Embedded Massive MTC Device Emulator for LTE using Software Defined Radios

Master Thesis, Autumn 2016 – Spring 2017

Anders Charly Rasmussen and Mathias Rønholt Kielgast
Wireless Communication Systems
Department of Electronic Systems

"Wire telegraph is a kind of a very, very long cat.
You pull its tail in New York and his head is meowing in Los Angeles.
And radio operates exactly the same way:
you send signals here, they receive them there.
The only difference is that there is no cat."

AALBORG UNIVERSITY
STUDENT REPORT

Antennas, Propagation and Radio Networking (APNet)
Department of Electronic Systems
Aalborg University
Fredrik Bajers Vej 7
DK-9220 Aalborg Ø, Denmark
Telephone: (+45) 9940 8600
Fax: 9940 9840
www.es.aau.dk

Field of education: Wireless Communication Systems
Title: Embedded Massive MTC Device Emulator for LTE using Software Defined Radios
Project period: Autumn 2016 – Spring 2017

Authors:

Anders Charly Rasmussen
arasm12@student.aau.dk

Mathias Rønholt Kielgast
mkielg11@student.aau.dk

Abstract:

An increase in machine type communication (MTC) devices on LTE networks, generally for uses within the internet of things (IoT), is occurring and is expected to continue with much increased growth. These new types of devices results in new communication patterns, which together with the massiveness of the expected amount of devices, initiated changes in LTE. A device emulating multiple devices on the network is sought in order to enable performance testing of the LTE protocol with regards to massive amounts of MTC devices.
A prototype using LTE Release 8 compliant code is developed, capable of running several devices in parallel and transmitting their combined signals through a common radio front-end. This functionality is achieved by splitting the physical layer of the LTE protocol into a common part for all devices, and an individual part tied to to each device. Tests verify the design principle, showing that connecting multiple MTC devices using the same radio front-end to an eNodeB is possible. The prototype developed enables up to 15 devices to synchronise and complete random access procedure with a commercial eNodeB at 5 MHz. Further tests show that bandwidth has great importance in the number of supported devices. This promises well for implementations using MTC specific protocols which operate at a much lower bandwidth, such as LTE-M and NB-IoT. The implementation can therefore be seen as a strong proof-of-concept for massive device emulators for LTE based networks.

Supervisors:
Petar Popovski
Dong Min Kim

Industrial supervisor:
Germán Corrales Madueño

Printed copies: 5
Page count: 74
Appendices: 25
Completed: June 7th, 2017

Preface

This Master's thesis is covering the work done by Anders Charly Rasmussen and Mathias Rønholt Kielgast, concluding their Wireless Communication Systems Master's programme at Aalborg University. The thesis covers the development of a "massiveMTC emulator" device, which is performed in a collaboration between Keysight Laboratories in Aalborg and the Department of Electronic Systems at Aalborg University.

The authors would like to thank Telenor Aalborg for granting access to eNodeB and core network test facilities, with special thanks to Simone Barbera of Telenor for his helpfulness. The authors would also like to thank Keysight Laboratories for the extensive access to their UXM wireless test setup. In addition, thanks is given to Nuno Kiilerich Pratas for guidance during the first parts of the thesis.

The thesis begins with an introduction giving an overview of the context of the research and current LTE development, before describing the objectives of the thesis. Following the introduction, preliminary research will be described to give the required insight and understanding of the technical aspects relevant for the development. Subsequently, a description of the test bed setup is given before the documentation of the prototype device development. The prototype functionality and performance is then evaluated, before the thesis is rounded off by conclusions and an overview of further work.

Citations are made by the use of the American Institute of Physics (AIP) style. This uses the citations numbered as e.g. [3], the full cite is then written in the bibliography in the end of the thesis. If a citation is placed before a full stop, it refers only to that sentence; when placed after a full stop, it refers to the entire paragraph. References for figures, tables and equation, are made according to the chapter name in which they are found, e.g. the second figure in chapter X would be referred as figure X.2. All graphs and pictures are referred to as figures.

Aalborg University, June 7th, 2017

Contents

Nomenclature

3GPP	3rd generation partnership project
ACK	Acknowledgement
ADC	Analogue-to-digital converter
AES	Advanced encryption standard
AIA	Authentication information answer
AIR	Authentication information request
AM	Acknowledged mode
AMC	Adaptive modulation and coding
ARP	Access reservation protocol
AS	Access stratum
ASME	Access security management entity
BCCH	Broadcast control channel
BCH	Broadcast channel
BLER	Block error rate
BSR	Buffer status reporting
C-RNTI	Cell radio network temporary identifier
CA	Carrier aggregation
CC	Country code
CCCH	Common control channel
CFO	Carrier frequency offset
CN	Core network
coPHY	Common physical layer
CP	Cyclic prefix
CQI	Channel quality indicator
CSI	Channel status information
DAC	Digital-to-analogue converter
DCCH	Dedicated control channel
DCI	Downlink control information
DDC	Digital down-converter
DEMUX	Demultiplexing
DFT	Discrete Fourier transform
DFT-S-OFDM	DFT-spread-OFDM
DL-SCH	Downlink shared channel
DRB	Data radio bearer
DRX	Discontinuous reception
DSP	Digital signal processor
DTCH	Dedicated traffic channel
DUC	Digital up-converter
E-UTRAN	Evolved UMTS terrestrial radio access network

eDRX	Extended discontinuous reception
EEA	EPS encryption algorithm
EIA	EPS integrity algorithm
eMTC	Enhanced MTC
eNodeB	Evolved node B
EPC	Evolved packet core
EPRE	Energy per RE
EPS	Evolved packet system
FDD	Frequency-division duplexing
FDMA	Frequency-division multiple access
FPGA	Field-programmable gate array
GPP	General purpose processor
GPRS	General packet radio service
GSM	Global system for mobile communications
GUI	Graphical user interface
HARQ	Hybrid automatic repeat request
HeNB	Home eNodeB
HSS	Home subscriber service
ICI	Inter-carrier interference
IF	Intermediate frequency
IMEI	International mobile equipment identity
IMSI	International mobile subscriber identity
inPHY	Individual physical layer
IoT	Internet of things
IP	Internet protocol
ISI	Inter symbol interference
LP-OFDMA	Linearly precoded OFDMA
LPWAN	Low-power wide-area network
LSB	Least significant bit
LTE	Long term evolution
LTE-A	Long term evolution advanced
LTE-M	LTE-MTC
M2M	Machine-to-machine communication
MAC	Multiple access control
MCC	Mobile country code
MCCH	Multicast control channel
MCH	Multicast channel
MCS	Modulation and coding scheme
MIB	Master information block
MIMO	Multiple input multiple output
MME	Mobility management entity
MNC	Mobile network code
MSG	Message
MSIN	Mobile subscriber identification number
MSINDN	Mobile station international subscriber directory number
MTC	Machine type communication

MTCH	Multicast traffic channel
MUX	Multiplexing
NACK	Negative acknowledgement
NAS	Non-access stratum
NB-IoT	Narrowband IoT
NDC	National destination code
OAI	OpenAirInterface
OFDM	Orthogonal frequency-division multiplexing
OFDMA	Orthogonal frequency-division multiple access
OP	Operator key
OS	Operating system
OSI	Open systems interconnection
OTA	Over-the-air
P-GW	PDN gateway
PAPR	Peak-to-average power ratio
PBCH	Physical broadcast channel
PCAP	Packet capture
PCCH	Paging control channel
PCFICH	Physical control format indicator channel
PCH	Paging channel
PCRF	Policy control and charging rules function
PDCCH	Physical downlink control channel
PDCP	Packet data convergence protocol
PDN	Packet data network
PDSCH	Physical downlink shared channel
PDU	Protocol data unit
PGA	Programmable gain amplifier
PHICH	Physical HARQ indicator channel
PHR	Power headroom reporting
PHY	Physical (layer)
PLMN	Public land mobile network
PMCH	Physical multicast channel
PPRB	Power per RB
PRACH	Physical random access channel
PRB	Physical resource block
PSM	Power saving mode
PSS	Primary synchronisation signal
PUCCH	Physical uplink control channel
PUSCH	Physical uplink shared channel
QoS	Quality of service
QPSK	Quadrature phase-shift keying
RA	Random access
RACH	Random access channel
RAO	Random access opportunity
RAP	Random access procedure
RAR	Random access response

