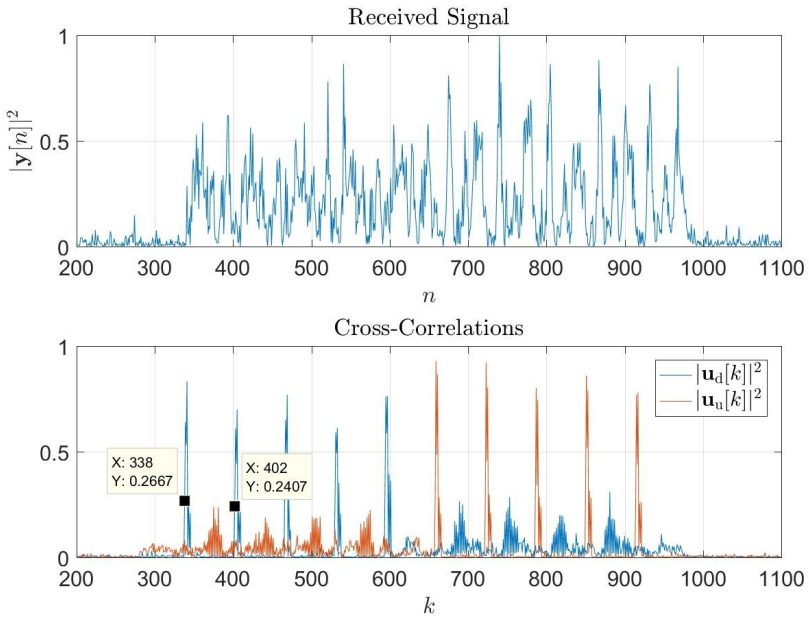


Figure A.2: Time synchronization example, taken from [76].

time very accurately by performing the cross-correlation with the down and up-chirps, respectively, as

$$\mathbf{u}_\mathrm{d}[k] = \frac{1}{N}\sum_{n=0}^{N-1} \mathbf{r}[k+n]e^{-j\pi\frac{n^2}{N}},\ \mathbf{u}_\mathrm{u}[k] = \frac{1}{N}\sum_{n=0}^{N-1} \mathbf{r}[k+n]e^{j\pi\frac{n^2}{N}}, \tag{A.3}$$

We then compare $|\mathbf{u}_\mathrm{d}[k]|^2$ and $|\mathbf{u}_\mathrm{u}[k]|^2$ to a threshold γ, where $|\cdot|$ returns the absolute value of its argument. The received signal is declared synchronized if $|\mathbf{u}_\mathrm{d}[\hat{k}+iN]|^2 > \gamma$ and $|\mathbf{u}_\mathrm{u}[\hat{k}+iN+5N]|^2 > \gamma$ simultaneously for all $i = 0, 1, 2, 3, 4$. In this case, the sequence starts at $\mathbf{y}[\hat{k}]$. Figure A.2 depicts the idea of using $\mathbf{u}_\mathrm{d}[k]$ and $\mathbf{u}_\mathrm{p}[k]$ for time synchronization, where we considered a fixed channel with $L = 8$ taps and SNR equal to 10 dB. One can observe that well-defined peaks appear in the cross correlation functions. More precisely, they are separated by $N = 64$ samples, as it is shown for the first two local peaks.

Afterwards, we can estimate the SNR by

$$\hat{\rho} = \frac{\frac{1}{10N}\sum_{i=0}^{10N-1}|\mathbf{y}[i+\hat{k}]|^2}{\frac{1}{5N-p}\sum_{i=p}^{5N-1}|\mathbf{y}[i+\hat{k}+10N]|^2} - 1, \tag{A.4}$$

where the numerator in (A.4) contains signal plus noise, providing an estimate of both signals added. The denominator contains only noise, providing an estimate of the noise power only, where the parameter p determines a shift on where we start estimating the noise power and is chosen to guarantee a noise only signal while estimating the noise power. Finally, we subtract 1 because the fraction provides an estimation of $(P_\mathrm{signal}+P_\mathrm{noise})/P_\mathrm{noise}$ which equals SNR + 1, resulting in an unbiased estimation. In practice, we average the SNR estimation in (A.4) over several chirp sequences for refinement.

A.3.2 Channel Estimation

Before performing the channel estimation, we need to correct the CFO, otherwise the reference chirp is shifted in frequency. The CFO estimation is done by averaging the phase of the auto-correlation function as in [211],

$$\hat{\phi} = \frac{1}{6\times 2\pi} \sum_{i=1}^{3} \arg\left\{\mathbf{m}[\hat{k}+iN]\right\} + \arg\left\{\mathbf{m}[\hat{k}+iN+5N]\right\}, \tag{A.5}$$

where $\mathbf{m}[k] = \frac{1}{N}\sum_{n=0}^{N-1} \mathbf{y}[k+n]^{\dagger}\mathbf{y}[k+n+N]$ is the auto-correlation function and $\arg\{\cdot\}$ returns the phase of the argument. Notice that (A.5) averages the phase of the auto-correlation function starting from the second down and up-chirps, since the channel is not circulant over the first chirps. After CFO compensation, the channel response in the frequency domain can be estimated using the zero forcing (ZF) approach

$$\mathbf{h}_{\mathrm{F}} = \frac{1}{8}\sum_{i=1}^{4} \frac{\mathbf{R}_{\hat{k}+iN}}{\mathrm{FFT}\{\mathbf{g}_{\mathrm{d}}\}} + \frac{\mathbf{R}_{\hat{k}+iN+5N}}{\mathrm{FFT}\{\mathbf{g}_{\mathrm{u}}\}}, \tag{A.6}$$

where $\mathbf{R}_q = \mathrm{FFT}\{(\mathbf{r}[q], \mathbf{r}[q+1], \cdots, \mathbf{r}[q+N-1])\}$ is the fast Fourier transform of the received signal starting in the index q with size N. Notice that by choosing $i = \{1,2,3,4\}$, we select the portion of the signal in which the channel is circulant over the chirps, allowing us to estimate the channel in the frequency domain as (A.6).