RB	Resource block
RE	Resource element
RF	Radio frequency
RLC	Resource link control
RMSE	Root mean square error
RRC	Radio resource control
RS	Reference signal
RSRP	RS received power
RSSI	Received signal strength indicator
S-GW	Serving gateway
SAE	System architecture evolution
SC-FDMA	Single carrier FDMA
SDR	Software defined radio
SDU	Service data unit
SFN	Subframe number
SFO	Sampling frequency offset
SI	System information
SIB	System information block
SISO	Single input single output
SN	Subscriber number
SNR	Signal-to-noise ratio
SR	Scheduling request
SRB	Signalling radio bearer
SRS	Sounding reference signals
SSS	Secondary synchronisation signal
TB	Transport block
TBS	Transport block size
TCP	Transmission control protocol
TDD	Time-division duplexing
TDMA	Time-division multiple access
TM	Transparent mode
TPC	Transmit power control
TTI	Transmission time interval
UCI	Uplink control information
UDP	User datagram protocol
UE	User equipment
UHD	USRP hardware driver
ULSCH	Uplink shared channel
UM	Unacknowledged mode
UMTS	Universal mobile telecommunications system
USIM	Universal subscriber identity module
USRP	Universal software radio peripheral
VNI	Virtual network interface
VoIP	Voice over IP
VPN	Virtual private network

List of Symbols

A	Amplitude
A_I	Amplitude of I component
A_Q	Amplitude of Q component
B	Bandwidth
f_c	Carrier frequency
f_s	Sampling frequency
G_{tx}	Transmission gain
I	In-phase (component)
K	Secret key
κ	Digital scaling factor
N_c	Number of sub-carriers
N_I	In-phase noise component
N_{IQ}	Total baseband noise
N_{MTC}	Number of MTC devices
N_Q	Quadrature noise component
p_d	Probability of detection
p_{fp}	Probability of false positives
P_N	Noise power
P_{PL}	Power attenuation due to path loss
Q	Quadrature (component)
t	Time variable
T_{coh}	Coherence period
T_{cp}	Cyclic prefix duration
τ_s	Delay spread
θ	Signal phase
T_s	Symbol duration

1. Introduction

The development and demand on mobile broadband have been rapidly increasing in the past 10 years, enabling fast internet almost everywhere in developed countries and with close to full technology penetration as subscribers is nearing all citizens in these countries. Developments on the wireless infrastructure have been made to provide the necessary capacity to support the vast growth in subscribers and traffic intensity from the first available mobile data access technology "general packet radio service" (GPRS, expansion of the GSM protocol) to "long term evolution advanced" (LTE-A) protocol, which have been deployed in the recent years. The LTE-A protocol, widely known as "4G" i.e. the 4th generation mobile communication, is as its name suggests an evolution of the "long term evolution" (LTE) protocol and by that, it is based on the same principles.

The LTE protocol is specified by the 3rd generation partnership project (3GPP), and was first introduced through Release 8 in December 2008. LTE-A was introduced in Release 10 (March 2011), after an enhancement of "standard" LTE in Release 9 (December 2009). After the introductory release of LTE-A, Release 11 and 12 has been made with enhancements to the LTE-A protocol, especially focused on providing faster connections with shorter delays through technologies such as carrier aggregation (CA) and higher order multiple input multiple output (MIMO), as well as overall improvements to the protocol. Also, as a part of the Release 12, implementation of a new category of communication, namely machine type communication (MTC), was introduced into the LTE-A functionality. [1]

Machine type communication (MTC), or machine-to-machine communication (M2M), is a vastly increasing market, due to a large increase within the internet of things (IoT): machines (or "things") communicate with each other without directly involving a user. By 2013, there was 9.1 billion IoT devices installed, and it is expected that there will be a total of 28.1 billion devices installed by 2020 [2]. An example of IoT devices is the implementation of smart grids by electricity, water, heat, and cooling providers. By implementing smart meters and other sensors which are remotely accessible throughout distribution grids, the way has been opened for instant readings of consumption, faster and eased leakage detection, easier and more efficient control of production, etc. [3].

Communication from sensors (and other MTC applications) give rise to a significantly different type of communication patterns, which requires changes in the established network in order for it to handle this efficiently. Communication demand from such devices can be both sporadic or with some (pseudo-)constant interval, but what is more distinctive to human operated communication, MTC sessions are mostly short with only small amounts of data to be transmitted; the amount of devices is however far greater in such systems compared to systems consisting of human operated devices, as city areas will have a very high density of devices. The massiveness of the devices in each cell is one of the main challenges of MTC, as a base station must be able to handle all the devices. Especially in terms of resilience against bursts in communication, in cases where many devices seek to send reports on the same time; an example of this could be in the event of power failure messages sent by electricity meters in a whole neighbourhood in case of power outage. Also, many of such devices will be battery powered and will therefore demand high efficiency and the support of low-power transmissions;

in these cases, it is important that transmission overhead must be small compared to the small packets to be relayed. [4, 5, 6]

One of the main points of the further development of LTE in order to be optimised for MTC is the "handshake" (the access reservation protocol (ARP)) which must be performed when a device is accessing the network. This step it vital, as it is generally required every time of a device is seeking to (re-)access the network, and in LTE it poses a high degree of transmission overhead compared to the expected size of data to transmit in MTC use cases. The future of mobile broadband, widely known as "5G", is partly going to be further enhancements of the LTE-A protocol (so-called LTE-A Pro) along with a new protocol named "New Radio". One of the main goals for the capabilities of 5G is to be able to compete with systems dedicated for IoT/MTC and not based on established systems provided by mobile network operators.

1.1 LTE basics

An overview of the basics of an LTE systems is given in this section, such that a better understanding and a clear definition of the aim of this thesis is possible.

The two main elements in LTE is the evolved node B (eNodeB), which is the name for the base station in a LTE system, and user equipment (UE), which is the devices accessing the network. Uplink transmission refers to transmission from the UE to the eNodeB, while downlink is the reverse. As discussed, MTC is becoming a focus of the further implementation and development of LTE – in this context, the term "MTC device" will be used for a device accessing the network. The wireless network nodes are illustrated in figure 1.1. The eNodeB is connected to a core network, which will be further discussed in chapter 3.

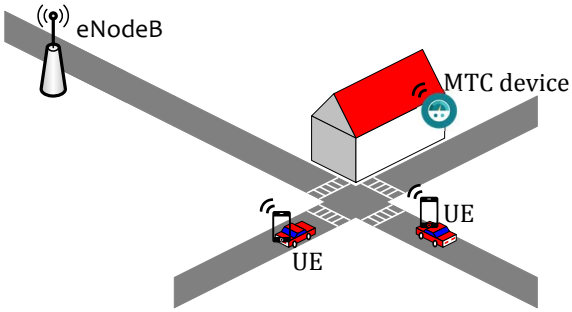


Figure 1.1: *MTC device (illustrated as a water meter in a household) and UEs (smart phones) accessing the network using a nearby eNodeB.*

The LTE protocol is divided into a protocol stack, which makes up what is comparable to the OSI model layers one to three. Further discussion of the LTE protocol principles, especially within the physical (PHY) layer, is found in chapter 2; overview of the protocol stack can be found in appendix A. In the following section, MTC solutions for LTE will be discussed, before this chapter will be concluded with a discussion of the aims and goals of this report.

1.2 Existing MTC technologies

Multiple solutions exist for wide-area MTC communication, both within licensed and unlicensed bands; solutions for unlicensed bands are typically known as low-power wide-area networks (LPWANs), while cellular technologies for licensed bands are grouped within their own category. Only the two most notable MTC protocols for licensed bands are investigated here, of which both technologies are further developments of LTE. The two technologies are LTE-MTC (LTE-M) and narrowband IoT (NB-IoT); an overview of the main differences can be seen in table 1.1. LTE-M is also known as enhanced MTC (eMTC).

TABLE 1.1: *Comparison of LTE MTC technologies, specification data combined from [7, 8, 9].*

	LTE Release 8	**LTE-M**	**NB-IoT**
Downlink peak rate	< 150 Mbps	< 1 Mbps	< 170 kbps
Uplink peak rate	< 50 Mbps	< 1 Mbps	< 250 kbps
Downlink bandwidth	1.08—18 MHz	1.08 MHz	180 kHz
Duplex mode	Full duplex	Full or half duplex	Half duplex
LTE compatibility	-	In-band	In-band, guard-band, and standalone
Power saving	-	eDRX and PSM	eDRX and PSM
Nodes supported per cell	In the order of hundreds or a few thousands	up to approx. 1 m	approx. 50 k to 200 k

These protocols are focused on supporting many devices and lower power consumption in the MTC device in order to prolong battery lifetime. The main change is therefore also apparent which is a reduction of bandwidth in both LTE-M and NB-IoT. This allows for simpler hardware implementation and in turn cheaper devices. A side effect of the reduced bandwidth is a natural reduction in data rate; this is however not so important as the amount of information is considerably lower for MTC devices, as previously discussed.