A.4 Hardware part list

The following list includes most relevant equipment used for the testbed for reference reasons:

Type	Manufacturer	Device	Description
Network	Cisco	c9500-48y4c Switch	Core Network switch
Network	FS.COM	SFP-10G-LR	Programmable SFP module for LWL
Network	FS.COM	100GBASE-LR4 QSFP28 Cisco	100 GBit SFP LWL adapter
Network	Meinberg	LANTIME M300	NTP Time Server
Network	FibroLan	µFalcon-MX-G 10G Switch	10 GBit Switch + PTP Master
Network	FibroLan	µFalcon-MX-G 10G Switch	10 GBit Switch + PTP Slave with TTL PPS and 10 MHz Output
Network	Netgear	M4300-8X8F	Stackable 16 Port 10G Network Switch
Computing	NI	PXIe-8880	Compute Node
Computing	HPE	ProLiant m710	Compute Node
Computing	HPE	ProLiant m510	Compute Node
SDR	NI	USRP 2974	USRP x310 compatible SDR + x86 CPU platform
SDR	NI	USRP 2944	USRP x310 compatible SDR
SDR	NI	USRP B205	Small size SDR

Table A.2: Compute and networking platforms used and evaluated for the testbed

Type	Manufacturer	Device	Description
RF	Tactron	AMP-2.5-500-6000-25-SMAF	0.5 W Amplifier 0.5-6.0GHz, 25dB Gain
RF	Mini Circuits	VLM-63-2W-S+	10 dBm Power Limiter
RF	Mini Circuits	ZX60-83LN12+	Low Noise Amplifier, 500 - 8000 MHz
RF	G-Wave	CB3750/90SK-D1	Bandpass filter 3705 - 3795 MHz
RF	Mini Circuits	ZVE-8G+	1 W Power amplifier, 2000 - 8000 MHz
RF	Alfa Network	ALR62NF	Lightning Protection
RF	Mini Circuits	ZFSWA2R-63DR+	Absorptive SPDT, Solid State Switch, 500 - 6000 MHz, 35ns
Antenna	MobilMark	MGRM-WHF	Broadband antenna with magnetic foot
Antenna	Ubiquiti	AM-3G18-120	3.3-3.8 GHz AirMax 2x2 MIMO sectorized antenna
Antenna	WiBOX	PA M3-14X	Sectorized antenna 3.4 - 3.8 GHz
Antenna	Panorama Antennas	LGMQM 47382458	4X4 MIMO LTE+WLAN+GPS antenna

Table A.3: RF components used and evaluated for the testbed

Bibliography

[1] *Franka Control Interface (FCI) Documentation*, Franka Emika GmbH, https://frankaemika.github.io/docs/libfranka.html, 2017.

[2] A. Bentkus, "5G for Connected Industries and Automation," Whitepaper. [Online]. Available: https://www.5g-acia.org/publications/5g-for-connected-industries-and-automation-white-paper/

[3] F. Piller, "Mass Customization," RWTH Aachen TIM Group, http://frankpiller.com/mass-customization/, Research Note.

[4] D. Jacob, "Shaping Future Production Landscapes," KUKA AG, Whitepaper.

[5] G. P. Fettweis, "The Tactile Internet: Applications and Challenges," *IEEE Vehicular Technology Magazine*, vol. 9, no. 1, pp. 64–70, March 2014.

[6] F. Kaltenberger, R. Knopp, M. Danneberg, and A. Festag, "Experimental Analysis and Simulative Validation of Dynamic Spectrum Access for Coexistence of 4G and Future 5G Systems," in *2015 European Conference on Networks and Communications (EuCNC)*, June 2015, pp. 497–501.

[7] H. Schubert, "Funkanlagenrichtlinie (RED): EU knüpft sich rekonfigurierbare Funksysteme vor," *Elektroniknet.de*, May 2020. [Online]. Available: https://www.elektroniknet.de/elektronik/kommunikation/eu-knuepft-sich-rekonfigurierbare-funksysteme-vor-176432.html

[8] J. Mitola, ""SDR Architecture for US Tactical Radios"," in *Software Radio*, E. Del Re, Ed. London: Springer London, 2001, pp. 157–164.

[9] "Speakeasy," Jan 2020, accessed: Nov. 10, 2020. [Online]. Available: https://en.wikipedia.org/wiki/SpeakEasy

[10] "History of the Forum and the SCA," *The Wireless Innovation Forum*, 2018. [Online]. Available: https://www.wirelessinnovation.org/what_is_the_sca

[11] "Software Defined Radio: Past, Present, and Future," National Instruments, Whitepaper. [Online]. Available: https://www.ni.com/de-de/innovations/white-papers/17/software-defined-radio--past--present--and-future.html

[12] S. Gallagher, "How to Blow $6 Billion on a Tech Project," *Ars Technica*, Jun 2012. [Online]. Available: https://arstechnica.com/information-technology/2012/06/how-to-blow-6-billion-on-a-tech-project/

[13] A. Sawall, "ORAN: Open-Source-Mobilfunk ist nicht umweltfreundlich," *Golem.de - IT-News für Profis*, Feb 2020. [Online]. Available: https://www.golem.de/news/oran-open-source-mobilfunk-ist-nicht-umweltfreundlich-2002-146650.html

[14] K. Thoma, *European Perspectives On Security Research*, ser. acatech DISKUTIERT. Springer-Verlag Berlin Heidelberg, 2011.

[15] J. Mitola and G. Q. Maguire, "Cognitive Radio: Making Software Radios more Personal," *IEEE Personal Communications*, vol. 6, no. 4, pp. 13–18, 1999.

[16] R. Datta, N. Michailow, S. Krone, M. Lentmaier, and G. Fettweis, "Generalized Frequency Division Multiplexing in Cognitive Radio," in *2012 Proceedings of the 20th European Signal Processing Conference (EUSIPCO)*, 2012, pp. 2679–2683.

[17] *Payload*, SIGFOX, https://build.sigfox.com/payload, Platform Information.

[18] V. Ç. Güngör and G. P. Hancke, Eds., *Industrial Wireless Sensor Networks*. CRC Press, Dec 2017.

[19] M. Gundall, J. Schneider, H. D. Schotten, M. Aleksy, D. Schulz, N. Franchi, N. Schwarzenberg, C. Markwart, R. Halfmann, P. Rost, D. Wübben, A. Neumann, M. Düngen, T. Neugebauer, R. Blunk, M. Kus, and J. Grießbach, "5G as Enabler for Industrie 4.0 Use Cases: Challenges and Concepts," in *2018 IEEE 23rd International Conference on Emerging Technologies and Factory Automation (ETFA). IEEE International Conference on Emerging Technologies and Factory Automation (ETFA-2018)*. IEEE, 2018.

[20] "IEEE Standard for System, Software, and Hardware Verification and Validation," *IEEE Std 1012-2016*, pp. 1–260, 2017.

[21] Raytheon BBN Technologies, "What is GENI?" *GENI*. [Online]. Available: https://www.geni.net/about-geni/what-is-geni/

[22] European Commission, "EU-Funded Projects on Future Internet Research & Experimentation," Oct 2017. [Online]. Available: https://ec.europa.eu/digital-single-market/en/programme-and-projects/project-factsheets-future-internet-research-experimentation

[23] PAWR, "Platforms for Advanced Wireless Research." [Online]. Available: https://advancedwireless.org/

[24] D. CONNECT, "Ex-post Evaluation of ICT Research in the Seventh Framework Programme," Tech. Rep., 2015. [Online]. Available: https://www.kowi.de/Portaldata/2/Resources/fp7/FP7-ICT-report-ex-post-evaluation.pdf

[25] 5GPPP, "5G Research in Horizon 2020," 2014. [Online]. Available: https://5g-ppp.eu/european-5g-actions/

[26] T. Kadur, H. Chiang, and G. Fettweis, "Experimental Validation of Robust Beam Tracking in a NLoS Indoor Environment," in *2018 25th International Conference on Telecommunications (ICT)*, June 2018, pp. 644–648.

[27] R. Eini, L. Linkous, N. Zohrabi, and S. Abdelwahed, "A Testbed for a Smart Building: Design and Implementation," in *Proceedings of the Fourth Workshop on International Science of Smart City Operations and Platforms Engineering*, ser. SCOPE '19. New York, NY, USA: Association for Computing Machinery, 2019, p. 1–6. [Online]. Available: https://doi.org/10.1145/3313237.3313296

[28] P. Marsch and G. Fettweis, Eds., *Coordinated Multi-Point in Mobile Communications: From Theory to Practice*. Cambridge University Press, 2011.

[29] ORBIT Consortium, "Open-Access Research Testbed for Next-Generation Wireless Networks." [Online]. Available: http://www.orbit-lab.org/

[30] International Network Generations Roadmap, "Testbed," IEEE, Tech. Rep. First Edition.

[31] S. Pollin, "Technical Overview of the ORCA Project," Sep 2018, CROWNCOM 2018. [Online]. Available: https://www.orca-project.eu/resources/presentations/

[32] S. Berger, M. Danneberg, P. Zanier, I. Viering, and G. Fettweis, "Experimental Evaluation of the Uplink Dynamic Range Threshold," *EURASIP Journal on Wireless Communications and Networking*, vol. 2015, no. 1, Oct. 2015. [Online]. Available: https://doi.org/10.1186/s13638-015-0447-6

[33] E. McCune, *Practical Digital Wireless Signals*, ser. The Cambridge RF and Microwave Engineering Series. Cambridge University Press, 2010.

[34] T. Frenzel, J. Rohde, and J. Opfer, "Elektromagnetische Schirmung von Gebäuden," Bundesamt für Sicherheit in der Informationstechnik, Tech. Rep. BSI TR-03209 - 1, April 2008.

[35] H. Nuszkowski, *Digitale Signalübertragung im Mobilfunk*. Vogt Verlag, 2010.

[36] Technical Specification Group Radio Access Network, "Evolved Universal Terrestrial Radio Access (E-UTRA); User Equipment (UE) Radio Transmission and Reception." 3rd Generation Partnership Project, Tech. Rep. TS 36.101, 2002.

[37] H. Friis, "Noise Figures of Radio Receivers," in *Proceedings of the IRE*, vol. 32, 1944, pp. 419–422.

[38] Y. Segal, Z. Hadad, I. Kitroser, and Y. Lieba, "OFDM Preamble Structure Analysis and Proposal," IEEE, Tech. Rep. IEEE 802.16abc-01/04, 2001.

[39] D. Tse and P. Viswanath, *Fundamentals of Wireless Communication*. Cambridge: Cambridge University Press, 2005.

[40] R. van Nee, "OFDM Physical Layer Specification for the 5 GHz Band," Tech. Rep. IEEE P802.11-98/12, jan 1998.

[41] *Concepts of Orthogonal Frequency Division Multiplexing (OFDM) and 802.11 WLAN*, Keysight Technologies, Inc. [Online]. Available: http://rfmw.em.keysight.com/wireless/helpfiles/89600B/WebHelp/Subsystems/wlan-ofdm/content/ofdm_basicprinciplesoverview.htm

[42] F. Pancaldi, G. M. Vitetta, R. Kalbasi, N. Al-Dhahir, M. Uysal, and H. Mheidat, "Single-Carrier Frequency Domain Equalization," *IEEE Signal Processing Magazine*, vol. 25, no. 5, pp. 37–56, 2008.

[43] R. Lavoie, *SC-FDMA: A Low Peak-To-Average-Power OFDM Modulation Scheme*, NuRAN Wireless Inc., Jan 2017. [Online]. Available: https://www.nutaq.com/blog/sc-fdma-low-peak-average-power-ofdm-modulation-scheme

[44] H. Kiesler and G. Antheil, "Secret Communication System," Patent US2 292 387A, Aug, 1942.

[45] G. Wunder, P. Jung, M. Kasparick, T. Wild, F. Schaich, Y. Chen, S. T. Brink, I. Gaspar, N. Michailow, A. Festag, L. Mendes, N. Cassiau, D. Ktenas, M. Dryjanski, S. Pietrzyk, B. Eged, P. Vago, and F. Wiedmann, "5GNOW: Non-Orthogonal, Asynchronous Waveforms for Future Mobile Applications," *IEEE Communications Magazine*, vol. 52, no. 2, pp. 97–105, 2014.

[46] M. Danneberg, A. Nimr, N. Michailow, M. Matthé, A.-B. Martinez, D. Zhang, and G. o. Fettweis, "Universal Waveforms Processor," in *EuCNC 2018*, pp. 357–362.

[47] N. Michailow, "Generalized Frequency Division Multiplexing Transceiver Principles," Ph.D. dissertation, 2016.

[48] I. Gaspar, "Waveform Advancements and Synchronization Techniques for Generalized Frequency Division Multiplexing," Ph.D. dissertation, 2016.

[49] M. Matthé, "Multiple-Input Multiple-Output Detection Algorithms for Generalized Frequency Division Multiplexing," Ph.D. dissertation, 2018.

[50] S. Ehsanfar, "Advanced Channel Estimation Techniques for Multiple-Input Multiple-Output Multi-Carrier Systems in Doubly-Dispersive Channels," Ph.D. dissertation, 2020.

[51] I. Gaspar, L. L. Mendes, M. Matthé, N. Michailow, D. Zhang, A. Albertiy, and G. P. Fettweis, "GFDM - A Framework for Virtual PHY Services in 5G Networks," *CoRR*, vol. abs/1507.04608, 2015. [Online]. Available: http://arxiv.org/abs/1507.04608

[52] "PHYDYAS Project." [Online]. Available: http://www.ict-phydyas.org/

[53] F. Kaltenberger, R. Knopp, C. Vitiello, M. Danneberg, and A. Festag, "Experimental Analysis of 5G Candidate Waveforms and their Coexistence with 4G Systems," in *2015 European Conference on Networks and Communications (EuCNC)*, 2015.

[54] M. Matthé, I. S. Gaspar, L. L. Mendes, D. Zhang, M. Danneberg, N. Michailow, and G. Fettweis, *Generalized Frequency Division Multiplexing: A Flexible Multi-Carrier Waveform for 5G*. Cham: Springer International Publishing, 2017, pp. 223–259. [Online]. Available: https://doi.org/10.1007/978-3-319-34208-5_9

[55] M. Danneberg, R. Datta, A. Festag, and G. Fettweis, "Experimental Testbed for 5G Cognitive Radio Access in 4G LTE Cellular Systems," in *2014 IEEE 8th Sensor Array and Multichannel Signal Processing Workshop (SAM)*, 2014, pp. 321–324.