There are two new schemes introduced to the LTE regarding power saving in both LTE-M and NB-IoT: extended discontinuous reception (eDRX) and power saving mode (PSM). eDRX allows the MTC device to sleep between transmissions without having to reattach. In PSM, the MTC device can enter a sleep state giving the eNodeB notice on when it will attempt to transmit again; it will then move to an idle state where the device is reachable in downlink for a set amount of time, before going back to sleep. [7, 10]

One of the biggest differences between LTE-M and NB-IoT is the placement of the bands: while they both are able to coexist with existing LTE implementations by reserving part of the total band, NB-IoT is also able to run stand-alone (not inside an LTE band) and within the guard bands surrounding the existing LTE bands. They are both, however, simplified versions of the LTE(-A) protocol, and as such relies on the same basic principles. [7, 11]

1.3 Objectives of thesis

Implementations on MTC technologies are currently under way, both within LTE-M and NB-IoT; they have been specified in LTE Release 13 [12, 13], currently the latest, and implementations are being developed and tested. How real implementations and roll-out of these protocols and their associated devices will affect the communication network as a whole is still unanswered, because these networks have not been tested with large number of devices at the scale of massive MTC. MTC communication is already a part of the network using non-MTC specific LTE protocol releases and this type is expected to increase – vastly when deployment MTC specific parts of the protocol is complete. The effects on the eNodeB and core network with this increase of devices and device network access pattern are a main unknown for the mobile network operators; an unknown which must be answered to be able to efficiently expand the network and provide reliable connections both to the new MTC devices as well as the current mostly human operated devices. The possibility to evaluate protocols with regards to massive amounts of devices is important with regards to evaluation of protocol performance in MTC use cases, making way for the necessary adaptations and readjustments required in the core network. Development and testing of LTE MTC technologies are therefore of high relevance.

Evaluation of full implementations of a given protocol yields important insight into protocol performance. Conventional simulations usually evaluate small parts of a protocol individually, but this does not necessarily offer a clear display of the bottlenecks and other performance limitations of the protocol as a whole; it is difficult to combine individual results for the ARP and authentication performance to a realistic evaluation of all the network system elements. The alternative to simulating is expensive and cumbersome, as this requires a massive test bed with coordinated control of each unit so that different access scenarios can be tested; this would be difficult to make in practice and certainly expensive in hardware.

To solve the problems present in both simulations and real-life testing, a third solution is to be developed: a massive MTC device emulator implemented using a software defined radio (SDR). This “massiveMTC” emulator should be able to emulate individual MTC devices and their protocol stack, while transmitting their individual signals through the same RF port on the SDR, i.e. combining the signals from each MTC device. Such a system should ease performance testing of the LTE protocol in scenarios with a varying amount of MTC devices as well as different accessing behaviours. Having the setup localised on a single SDR board connected to a host PC allows for simple initiation and control while maintaining a wide range of applications, because such a unit can be used not only in a simple test bed but also connected to commercial eNodeBs, due to the fact that it performs live emulation which reacts directly with the eNodeB.

A massiveMTC device should therefore provide the ability to:

- Emulate entire protocol stacks of multiple MTC devices.
- Emulate individual channel conditions for the MTC devices.
- Provide information based on a realistic demodulator (as both up- and downlink are transmitted on a real channel).
- Have centralised control of the behaviour of each MTC device.

The design principle is illustrated in figure 1.2, where all functionalities are collected within one device.

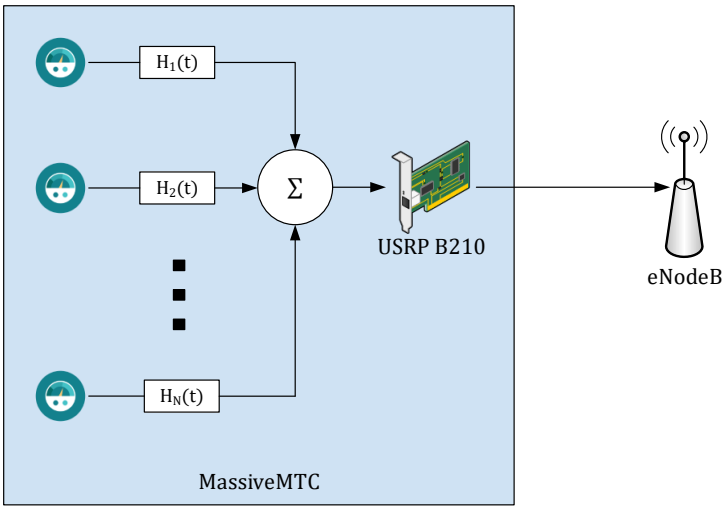


FIGURE 1.2: *MassiveMTC solution principle, where all devices are virtually created within the one massiveMTC device.*

One of the key elements to be evaluated in the protocol is the ARP, which potentially is a bottleneck as well as constituting non-negligible overhead. Simulation of this in LTE can be difficult to make in a way that provides insight of the performance of real-life systems, as such a simulation must implement not "just" realistic channel demodulation performance, but also how well the succeeding steps in the procedure can be performed dependent on how many nodes are active on the system, and how many are trying to get access to the network at the same time. It is currently sought to develop new versions of the attachment and authentication procedures in order to access the network for MTC devices [14, 15, 16], so that overhead from these initial steps can be reduced to obtain better efficiency when transmitting and increased support of massive numbers of devices.

The massiveMTC device will be developed using LTE Release 8 of LTE due to several considerations. While this implementation will strictly speaking constitute parallel UEs, it can be moderated to behave as MTC devices on a "standard" LTE network. The MTC protocols in development for LTE are in the deployment and testing phases and are therefore not directly implementable [17, 18]; but they generally rely on the same principles as standard LTE as they seek as much compatibility with existing LTE systems as possible [19]. The developed massiveMTC device will therefore be a viable platform for implementations of the MTC protocols to be used for further development process. It can thereby serve as a strong proof-of-concept for "massive device emulators" and a platform for further development of such massive devices in LTE protocol development.

1.3.1 Development context

The massiveMTC device is an implementation of the communication protocol to be emulated for multiple devices. This in itself can evaluate random access procedures and more, however, it can not on its own produce network traffic at the eNodeB and core network as would result from real devices accessing the network. Application layer behaviour must be added on top of the massiveMTC device implementation in order to be able to produce realistic traffic patterns.

Therefore in parallel to the development of massiveMTC, the development of a virtualisation of IoT operating systems (OSs) is carried out. This is carried out separately from the work on the massiveMTC device and is not part of this thesis. Each virtual IoT OS should then enable

the emulation of some specific MTC device with a designated network communication pattern (e.g. transmission interval, payload, etc.). With such a complete setup, the LTE network can be tested in regards to how different device behaviours will affect the system, and also if different combinations of devices on the network will be able to get the quality of service (QoS) required by each device. The principle of the whole system setup is illustrated in figure 1.3.

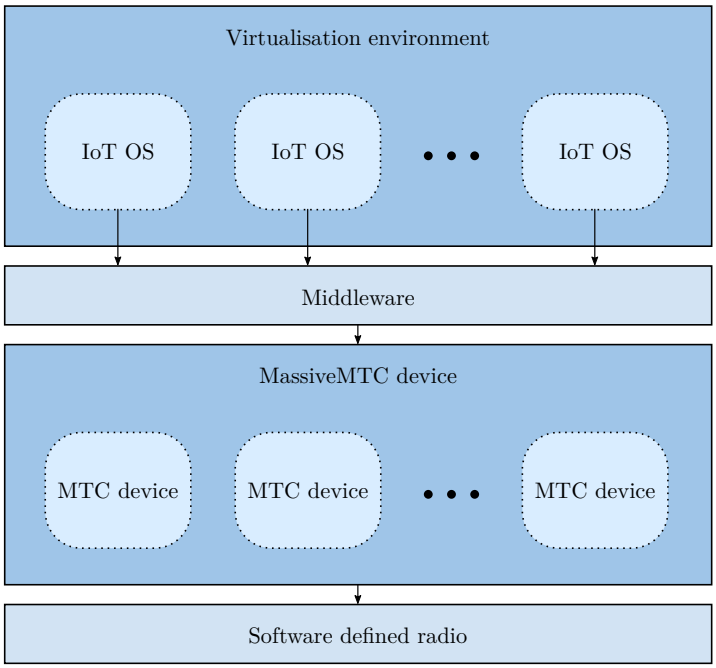


Figure 1.3: *MassiveMTC solution principle in context.*

The solution principle is devised such that a middleware also is to be developed, which routes information between the virtualisation environment and the massiveMTC device. The IoT virtualisation environment is to be run on a separate PC (or on an embedded device), and the middleware handles the traffic between one IoT OS instance within this environment and its corresponding MTC device protocol stack, thereby seemingly having its own radio to access the network. The initial development of the virtualisation environment is documented in [20], and work to implement this on an embedded platform as well as adding additional features is documented in [21]. The middleware is developed to be implemented either at the virtualisation environment PC, or in case of this being on an embedded device with a much stricter hardware limitations, on a standalone device, e.g. a programmable router or a raspberry pi. The current design of the middleware is a virtual private network (VPN) server solution to which both the massiveMTC device host PC and the virtualisation environment host are connected; on the middleware server, routing tables are managed between the virtualisation instances and the massiveMTC instances. As stated, the specification and development of neither the middleware nor the virtualisation environment is within the scope of this thesis, only an interface from the massiveMTC implementation to the middleware is provided.

2. Preliminary research

Before development on the massiveMTC device is possible, some technical details must be reviewed and discussed. In this chapter, baseband signal fundamental theory and the principle of SDRs are reviewed, as these are important fundamentals in regards to the implementation. Also, the core principles of LTE, especially for the lower layers, are discussed and a preliminary analysis of the LTE ARP is performed by use of simulations.

2.1 Baseband signals

Signal processing within wireless communication technology is usually performed on the baseband level, sometimes known as the information channel, in order to save processing power. The baseband signal can be written as a sum of two vectors, representing two signals in quadrature, known as the in-phase (I) and quadrature (Q) components; I and Q are used as a complex representation of a real signal.

This representation is viable for many digital modulation techniques, however only amplitude modulation will be discussed as this is the most dominant modulation form in LTE, although the principle remains the same for all modulation types. An example of a complex representation in a constellation diagram is shown in figure 2.1.

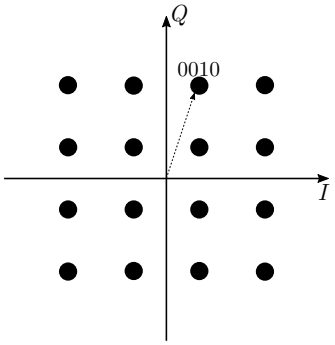


FIGURE 2.1: *Example of I and Q components in a constellation diagram using 16QAM modulation.*

The I/Q components are widely used within RF processing and are also the representation used for SDR modules. Due to this, a short review of I/Q fundamentals is given here. The magnitude and phase of any signal can be defined from the Cartesian form:

$$A = \sqrt{A_I^2 + A_Q^2}\,, \qquad \theta = \tan^{-1}\left(\frac{A_Q}{A_I}\right), \tag{2.1}$$

where A is the total amplitude of the vector, A_I and A_Q are the I and Q component amplitudes, respectively, and θ is the signal phase. Any constellation point can be obtained by having a relative difference in amplitude of the I and Q components. [22]

The total baseband signal can then be described in complex form:

$$s_B(t) = A_I(t) + jA_Q(t)\,, \tag{2.2}$$

where t is time. This signal can then be upconverted to RF signal:

$$s(t) = A_I(t)\cos(2\pi f_c t) - A_Q(t)\sin(2\pi f_c t)\,, \tag{2.3}$$

where f_c is the carrier frequency, and the negative sign of the Q component is by convention. With f_c specified, the sinusoidal terms are predictable, meaning that all the information is within the I/Q amplitude terms, i.e. A_I and A_Q. The I and Q terms can be summed, as the two signals are orthogonal to each other, and therefore do not interfere. [22]

In order to downconvert from passband to baseband, the in-phase amplitude component, A_I, of the incoming signal, $s(t)$, can be obtained, using a multiplication of $2\cos(2\pi f_c t)$:

$$\begin{aligned} s(t)2\cos(2\pi f_c t) &= 2A_I\cos^2(2\pi f_c t) - 2A_Q\sin(2\pi f_c t)\cos(2\pi f_c t) \\ &= A_I(t) + A_I(t)\cos(4\pi f_c t) - A_Q\sin(4\pi f_c t) \end{aligned} \tag{2.4}$$

The second and third term can be removed with a low-pass filter due to the high frequency carrier, f_c, thus leaving only A_I after filtering. The quadrature amplitude component, A_Q, can be obtained in a similar fashion, using a sine instead of a cosine for the multiplication. It is therefore seen that no information loss during the conversion between passband and baseband. [22]

It is important to note than an identical decrease in both I and Q components will result in a reduced amplitude of the signal while the phase remains unchanged; this can be used to emulate path loss on a channel. Changing I or Q independently will introduce amplitude and phase distortion of the signal; changing one or both components randomly can be used to emulate noise on a channel. Illustrations of this can be seen in figure 2.2.

(a) Reduced amplitude. (b) Phase change. (c) Frequency change.

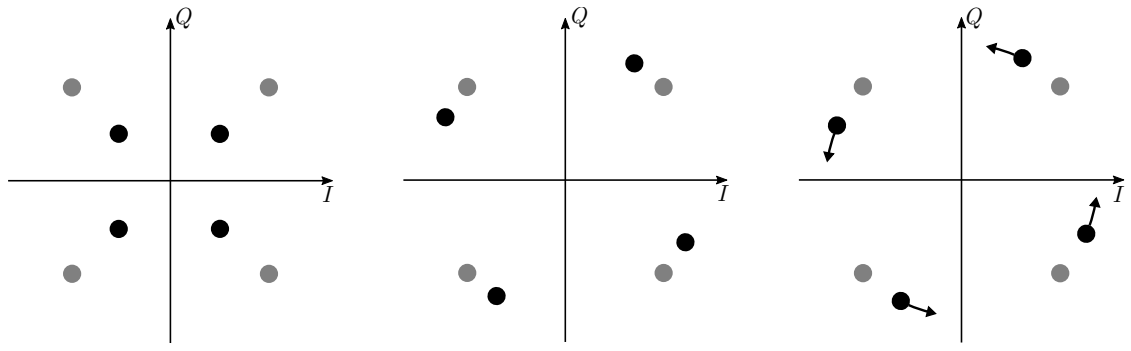


Figure 2.2: *Illustration of how a change in amplitude, phase, or frequency will affect I/Q samples. Grey dots represent the original samples.*

2.2 Software defined radios

Software defined radios (SDRs) can be thought of as reprogrammable radios where the physical layer can be modified significantly using software. These modifications can be split up into two parts; baseband parameters, such as modulation and error correction code, and radio front-end features such as configurable bandwidth and carrier frequency. An SDR is therefore a flexible communication platform capable of adapting to different applications.

There are more than one way to build an SDR, common to all of them is that a combination of hardware and software is required. Different signal processing devices exist which are useful when choosing an SDR platform, advantages of the three most commonly used are summarised below [23, 24]:

- General purpose processors (GPPs), also known as a "normal microprocessor", which are advantageous due to their flexibility and low costs.
- Digital signal processors (DSPs), specialised in performing signal processing calculations on a large amount of data in real time, with a relatively good power efficiency compared to GPPs.
- Field-programmable gate arrays (FPGAs), computationally powerful, but power inefficient and more cumbersome to program.

GPPs and DSPs are both programmed by software (e.g. in C-code), whereas an FPGA is programmed by use of a hardware descriptive language (e.g. VHDL), which strictly speaking is not "software". Some of the newest releases of commercial SDRs now offer a combination of a GPP and an FPGA, see for example [25]. Other implementation types have the software running on a host PC connected to the FPGA, which in a way is a combination of a GPP and an FPGA solution, discussed further in section 2.2.3. Purely GPP based SDRs are infeasible for today's requirements to sampling bandwidth, latency/jitter etc. with current architectures and are therefore generally not used, at least if complexity, power consumption, and size are to be taken into account [24].

Changing e.g. waveform and/or coding in an FPGA based SDR takes more time to reconfigure in comparison to a GPP based, as reconfiguration of the FPGA logic can be necessary. This can potentially be a drawback, especially if the SDR has to change configurations several times within short timespans. Whether this is necessary depends on the limitations of the FPGA and the specific use case, and the drawback can potentially be countered by pre-compiling configurations in a fixed region of the FPGA memory in advance while still having regions available for reconfiguration; this will however require more memory on the FPGA, increasing the cost of the unit. In order to mitigate memory usage, some processes can be moved from the FPGA to software (on the GPP or connected PC). [24]

Some of the main application areas of SDRs are cognitive radio, as it is crucial that the radio is able to sense existing radio signals or unexpected channel conditions and adapt by switching to an unoccupied frequency band and/or change the transmission bandwidth. Another perk of SDRs is the ability to use adaptive coding and modulation in dedicated hardware based on current channel conditions in order to guarantee a certain link performance, as well as being able to deploy a hierarchical network solution; an example of this is to use Wi-Fi when possible, and then switching to LTE when there is insufficient coverage of the preferred network. Some of the general drawbacks of using an SDR is the cost of the units, as they can be expensive dependent of what is required of them, as well as their power consumption when running. [24]

2.2.1 RF architectures

Multiple RF architectures exist, below are some of the most commonly used architectures used in SDRs explained. Only the receiver architectures will be discussed as the transmitter architectures generally are equivalent.