[56] G. Cherubini, E. Eleftheriou, and S. Olcer, "Filtered Multitone Modulation for VDSL," in *Seamless Interconnection for Universal Services. Global Telecommunications Conference. GLOBECOM'99. (Cat. No.99CH37042)*, vol. 2, Dec 1999, pp. 1139–1144 vol.2.

[57] Y. Medjahdi, R. Zayani, H. Shaïek, and D. Roviras, "WOLA Processing: A Useful Tool for Windowed Waveforms in 5G with Relaxed Synchronicity," in *Communications Workshops (ICC Workshops), 2017 IEEE International Conference on*. IEEE, 2017, pp. 393–398.

[58] M. Loy, R. Karingattil, and L. Williams, "ISM-Band and Short Range Device Regulatory Compliance Overview," May 2005. [Online]. Available: http://www.ti.com/lit/an/swra048/swra048.pdf

[59] "Frequenzplan," Bundesnetzagentur, Tech. Rep., Oct 2019. [Online]. Available: https://www.bundesnetzagentur.de/DE/Sachgebiete/Telekommunikation/Unternehmen_Institutionen/Frequenzen/Grundlagen/Frequenzplan/frequenzplan-node.html

[60] "5G Spectrum Fees for Local Usages," Oct 2019. [Online]. Available: https://www.bundesnetzagentur.de/SharedDocs/Pressemitteilungen/EN/2019/20191031_LokalesBreitband.html

[61] F. Barrau, B. Paille, E. Kussener, and D. Goguenheim, "Distance Measurement Using Narrowband ZigBee Devices," in *2014 23rd Wireless and Optical Communication Conference (WOCC)*, May 2014, pp. 1–6.

[62] "IEEE 802.15.4," Aug 2019, accessed: Nov. 10, 2020. [Online]. Available: https://de.wikipedia.org/wiki/IEEE_802.15.4

[63] "Distributed Coordination Function," May 2020, accessed: Nov. 10, 2020. [Online]. Available: https://en.wikipedia.org/wiki/Distributed_coordination_function

[64] J. Crane, "ZigBee and WiFi Coexistence," *MetaGeek*, Feb 2018. [Online]. Available: https://www.metageek.com/training/resources/zigbee-wifi-coexistence.html

[65] "Machine Learning Algorithms Development and Implementation," eWINE Project Consortium, Tech. Rep., Dec 2016. [Online]. Available: https://ewine-project.media.martel-innovate.com/wp-content/uploads/sites/29/2017/01/eWINE_D5.1_Final_v1.0.pdf

[66] Y. Hwang, "Unlicensed LTE Explained - LTE-U vs. LAA vs. LWA vs. Multefire," *Explanations & Tutorials*, Jun 2017. [Online]. Available: https://www.leverege.com/blogpost/unlicensed-lte-lte-u-vs-laa-vs-lwa-vs-multefire

[67] A. Feng, "An Overview Of WirelessHART's OSI Layers," *WirelessHART Made Easy*, Nov 2011. [Online]. Available: https://www.awiatech.com/an-overview-of-wirelesshart%e2%80%99s-osi-layers/

[68] S. Goodley, "O2 Poised to Receive Millions from Ericsson over Software Failure," *The Guardian*, Dec 2018. [Online]. Available: https://www.theguardian.com/business/2018/dec/09/o2-poised-to-receive-millions-from-ericsson-over-software-failure

[69] S. Gallagher, "The Army's Costly Quest for the Perfect Radio Continues," *Ars Technica*, Mar 2018. [Online]. Available: https://arstechnica.com/information-technology/2018/03/the-armys-costly-quest-for-the-perfect-radio-continues/

[70] "Joint Tactical Radio System," Mar 2019, accessed: Nov. 10, 2020. [Online]. Available: https://en.wikipedia.org/wiki/Joint_Tactical_Radio_System

[71] V. J. Kovarik and R. Muralidharan, "Model-Based Systems Engineering: Lessons Learned from the Joint Tactical Radio System," *Journal of Signal Processing Systems*, vol. 89, no. 1, pp. 97–106, Oct 2017. [Online]. Available: https://doi.org/10.1007/s11265-016-1218-2

[72] University Communications, "NSF Announces Raleigh and Cary as Testbed Sites for Advanced Wireless," *NC State News*, Sep 2019. [Online]. Available: https://news.ncsu.edu/2019/09/mobile-5g-wireless-networks/

[73] "Ottawa Firms Central to New $400M Public-Private 5G Partnership," *Ottawa Business Journal*. [Online]. Available: https://obj.ca/article/techopia-ottawa-firms-central-new-400m-public-private-5g-partnership

[74] D. Grundke, "The Gap in Wireless Communications: Ultra- Reliability and Low Latency," 2018. [Online]. Available: https://e2e.ti.com/blogs_/b/industrial_strength/archive/2018/06/12/the-gap-in-wireless-communications-ultra-reliability-and-low-latency

[75] T. Basmer, H. Schomann, and S. Peter, "Implementation Analysis of the IEEE 802.15.4 MAC for Wireless Sensor Networks," in *2011 International Conference on Selected Topics in Mobile and Wireless Networking (iCOST)*, 2011, pp. 7–12.

[76] M. Danneberg, R. Bomfin, S. Ehsanfar, A. Nimr, Z. Lin, M. Chafii, and G. Fettweis, "Online Wireless Lab Testbed," in *2019 IEEE Wireless Communications and Networking Conference Workshop (WCNCW)*, 2019, pp. 1–5.

[77] "mmWave Access and Backhaul Link Tests and Presentation of Final Demonstrator," MiWaveS project consortium, Tech. Rep., 2017.

[78] J. Vieira, S. Malkowsky, K. Nieman, Z. Miers, N. Kundargi, L. Liu, I. Wong, V. Öwall, O. Edfors, and F. Tufvesson, "A Flexible 100-Antenna Testbed for Massive MIMO," 2014, iEEE Globecom Workshop, 2014 : - Massive MIMO: From Theory to Practice ; Conference date: 08-12-2014 Through 08-12-2014.