The simplest receiver is the "zero-IF" receiver; this is also called a homodyne or direct-conversion receiver. This receiver translates the RF signal from carrier frequency directly down to baseband samples, i.e. zero-IF means that there is no intermediate frequency (IF), such that the signal fed through the analogue-to-digital converter (ADC) is centred around 0 Hz. One of the problems with this design is the so-called "flicker noise". Flicker noise is generated by semiconductors around 0 Hz; this can especially be a problem for narrowband transmission in zero-IF as the entire wanted signal is around 0 Hz. In general, a minimal DC offset of the baseband signal is required, as well as closely matched gain and phase of the I and Q components in order to avoid spurious signals. This can however be compensated by adding a feedback loop. [24]

Another approach is the "direct-IF" conversion, or digital-IF, where a regular mixer translates the signal from carrier frequency to an IF; the ADC then converts the IF signal before it is downconverted to baseband. This approach avoids flicker noise present in a zero-IF converter. A bandpass filter is required in this design after the conversion to IF, which may be required to be tunable if large carrier frequency ranges are to be supported. When using more than one IF, the receiver is called a "super-heterodyne"; this is a widely used principle in conventional transceiver implementations. [24]

2.2.2 Digital converters

The digital down- and upconverters that are used in SDR implementations can be understood as software defined implementations of normally otherwise hardware implemented radio, such as filters, amplifiers, and baseband processing. The implementation is often of the direct-IF type, where the signal is downconverted to an IF and then sampled before being downconverted to baseband in the digital domain. The by far most used implementation of the digital converters is by using FPGAs, which is why this section will only discuss such implementations. The FPGA is sometimes used to only host the digital converters, although it is possible to do more signal processing on the chip [26], such as source and channel coding.

After the signal has passed through the ADC, it will continue into a digital down-converter (DDC) which is composed of three major components:

- A digital local oscillator operating at IF.
- A digital mixer, which, using the local oscillator, mixes with the signal, creating two new signals, one at double the IF and one around baseband (this happens due to the frequency shifting properties of the Fourier transform).
- A low-pass filter, filtering the signal at double IF frequency; this filter can also include a decimation filter, removing a set amount of samples.

The receiver chain is illustrated in figure 2.3. [27]

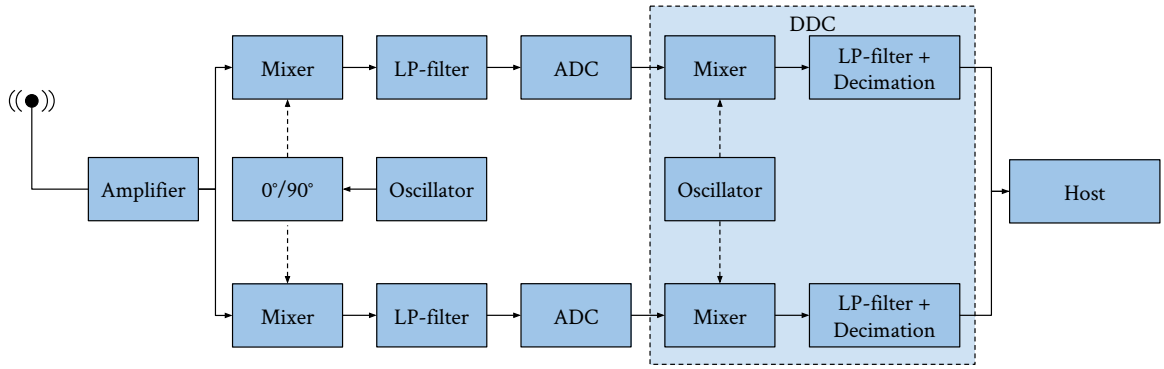


FIGURE 2.3: *Simplified receiver chain of an SDR.*

In SDRs, the frequency of the local oscillator can be changed, allowing change in sampling bandwidth, changing the RF frequency, and gain of the RF front-end is also possible, usually through a drive amplifier. [27]

The local oscillator in the DDC is able to switch between frequencies while maintaining a continuous phase, making it possible to sweep across a frequency range without creating any transients which would otherwise appear due to instantaneous phase changes [27]. A decimation filter is a filter which only keeps one of N samples, dropping the sample rate by a factor N, this is an effective way of lowering the amount of samples without losing any information, as long as the Nyquist theorem is fulfilled. The Nyquist-Shannon sampling theorem states that a band-limited signal can be uniquely determined by samples if $f_s > 2B$, where f_s is the sampling frequency and B is the bandwidth of the band-limited signal[1]. If this criterion is not uphold, aliasing of the signal will occur. [28]

The transmitter chain is slightly different; naturally, it utilises a digital up-converter (DUC) instead of a DDC, in which the major difference is that instead of a low-pass decimation filter, an interpolating filter is used. This performs the opposite operation of the decimation filter, i.e. adding more points in between the baseband input samples f_s/N, creating the interpolated output at rate f_s, thereby interpolating with a factor N. The transmitter chain is illustrated in figure 2.4. [27]

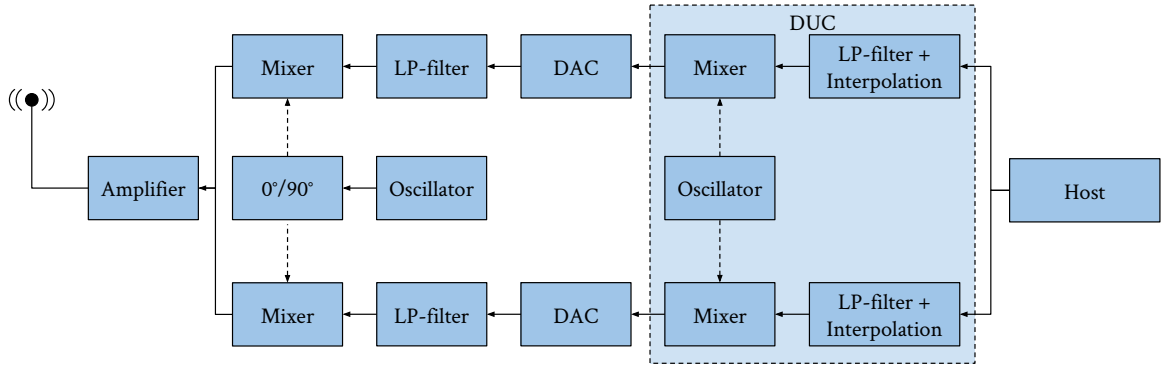


FIGURE 2.4: *Simplified transmitter chain of an SDR.*

The hardware (i.e. FPGA) implementation of DDC and DUC significantly lower the processor workload of the host PC, as it would otherwise have to process data at the ADC sample rate which

[1] B is known as the "Nyquist frequency", and $2B$ as the "Nyquist rate".[28]

can be several million samples per second, which is not feasible with modern processors. By having the conversion and decimation/interpolation performed in hardware, the software running on the host PC has to process signals at a much lower rate and bandwidth compared to the ADC or DAC. An added advantage is that some extra signal processing can take place in the FPGA and that a reduction in data-rate to and from the host PC means that less data has to be stored in memory.

2.2.3 USRP

The SDR platform used in this thesis is an USRP, developed by Ettus Research. Their approach on the applied B200-series is to compute all baseband operations on a PC, which acts as a "host" for the board, and that the USRP is working as a "radio peripheral", effectively giving the host PC the ability to decode radio signals. Ettus Research also provides as a library which makes it possible to communicate with the USRP from the host PC. The USRP library is open source, meaning that it is possible to change in the signal processing functions, as well as implementing additional logic on the USRP board which might be useful for some applications, e.g. in use cases where the modulation or error correction method are constant. [23]

Ettus Reseach provides "daughter boards" which can be mounted on the regular boards in order to provide a larger frequency range or bandwidth, as well as the ability to synchronise multiple boards. All baseband processing to the host PC, which enables the user to work with the signals using C instead of hardware languages. [23]

2.3 LTE principles

The LTE protocol is split into a protocol stack, illustrated in figure 2.5, in which there are four main layers:

- Packet data convergence protocol (PDCP) layer
- Resource link control (RLC) layer
- Multiple access control (MAC) layer
- Physical (PHY) layer

The PDCP layer handles IP packets, passed from the application layer (using TCP/UDP protocol). [29]

The protocol stack is illustrated in figure 2.5. All the layers are described in appendix A along with a short overview of their individual layer responsibilities.

For downlink transmission, LTE uses an orthogonal frequency-division multiple access (OFDMA) scheme, i.e. relies on the wideband technique orthogonal frequency-division multiplexing (OFDM) to obtain spectrum efficiency and resilience against frequency selective channels. OFDM is a multi-carrier technique: it divides the allocated spectrum into orthogonal subcarriers, each on which signals are modulated using conventional modulation methods. For uplink transmission, the related single carrier FDMA (SC-FDMA) is used. The basics of these techniques are discussed in appendix B.