[79] "Setting World Records in 5G Wireless Spectral Efficiency," National Instruments, Whitepaper. [Online]. Available: https://www.ni.com/en-rs/innovations/case-studies/19/setting-world-records-in-5g-wireless-spectral-efficiency.html

[80] M. Rosker, "Spectrum Collaboration Challenge (SC2)," *Defense Advanced Research Projects Agency*. [Online]. Available: https://www.darpa.mil/program/spectrum-collaboration-challenge

[81] The Electronic Resurgence Initiative, "The Colosseum RF Testbed," 2018, ERI Summit. [Online]. Available: https://eri-summit.darpa.mil/docs/ERIPoster_Applications_SC2_Colosseum.pdf

[82] DARPA, "World's Most Powerful RF Emulator to Become National Wireless Research Asset," Mar 2019. [Online]. Available: https://www.darpa.mil/news-events/2019-09-03

[83] "DARPA Announces Transfer of SC2 Colosseum Hardware to PAWR Program," Nov 2019. [Online]. Available: https://advancedwireless.org/darpa-announces-transfer-sc2-colosseum-hardware-pawr-program/

[84] N. Michailow and M. Danneberg, *LTE Advanced Testbed Documentation*, CREW Project consortium. [Online]. Available: http://www.crew-project.eu/book/export/html/50

[85] M. Grieger, M. Danneberg, G. Fettweis, M. Amro, A. Landolsi, S. A. Zummo, and J. Voigt, "Uplink Coordinated Multi-Point in field trials and ray tracing simulations," in *The 8th European Conference on Antennas and Propagation (EuCAP 2014)*, 2014, pp. 1775–1779.

[86] V. Handziski, A. Köpke, A. Willig, and A. Wolisz, "Twist: A Scalable and Reconfigurable Testbed for Wireless Indoor Experiments with Sensor Networks," *Proceedings of the second international workshop on Multi-hop ad hoc networks: from theory to reality - REALMAN 06*, May 2006. [Online]. Available: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.67.8788&rep=rep1&type=pdf

[87] *System & Network Architecture*, IMEC. [Online]. Available: https://doc.ilabt.imec.be/ilabt/wilab/architecture.html#w-ilab-2-architecture

[88] *Portable Testbed - IMEC iLab.t Documentation*, IMEC. [Online]. Available: https://doc.ilabt.imec.be/ilabt/portabletestbed/index.html

[89] IMEC iLab.t, *Virtual Wall Testbed*. [Online]. Available: https://doc.ilabt.imec.be/ilabt-documentation/

[90] Mellanox, "Mellanox Innova™-2 Flex Open Programmable SmartNIC." [Online]. Available: https://www.mellanox.com/products/smartnics/innova-2-flex

[91] *PlanetLab: An Open Platform for Developing, Deploying, and Accessing Planetary-Scale Services*, The Trustees of Princeton University. [Online]. Available: https://www.planet-lab.org/

[92] L. Peterson, "It's Been a Fun Ride," *Systems Approach*, Mar 2020. [Online]. Available: https://www.systemsapproach.org/blog/its-been-a-fun-ride

[93] OneLab project consortium, "Onelab." [Online]. Available: https://onelab.eu/

[94] R. Dipankar *et al.*, "Challenge: COSMOS: A City-Scale Programmable Testbed for Experimentation with Advanced Wireless," in *The 26th Annual International Conference on Mobile Computing and Networking (MobiCom '20)*, 09 2020, p. 13.

[95] COSMOS project consortium, "Cloud Enhanced Open Software Defined Mobile Wireless Testbed for City-Scale Deployment." [Online]. Available: https://cosmos-lab.org/

[96] Connectcentre.ie, *Iris - The Reconfigurable Radio Testbed*. [Online]. Available: https://iris-testbed.connectcentre.ie/

[97] M. Kist, J. Rochol, L. A. DaSilva, and C. B. Both, "HyDRA: A Hypervisor for Software Defined Radios to Enable Radio Virtualization in Mobile Networks," in *2017 IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, May 2017, pp. 960–961.

[98] S. Caban, J. A. Garcia Naya, and M. Rupp, "Measuring the Physical Layer Performance of Wireless Communication Systems: Part 33 in a Series of Tutorials on Instrumentation and Measurement," *IEEE Instrumentation Measurement Magazine*, vol. 14, no. 5, pp. 8–17, 2011.

[99] European Commission, *Technology Readiness Levels (TRL)*, 2014. [Online]. Available: https://ec.europa.eu/research/participants/data/ref/h2020/wp/2014_2015/annexes/h2020-wp1415-annex-g-trl_en.pdf

[100] R. Verdone and A. Zanella, *Pervasive Mobile and Ambient Wireless Communications COST Action 2100*. Springer London, 2012.

[101] *Technisches Systemhandbuch: Luftkühlung*, RITTAL. [Online]. Available: https://www.rittal.com/imf/none/5_3955/Rittal_Technisches_Systemhandbuch_Luftk%C3%BChlung_5_3955/

[102] R. Nissel, M. Lerch, and M. Rupp, "Experimental Validation of the OFDM Bit Error Probability for a Moving Receive Antenna," in *2014 IEEE 80th Vehicular Technology Conference (VTC2014-Fall)*, Sep. 2014, pp. 1–5.

[103] Ettus Research, "USRP E310 Embedded Software Defined Radio (SDR)." [Online]. Available: https://www.ettus.com/all-products/e310/

[104] National Instruments, "mmWave Transceiver System." [Online]. Available: http://sine.ni.com/nips/cds/view/p/lang/de/nid/213722

[105] X. Wang, M. Laabs, D. Plettermeier, K. Kosaka, and Y. Matsunaga, "MIMO Antenna Array System with Integrated 16x16 Butler Matrix and Power Amplifiers for 28GHz Wireless Communication," in *2019 12th German Microwave Conference (GeMiC)*, March 2019, pp. 127–130.

[106] ThinkRF, "D4000 RF Downconverter." [Online]. Available: https://www.thinkrf.com/products/rf-downconverters/d4000-rf-downconverter/

[107] T. Halsig, D. Cvetkovski, E. Grass, and B. Lankl, "Measurement Results for Millimeter Wave Pure LOS MIMO Channels," in *2017 IEEE Wireless Communications and Networking Conference (WCNC)*, 2017, pp. 1–6.

[108] "Introduction to the NI MIMO Prototyping System Hardware," National Instruments, Whitepaper. [Online].

Available: https://www.ni.com/de-de/innovations/white-papers/16/introduction-to-the-ni-mimo-prototyping-system-hardware.html

[109] C. Mehlführer, S. Caban, and M. Rupp, "Experimental Evaluation of Adaptive Modulation and Coding in MIMO WiMAX with Limited Feedback," *EURASIP Journal on Advances in Signal Processing*, vol. 2008, no. 1, Dec. 2007. [Online]. Available: https://doi.org/10.1155/2008/837102

[110] IMEC iLab.t, *w-iLab.t Testbed*. [Online]. Available: https://doc.ilabt.imec.be/ilabt-documentation/wilabfacility.html

[111] Bundesnetzagentur, "Versuchsfunk." [Online]. Available: https://www.bundesnetzagentur.de/DE/Sachgebiete/Telekommunikation/Unternehmen_Institutionen/Frequenzen/SpezielleAnwendungen/Versuchsfunk/versuchsfunk-node.html

[112] FCC Office of Engineering and Technology, *Program Experimental License*, FCC. [Online]. Available: https://apps.fcc.gov/els/ProgramExpLicensePurposeOption.do

[113] B. Vermeulen *et al.*, "D2.2: Federation Operations, Tools and Support," Fd4Fire project consortium, Tech. Rep. [Online]. Available: https://fed4fire.eu/wp-content/uploads/sites/10/2019/07/d2.02.pdf

[114] *HARQ Timing*, Keysight Technologies. [Online]. Available: http://rfmw.em.keysight.com/wireless/helpfiles/n7624b/Content/RT/harq_level_mode.htm

[115] M. S. Gast, *802.11 Wireless Networks: the Definitive Guide*. OReilly Media, 2017.

[116] "Short Interframe Space," May 2019, accessed: Nov. 10, 2020. [Online]. Available: https://en.wikipedia.org/wiki/Short_Interframe_Space

[117] L. Ariza and T. Laurent, *Setup for Real Time Performance*, OpenAirInterface, Dec 2016. [Online]. Available: https://gitlab.eurecom.fr/oai/openairinterface5g/-/wikis/setup-for-real-time-performance

[118] MSCSP Laboratories, "MIMO RUSK Channel Sounder," TU Ilmenau. [Online]. Available: https://www.tu-ilmenau.de/en/ei-ms-csp/mscsp-labs/mimo-rusk-channel-sounder/

[119] *IEEE® 802.11™ WLAN - OFDM Beacon Receiver with USRP® Hardware*, Matlab. [Online]. Available: https://de.mathworks.com/help/supportpkg/usrpradio/examples/ieee-802-11-tm-wlan-ofdm-beacon-receiver-with-usrp-r-hardware.html

[120] "srsLTE - Your own Mobile Network." [Online]. Available: https://www.srslte.com/

[121] Ettus Research, "USRP B205mini-i." [Online]. Available: https://www.ettus.com/all-products/usrp-b205mini-i/

[122] "RTL-SDR Blog." [Online]. Available: http://www.rtl-sdr.com

[123] G. Ghiaasi, M. Ashury, D. Vlastaras, M. Hofer, Zhinan Xu, and T. Zemen, "Real-time Vehicular Channel Emulator for Future Conformance Tests of Wireless ITS Modems," in *2016 10th European Conference on Antennas and Propagation (EuCAP)*, April 2016, pp. 1–5.

[124] X. Jiao, I. Moerman, W. Liu, and F. A. P. de Figueiredo, "Radio Hardware Virtualization for Coping with Dynamic Heterogeneous Wireless Environments," in *Cognitive Radio Oriented Wireless Networks*, P. Marques, A. Radwan, S. Mumtaz, D. Noguet, J. Rodriguez, and M. Gundlach, Eds. Cham: Springer International Publishing, 2018, pp. 287–297.

[125] "Application-Specific Integrated Circuit," Jun 2020, accessed: Nov. 10, 2020. [Online]. Available: https://en.wikipedia.org/wiki/Application-specific_integrated_circuit

[126] O. Arnold, "MPSoC Design: The Tomahawk Approach," Nov 2014.

[127] M. Castro, F. Dupros, E. Francesquini, J. Méhautk, and P. O. A. Navaux, "Energy Efficient Seismic Wave Propagation Simulation on a Low-Power Manycore Processor," in *2014 IEEE 26th International Symposium on Computer Architecture and High Performance Computing*, 2014, pp. 57–64.

[128] "End-to-end Demonstration and Evaluation of Showcases," eWINE Project Consortium, Tech. Rep., May 2018. [Online]. Available: https://ewine-project.media.martel-innovate.com/wp-content/uploads/sites/29/2018/05/eWINE_D2.2_Final.pdf

[129] I. Seskar, *About OMF*, Rutgers University. [Online]. Available: https://omf.orbit-lab.org/wiki/WikiStart

[130] T. Betz, "Netzwerkexperimente mit NEPI." Technische Universität München. [Online]. Available: https://www.net.in.tum.de/fileadmin/TUM/NET/NET-2014-08-1/NET-2014-08-1_02.pdf

[131] IMEC, "jFed is a Java-based Framework for Testbed Federation," *jFed*. [Online]. Available: https://jfed.ilabt.imec.be/

[132] P. Gallo *et al.*, *WiSHFUL UPI Packages*, WiSHFUL Consortium, 2016. [Online]. Available: https://wishful-project.github.io/wishful_upis/wishful_upis.html

[133] *GNU Radio*. [Online]. Available: https://wiki.gnuradio.org/index.php/Main_Page

[134] J. Travis and J. Kring, *LabVIEW for Everyone - Graphical Programming Made Easy and Fun*. London: Prentice Hall, 2007.

[135] D. Hristu-Varsakelis and W. S. Levine, Eds., *Handbook of Networked and Embedded Control Systems*. Birkhäuser Boston, 2005. [Online]. Available: https://doi.org/10.1007/b137198

[136] "MyHDL." [Online]. Available: http://www.myhdl.org/

[137] "High-Level Synthesis," Apr 2020, accessed: Nov. 10, 2020. [Online]. Available: https://en.wikipedia.org/wiki/High-level_synthesis

[138] Z. Li, A. Nimr, and G. Fettweis, "Implementation and Performance Measurement of Flexible Radix-2 GFDM Modem," in *IEEE 5G World Forum (WF-5G)*, Dresden, Germany, Sep 2019.

[139] M. Danneberg and Z. Lin, *Online Wireless Lab*, TU Dresden. [Online]. Available: http://owl.ifn.et.tu-dresden.de/

[140] "Altneubau," Apr 2018, accessed: Nov. 10, 2020. [Online]. Available: https://de.wikipedia.org/wiki/Altneubau

[141] K. Rathi, "An Interview With the Inventor of USRP: Happy 10th Birthday," Jan 2015. [Online]. Available: https://www.ettus.com/an-interview-with-the-inventor-of-usrp-on-the-10th-birthday/

[142] *General Data Protection Regulation (GDPR) Compliance Guidelines*, Proton Technologies AG. [Online]. Available: https://gdpr.eu/

[143] Paciello Group, "EU Directive on the Accessibility of Public Sector Websites and Mobile Applications," Apr 2018. [Online]. Available: https://developer.paciellogroup.com/blog/2018/04/eu-directive-on-the-accessibility-of-public-sector-websites-and-mobile-applications/

[144] National Instruments, "LTE Application Framework." [Online]. Available: http://sine.ni.com/nips/cds/view/p/lang/de/nid/213083

[145] ——, "802.11 Application Framework." [Online]. Available: http://sine.ni.com/nips/cds/view/p/lang/de/nid/213084

[146] ——, "LabVIEW Communications System Design Suite." [Online]. Available: https://www.ni.com/de-de/shop/wireless-design-test/software-suites-for-wireless-design-and-test-category/what-is-labview-communications.html

[147] Ettus Research, "USRP X310 High Performance Software Defined Radio." [Online]. Available: https://www.ettus.com/all-products/x310-kit/

[148] M. Danneberg *et al.*, "Flexible GFDM Implementation in FPGA with Support to Run-Time Reconfiguration," in *2015 IEEE 82nd Vehicular Technology Conference (VTC2015-Fall)*, Sept 2015, pp. 1–2.