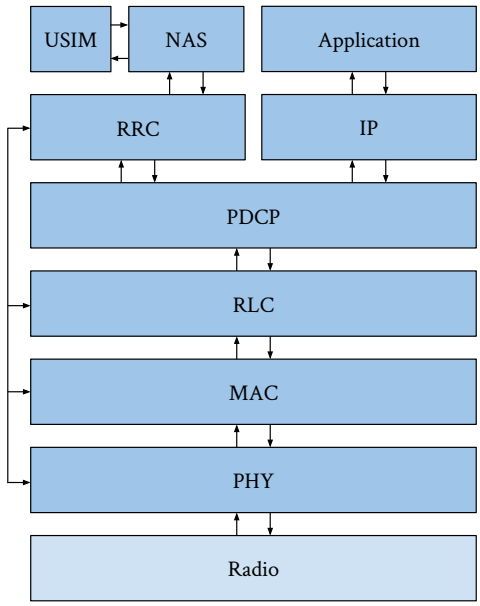


FIGURE 2.5: *LTE protocol stack of a UE.*

The LTE protocol is specified both for frequency-division duplexing (FDD) and time-division duplexing (TDD); by definition they result in different up-/download spectrum allocation as well as frequency deployment on a network perspective, each having their own pros and cons. Following discussion (as well as the above-mentioned appendix) will focus only on FDD systems; most core principles are however valid across both FDD and TDD implementations.

2.3.1 Frame structure

In LTE, the OFDMA principle is applied using both frequency-division multiple access (FDMA) and time-division multiple access (TDMA). This results in a frame that is composed of elements defined in both time and frequency. The highest element in time domain is a frame[2]; a frame in LTE lasts 10 ms and consists of ten subframes, each 1 ms. Subsequently, each subframe consists of two slots, where each slot is 0.5 ms. Every slot consists of either six or seven OFDM symbol durations depending on if cyclic prefix (CP) is extended or not. Combining the time domain with the frequency domain results in an overall frame structure for LTE, namely the resource grid.

The smallest element to be allocated for an UE is a resource block (RB), which has the duration of a slot and bandwidth of a physical resource block (PRB), i.e. 180 kHz; the PRB is divided into 12 subcarriers with 15 kHz spacing. Within a RB, resource elements (REs) are defined; these are the smallest elements in the resource grid and describe a single symbol modulated onto a subcarrier, i.e. it is one subcarrier wide in frequency and lasts one symbol duration in time. RBs are allocated in pairs within the same subframe, usually within the same PRB[3]. The downlink resource grid can be seen in figure 2.6 and in greater detail in figure 2.7. [29, 30]

[2] On the physical layer; larger elements are defined on higher layers.
[3] Frequency hopping within a subframe (known as *distributed mapping*) is however supported.

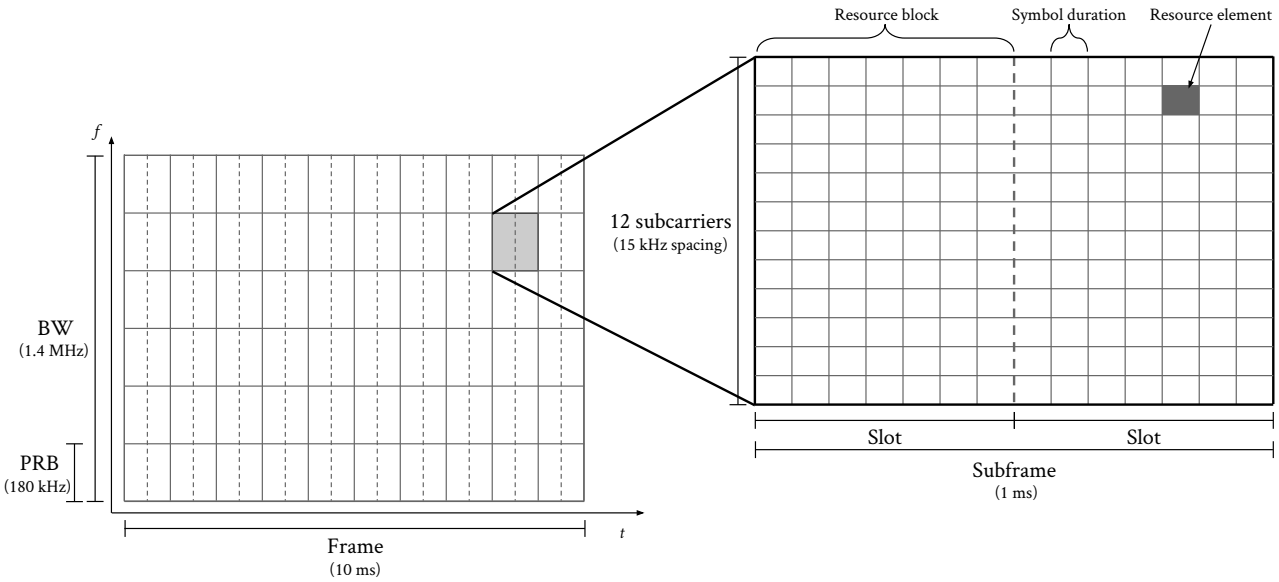


Figure 2.6: *OFDMA/LTE resource grid, illustrating how frequency and time are divided into blocks (example with lowest allowed bandwidth in LTE of 1.4 MHz excluding guard bands).*

Transport blocks (TBs) are passed from the MAC layer to the PHY layer, which is then mapped onto allocated RBs in the resource grid. The MAC layer may pass TBs once for every transmission time interval (TTI), where a TTI corresponds to one subframe duration. The TB size (TBS) is therefore dependent on the current number of allocated PRBs and the modulation and coding scheme (MCS). [29, 30]

2.3.2 Channels

There are three types of channels in LTE: logical channels, transport channels, and physical channels. Logical channels provide services between the RLC and MAC layer of the protocol and are generally deciding what type of information is transmitted, e.g traffic, control, or broadcast channels. Transport channels reside lower in the protocol stack, and are responsible for carrying data from the MAC layer to the PHY layer. The PHY layer defines how the data is transmitted, e.g. encoding and interleaving. Physical channels are the lowest level channels in LTE; they are mapped to the resource elements in the resource grid. An illustration of how the physical channels are mapped to the resource grid can be seen in figure 2.7. [29, 30]

As can be seen from the figure, multiple channels exist; the following is a list of all physical channels in LTE for both downlink and uplink.

Downlink:

- Physical broadcast channel (PBCH)
- Physical downlink control channel (PDCCH)
- Physical downlink shared channel (PDSCH)
- Physical control format indicator channel (PCFICH)
- Physical HARQ indicator channel (PHICH)