[149] M. Danneberg, N. Michailow, I. Gaspar, M. Matthé, Dan Zhang, L. L. Mendes, and G. Fettweis, "Implementation of a 2 by 2 MIMO-GFDM transceiver for Robust 5G Networks," in *2015 International Symposium on Wireless Communication Systems (ISWCS)*, 2015, pp. 236–240.

[150] M. Danneberg, R. Bomfin, A. Nimr, Z. Li, and G. Fettweis, "USRP-Based Platform for 26/28 GHz mmWave Experimentation," in *2020 IEEE Wireless Communications and Networking Conference Workshops (WCNCW)*, 2020, pp. 1–6.

[151] M. Matthé, M. Danneberg, D. Zhang, and G. Fettweis, *Flexible Physical Layer Design*. Cambridge University Press, 2017, p. 133–150.

[152] J. Bonér, D. Farley, R. Kuhn, and M. Thompson, "Glossary - The Reactive Manifesto," *The Reactive Manifesto*, 2014. [Online]. Available: https://www.reactivemanifesto.org/glossary#Back-Pressure

[153] I. Gaspar, A. Navarro, N. Michailow, and G. Fettweis, "FPGA Implementation of Generalized Frequency Division Multiplexing Transmitter using NI LabVIEW and NI PXI Platform," 10 2013.

[154] M. Matthé, L. L. Mendes, I. Gaspar, N. Michailow, D. Zhang, and G. Fettweis, "Precoded GFDM Transceiver with Low Complexity Time Domain Processing," *EURASIP Journal on Wireless Communications and Networking*, p. 138, 2016.

[155] T. M. Schmidl and D. C. Cox, "Robust Frequency and Timing Synchronization for OFDM," *IEEE Transactions on Communications*, vol. 45, no. 12, pp. 1613–1621, 1997.

[156] T. F. Collins, R. Getz, D. Pu, and A. M. Wyglinski, *Software-defined radio for engineers*. Artech House, 2018.

[157] L. Song and J. Shen, *Evolved Cellular Network Planning and Optimization for UMTS and LTE*. CRC Press, 2010.

[158] H. Minn, M. Zeng, and V. K. Bhargava, "On Timing Offset Estimation for OFDM Systems," *IEEE Communications Letters*, vol. 4, no. 7, pp. 242–244, 2000.

[159] A. B. Awoseyila, C. Kasparis, and B. G. Evans, "Improved Preamble-Aided Timing Estimation for OFDM Systems," *IEEE Communications Letters*, vol. 12, no. 11, pp. 825–827, November 2008.

[160] J. Abdoli, M. Jia, and J. Ma, "Filtered OFDM: A New Waveform for Future Wireless Systems," *IEEE Workshop on Signal Processing Advances in Wireless Communications, SPAWC*, vol. 2015-Augus, pp. 66–70, 2015.

[161] N. Michailow *et al.*, "Generalized Frequency Division Multiplexing: A Flexible Multi-Carrier Modulation Scheme for 5th Generation Cellular Networks," in *Proceedings of the German microwave conference (GeMiC 12)*, Ilmenau, Germany, 2012, pp. 1–4.

[162] A. Viholainen, T. Ihalainen, T. H. Stitz, M. Renfors, and M. Bellanger, "Prototype Filter Design for Filter Bank Based Multicarrier Transmission," in *Proceedings of European Signal Processing Conference (EUSIPCO)*, 2009. [Online]. Available: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.184.6165

[163] S. H. Chung and P. J. Mclane, "Code Hopping - Direct Sequence Spread Spectrum to Compensate for Intersymbol Interference in an Ultra-Wideband System," *IEEE Transactions on Communications*, vol. 56, no. 11, pp. 1785–1789, November 2008.

[164] X. Ouyang and J. Zhao, "Orthogonal Chirp Division Multiplexing," *IEEE Transactions on Communications*, vol. 64, no. 9, pp. 3946–3957, Sept 2016.

[165] M. Danneberg, A. Nimr, M. Matthé, and G. Fettweis, "Network Slicing for Industry 4.0 Applications - Initial RAN Testbed Results," January 2018, IEEE Softwarization.

[166] *Fast Fourier Transform*, 9th ed., XILINX, Oct 2017. [Online]. Available: https://www.xilinx.com/support/documentation/ip_documentation/xfft/v9_0/pg109-xfft.pdf

[167] *7 Series FPGAs Configurable Logic Block*, XILINX. [Online]. Available: https://www.xilinx.com/support/documentation/user_guides/ug474_7Series_CLB.pdf

[168] S. M. Alamouti, "A Simple Transmit Diversity Technique for Wireless Communications," *IEEE Journal on Selected Areas in Communications*, vol. 16, no. 8, pp. 1451–1458, 1998.

[169] M. Matthe, L. L. Mendes, I. Gaspar, N. Michailow, D. Zhang, and G. Fettweis, "Multi-User Time-Reversal STC-GFDMA for Future Wireless Networks," *EURASIP Journal on Wireless Communications and Networking*, vol. 2015, no. 1, p. 132, May 2015. [Online]. Available: https://doi.org/10.1186/s13638-015-0366-6

[170] V. Bose, "A Software Driven Approach to SDR Design," *COTS Journal*, Jan 2004. [Online]. Available: http://www.cotsjournalonline.com/articles/view/100056

[171] FAST project consortium, "fast - Fast Actuators Sensors & Transceivers." [Online]. Available: https://de.fast-zwanzig20.de/

[172] "Campus Navigator TU Dresden." [Online]. Available: https://navigator.tu-dresden.de/etplan/bar/02

[173] National Center for High Performance Computing, "Clonezilla." [Online]. Available: https://clonezilla.org/

[174] J. White, G. Jourjon, T. Rakatoarivelo, and M. Ott, "Measurement Architectures for Network Experiments with Disconnected Mobile Nodes," in *Testbeds and Research Infrastructures. Development of Networks and Communities*, T. Magedanz, A. Gavras, N. H. Thanh, and J. S. Chase, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, pp. 315–330.

[175] M. Jacobs and M. Satran, "Interprocess Communications - Win32 Apps," May 2018. [Online]. Available: https://docs.microsoft.com/de-de/windows/win32/ipc/interprocess-communications?redirectedfrom=MSDN

[176] F. Rademacher, J. Sorgalla, and S. Sachweh, "Challenges of Domain-Driven Microservice Design: A Model-Driven Perspective," *IEEE Software*, vol. 35, no. 3, pp. 36–43, May 2018.