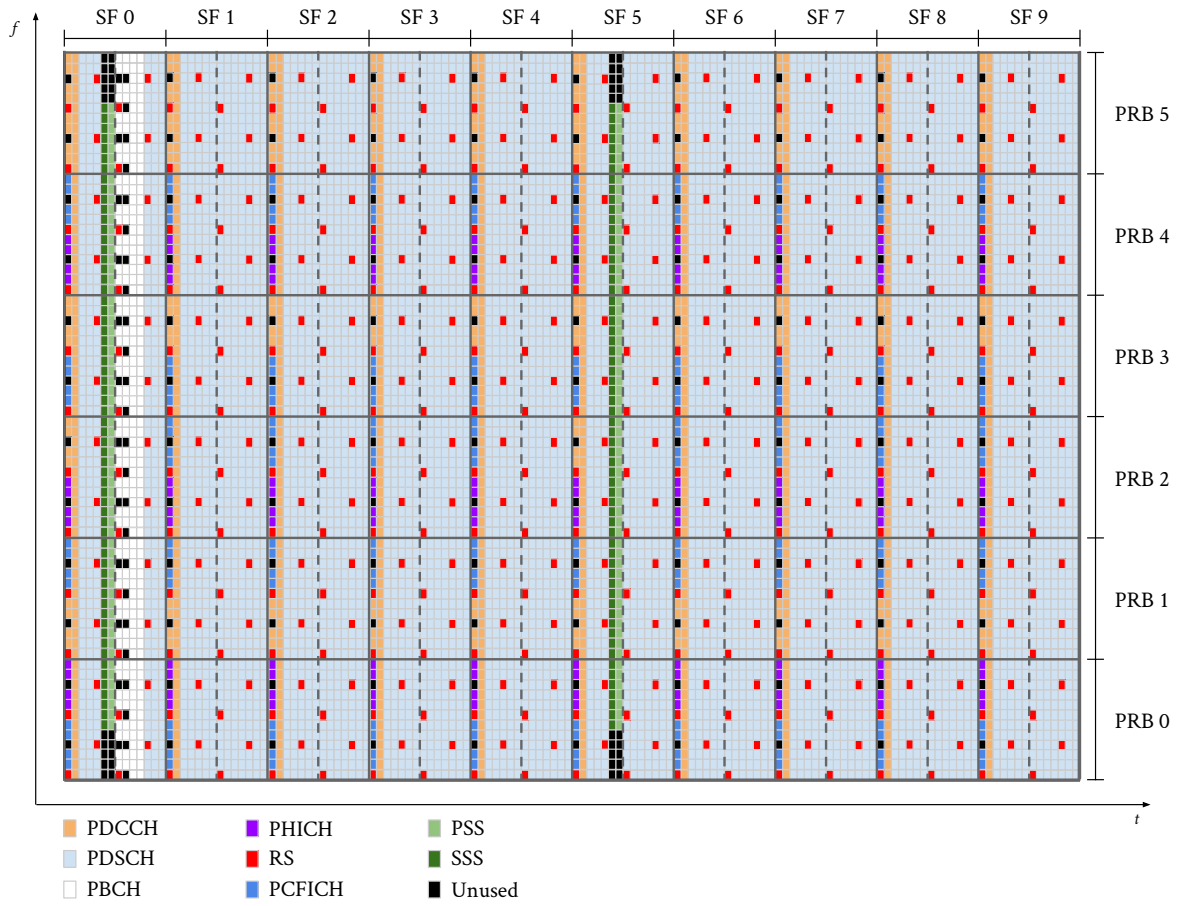


FIGURE 2.7: *Mapping of physical channels to the downlink resource grid for 1.4 MHz bandwidth.*

Uplink:

- Physical uplink control channel (PUCCH)
- Physical uplink shared channel (PUSCH)
- Physical random access channel (PRACH)

How logical, transport, and physical channels are mapped is reviewed in appendix A.2.

PDCCH is responsible for downlink resource allocation of the PDSCH on which the UE can receive data from the eNodeB as well as resource allocation in uplink ("uplink grants"). In uplink, PUCCH is used to transmit uplink control information (UCI) if no data is present, otherwise UCI is carried over to PUSCH (since an UE cannot transmit both PUCCH and PUSCH in the same subframe). UCI typically consists of hybrid automatic repeat request (HARQ) information, channel quality indicator (CQI) reports, as well as scheduling requests for uplink transmissions. [29, 30]

LTE is using adaptive modulation and coding (AMC) to obtain link adaptation; this is done to optimise the system capacity and coverage dependent on the signal quality available for the different UEs. Therefore, the uplink modulation used as well as the coding scheme applied is determined for each UE based on the latest received signal quality; the decision of the given modulation and coding scheme (MCS) is made by the eNodeB and the decision is contained in the downlink control

information (DCI) that also contains the uplink grant. The downlink MCS is also given in DCIs; the decision is made using CQI reports from the UE, this is discussed further in section 2.3.4. [29]

2.3.3 Establishing connection in LTE

When a device seeks to establish connection on an LTE network, three major steps are to succeed:

- Synchronisation with cell
- Obtain cell system information
- Access reservation

Frequency and time synchronisation is obtained by the decoding of primary and secondary synchronisation signals (PSS and SSS), which are transmitted in the beginning and in the middle of all frames, i.e. every 5 ms (light and dark green in figure 2.7); being able to track PSS and SSS yields subframe level synchronisation and physical layer cell identity, as this is embedded in SSS. [29, 31]

System information is received through master information block (MIB) and system information blocks (SIBs). MIB (transmitted on the PBCH) is the first packet to be decoded by the device, in which the most important parameters are available, e.g. system bandwidth and PHICH format. Having this information, the DCI containing system information can be found on PDCCH, which contains the positions of SIB messages. [29]

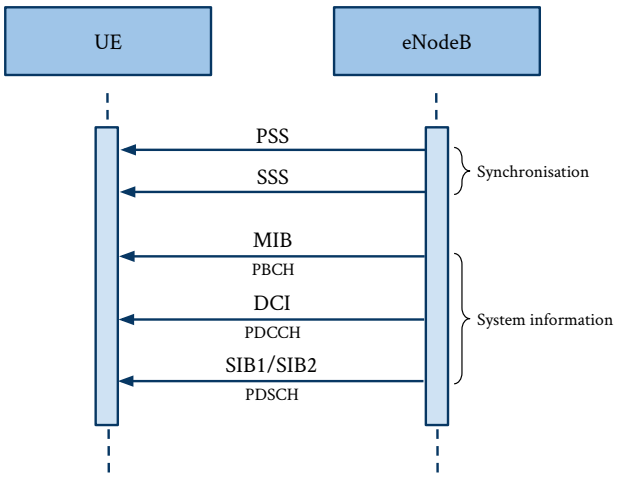


Figure 2.8: *Sequence diagram of synchronisation and download of system information.*

The SIB1 message contains cell access information and information necessary to locate other SIB messages. SIB2 messages contains control and shared channel configuration, including random access (RA) information necessary for the ARP. [29]

Synchronisation and system information messages are displayed in figure 2.8.

2.3.3.1 Access reservation protocol (ARP)

After cell search, synchronisation and subsequent successful decoding of MIB and SIB, the device can seek to successfully connect to an eNodeB through the ARP. The ARP is generally contention-based[4] and is constituted by a random access procedure (RAP) to obtain resource allocation. The UE will calculate the position of the next possible random access opportunity (RAO) in the resource grid and decide which RA preamble to transmit, using information about timing and preamble format obtained from the SIB2 message. It should be noted that the UE will retransmit the preamble with increased transmit power after a random back-off period, if it does not receive a response. [30]

The UE will then wait on a response from the eNodeB on PDSCH, namely the random access response (RAR, or simply MSG2). The RAR contains information about uplink timing adjustment, a temporary identity of the UE (temporary C-RNTI), and when (and where) the next message should be transmitted on PUSCH. It will usually be undetectable by the eNodeB, if two (or more) UEs have chosen the same preamble to be sent at the same RAO; the eNodeB will issue a RAR which will be received and understood by both (or all) UEs. Collisions are therefore not detected until the UEs' response to the RAR, namely the MSG3 which includes the RRC connection request.

The connection request is transmitted on PUSCH, and includes the received temporary identity along with specific subscriber details. As the contents of the messages are different between the colliding UEs, the collision can be detected and the procedure is stopped for the given UEs; these will have to restart the RAP. The eNodeB can specify in the RAR if the UEs should apply a random back-off before restarting the RAP. RRC connection setup (MSG4 on PDSCH) is the reply from the eNodeB to the connection request; this stage is called the contention resolution step. It contains the response to the connection requests. It is sent to the connection requests that are collision free, and allocates uplink resources for the UE if any are available, or denies the request if no resources are available. [29, 31, 32]

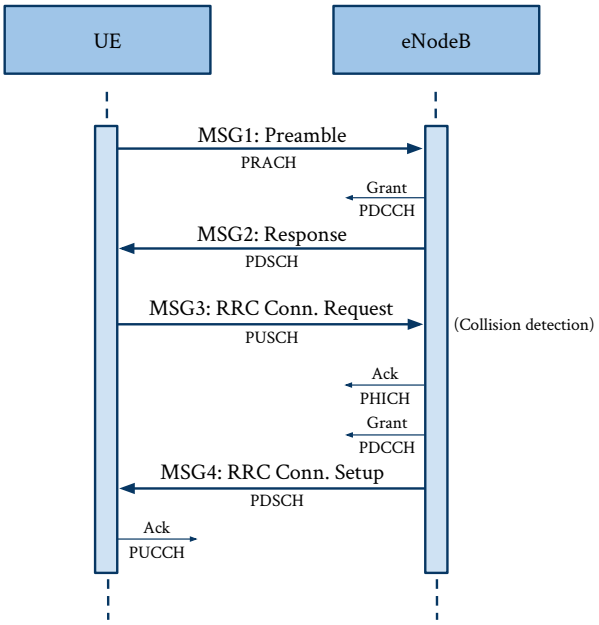


FIGURE 2.9: *Sequence diagram of random access procedure in LTE.*

[4]Can be contention-free in handover scenarios.

The steps in a successful and collision free RAP is illustrated in figure 2.9. It is followed by authorisation steps for the connection to be authorised by and registered on the core network; this is described in the following section.

2.3.3.2 Authorisation and other subsequent steps

Following a successful RAP, additional signalling is required in order to transmit and receive user data from the network. In RRC connection request and setup messages of the RAP, the RRC layer messaging is configuring the signalling radio bearer (SRB) of type one [29]; the RRC signalling is specified in [33]. MSG4 is responded to with the RRC connection setup complete message, in which the initial NAS layer message is concatenated; this is called the attach request or the NAS service-request message [30, 32].