[177] T. Funkhouser, "Inter-Process Communication," Introduction to Programming Systems - Lecture Notes. [Online]. Available: https://www.cs.princeton.edu/courses/archive/spring03/cs217/lectures/Communication.pdf

[178] J. Lauener and W. Sliwinski, "How to Design & Implement a Modern Communication Middleware Based on ZeroMQ," Oct 2017.

[179] T. Fountain, *Converting an X310 into an NI-USRP RIO*, National Instruments, 2016. [Online]. Available: https://kb.ettus.com/Converting_an_X310_into_an_NI-USRP_Rio

[180] National Instruments, "USRP Hardware Driver Software." [Online]. Available: https://github.com/EttusResearch/uhd

[181] B. Clerckx and C. Oestges, "Chapter 14 - MIMO in LTE, LTE-Advanced and WiMAX," in *Mimo Wireless Networks (Second Edition)*, second edition ed., B. Clerckx and C. Oestges, Eds. Oxford: Academic Press, 2013, pp. 597 – 635. [Online]. Available: http://www.sciencedirect.com/science/article/pii/B9780123850553000146

[182] X. Wang, M. Laabs, D. Plettemeier, K. Kosaka, and Y. Matsunaga, "28 GHz Multi-Beam Antenna Array based on Wideband High-dimension 16x16 Butler Matrix," in *2019 13th European Conference on Antennas and Propagation (EuCAP)*, March 2019, pp. 1–4.

[183] M. Sigmund, "Real-Time Channel Sounding Implementation," 2020.

[184] M. Danneberg, Z. Li, P. Kühne, A. Nimr, S. Ehsanfar, M. Chafii, and G. Fettweis, "Real-Time Waveform Prototyping," in *2019 IEEE 20th International Workshop on Signal Processing Advances in Wireless Communications (SPAWC)*, July 2019, pp. 1–5.

[185] S. Bradner and J. McQuaid, "Benchmarking Methodology for Network Interconnect Devices," Mar 1999. [Online]. Available: https://www.ietf.org/rfc/rfc2544.txt

[186] "Y.156sam," Jun 2019, accessed: Nov. 10, 2020. [Online]. Available: https://en.wikipedia.org/wiki/Y.156sam

[187] P. Emmerich, S. Gallenmüller, D. Raumer, F. Wohlfart, and G. Carle, "MoonGen: A Scriptable High-Speed Packet Generator," in *Internet Measurement Conference 2015 (IMC'15)*, Tokyo, Japan, Oct. 2015.

[188] M. Majkowski, "How to Receive a Million Packets per Second," *The Cloudflare Blog*, Jun 2019. [Online]. Available: https://blog.cloudflare.com/how-to-receive-a-million-packets/

[189] A. Anisimov, "LTE Throughput Calculator." [Online]. Available: http://anisimoff.org/eng/lte_throughput_calculator.html

[190] Barkhausen Institut, "Roboter-airhockey." [Online]. Available: https://www.barkhauseninstitut.org/en/research/lab/low-latency-robotic-airhockey-1

[191] National Instruments, "HPE Edgeline EL1000: Converged IoT PXI Chassis." [Online]. Available: https://www.ni.com/de-de/support/model.hpe-edgeline-el1000.html

[192] W. Nitzold, "Prototyping LTE-WiFi Interworking on a Single SDR Platform," GNU Radio Conference 2019.

[193] "Doxygen," Apr 2020, accessed: Nov. 10, 2020. [Online]. Available: https://en.wikipedia.org/wiki/Doxygen

[194] *USRP Hardware Driver and USRP Manual: FPGA Manual*, Ettus Research. [Online]. Available: https://files.ettus.com/manual/md_fpga.html

[195] G. M. Ung, "Threadripper 3990X Review Roundup: AMD's 64-core CPU Can Play Crysis, But It's Not for Everyone," *PCWorld*, Feb 2020. [Online]. Available: https://www.pcworld.com/article/3520994/threadripper-3990x-review-roundup-amds-64-core-cpu-can-play-crysis-but-its-not-for-everyone.html

[196] Epiq Solutions, "Sidekiq™." [Online]. Available: https://www.epiqsolutions.com/rf-transceiver/sidekiq/

[197] "Larrabee (Microarchitecture)," Feb 2020, accessed: Nov. 10, 2020. [Online]. Available: https://en.wikipedia.org/wiki/Larrabee_(microarchitecture)

[198] E. Grayver and A. Utter, "Extreme Software Defined Radio – GHz in Real Time," in *2020 IEEE Aerospace Conference*, 2020, pp. 1–10.

[199] "Study on New Radio Access Technology: Radio Access Architecture and Interfaces (Release 14)," Mar 2017. [Online]. Available: https://portal.3gpp.org/desktopmodules/Specifications/SpecificationDetails.aspx?specificationId=3056

[200] G. Urs, *Cyclictest*, Interstaatliche Hochschule für Technik Buchs NTB, Oct 2019. [Online]. Available: https://wiki.ntb.ch/infoportal/software/linux/cyclictest/start

[201] Microwaves101, "Unstable Amplifier Examples." [Online]. Available: https://www.microwaves101.com/encyclopedias/unstable-amplifier-examples

[202] M. Danneberg, "Entwurf und Realisierung einer Backplane für ein Phased-Array Antennensystem."

[203] CRAWDAD, "A Community Resource for Archiving Wireless Data At Dartmouth." [Online]. Available: https://crawdad.org/

[204] ImageNet, "About ImageNet." [Online]. Available: http://image-net.org/about-overview

[205] *FPGAs for Wireless Engineers Series: Go from a Concept to FPGA Code with No HDL Expertise*, National Instruments, 2018. [Online]. Available: http://www.ni.com/tutorial/54245/en/

[206] C. Puppe, "Container - Orchestrierung im Wandel," *iX-Extra*, p. 118–119, Jun 2020.

[207] A. Plummer and K. Taylor, "Development and Operations on the Defense Advanced Research Project Agency's Spectrum Collaboration Challenge," *Johns Hopkins APL Technical Digest*, vol. 35. [Online]. Available: https://www.jhuapl.edu/Content/techdigest/pdf/V35-N01/35-01-Plummer.pdf

[208] *PYNQ Introduction - Python Productivity for Zynq (Pynq)*, XILINX, 2018. [Online]. Available: https://pynq.readthedocs.io/en/v2.5.1/

[209] Q. Li, G. Wu, A. Papathanassiou, and U. Mukherjee, "Radio Slicing," December 2017, IEEE Softwarization.

[210] D. Chu, "Polyphase Codes with Good Periodic Correlation Properties," *IEEE Trans. Inf. Theory*, vol. 18, no. 4, pp. 531–532, July 1972.

[211] S. Boumard and A. Mammela, "Robust and Accurate Frequency and Timing Synchronization Using Chirp Signals," *IEEE Trans. on Broad.*, vol. 55, no. 1, pp. 115–123, March 2009.