After receiving RRC connection setup complete from the UE, the eNodeB will transmit an authentication request message, containing information received from the core network which should be used for authentication of the UE. The UE will then use data from this message to prepare the authentication response message, which contains authentication information from its SIM card. A more in-depth explanation of the authentication procedure can be found in appendix C.

Upon successful authentication, a security mode command is transmitted on PDSCH in order to activate integrity protection and ciphering. Integrity protection allows the receiver to detect packet insertion or replacement, while ciphering protects from a third party receiving and decoding the signal. The UE will verify the content of the security mode command messages and if successful, apply integrity protection and cipher to subsequent messages before answering with a security mode complete message. It should be noted that this message is the last message which is not protected by ciphering. The eNodeB will transmit a RRC connection reconfiguration message, responsible for establishing SRBs of type two and the data radio bearer (DRB); it may also include eventual piggybacked NAS layer messages. The UE will, if successful, answer with a RRC connection reconfiguration complete message, allowing for user data transfer using a DRB. [29, 34]

The eNodeB is allowed to send RRC connection reconfiguration before receiving security mode complete, it should however release the connection if one of them fails [29]. The necessary signalling for successful authentication, including acknowledgements, scheduling requests (SRs), as well as up- and downlink grants are shown in figure 2.10.

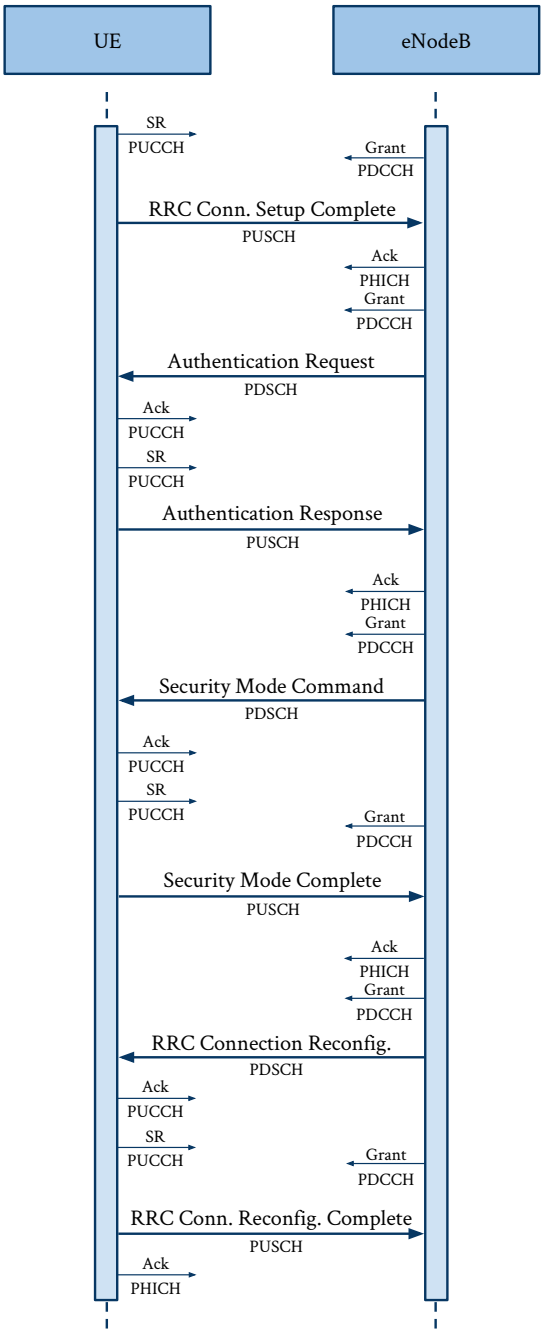


FIGURE 2.10: *Additional signalling following the RAP in LTE, produced using information from [16, 29].*

2.3.4 Power control

Power control is an important part of the LTE protocol to enable reliable mobile communication. In LTE, intra-cell interference is not an issue in the same way, as it has been in previous mobile communication technologies, as the OFDMA by design provides orthogonality between uplink streams. However, efficient power control is still an important feature, as it ensures sufficient transmitted energy in order to provide the desired QoS while keeping power consumption in battery powered devices as low as possible. Generally, transmission power in LTE is calculated using power per RB (PPRB) or energy per RE (EPRE). Uplink data rates on the channel are controlled by the MCS, while for a

given MCS setting the transmission power is kept constant. The power control consists both of an open loop method and a closed loop method using transmit power control (TPC); the PPRB is kept constant for all RBs on transmission. Also for uplink, the "power headroom" is tracked to not exceed the maximum power allowed in resource allocation. [29]

For downlink transmission, there is no individual power regulation for signals to different UEs. Instead, the channel conditions of the downlink channel is reported using CQI, so that appropriate MCS can be used [29]. Additional types of channel status information (CSI) signalling are specified for MIMO transmission [29]. The CQI is related to the signal-to-noise ratio (SNR), however, how the CQI is mapped to a specific channel condition is vendor specific; the CQI scale is defined to go from 1 to 15, where 15 represents excellent channel conditions [35]. The CQI values corresponds to different MCS settings. Also, power offsets between channels are introduced to boost reference signal (RS) detection [29].

In the following sections, an overview of some of the essential power control elements is given. Power control is specified in [36].

2.3.4.1 Open loop uplink power control

By definition, closed loop power control is not possible for the ARP signalling. Until attachment to the cell, open loop control is therefore applied. The procedure applies a target received power specific for the RA preamble (set by the eNodeB, indicated in SIB2 messages) and an estimation of the path loss between the eNodeB and UE. The path loss is estimated from the RS received power (RSRP) and its transmission power, which is also indicated in the SIB2 messages. The transmission power is limited by a maximum power specified by device category. [29, 36]

The open loop estimation is an estimate of the downlink channel, while its application is for uplink. This is carried out under the assumption that the uplink and downlink channel are long term reciprocal. [29]

2.3.4.2 Closed loop uplink power control

The closed loop uplink power control is possible after attachment to the cell, enabling feedback in the control system. The power control is then handled individually for sounding reference signals (SRS), PUCCH, and PUSCH transmission. Transmission on all three channels are bounded by the same maximum transmission power. Each channel power is calculated using different parameters, partly specified by upper layers and partly given in PDCCH data. However, all three algorithms incorporate both an open loop component (direct path loss estimation as for ARP) as well as closed loop calculations that changes on a larger time scale ("semi static"). The power control for all three channels is revised every subframe, so that power control is performed for every 1 ms. [29, 36]

2.3.4.3 Power headroom

The power headroom is a measure of how close the UE transmission power on PUSCH is to the maximum transmission power of the UE. It is MAC layer control information that is used in the eNodeB when allocating resources; power headroom reportings (PHRs) are sent to ensure that additional RBs within a given subframe are not allocated to a device, if this would require the maximum transmission power to be exceeded. Transmission of a PHR is occurring periodically and if

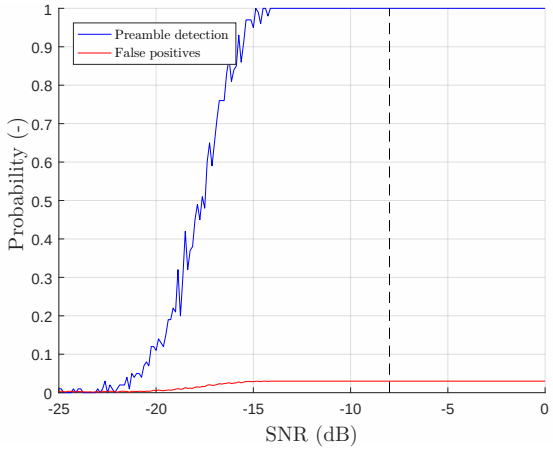
the estimated path loss changes more than a given threshold. The PHRs are only transmitted within subframes in which the UE has an uplink grant. [29]

2.4 Preliminary analysis of PRACH detection

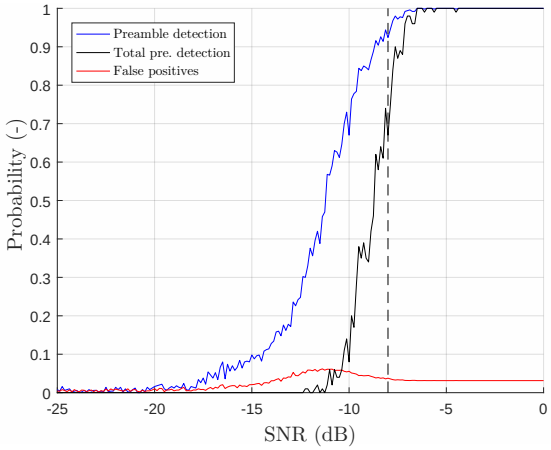
In order to understand the limitations of multiple UEs attempting to connect to the network, simulations have been carried out to investigate the expected preamble detection probability for a given RAO. This test is performed using a range of UEs, as seen in figure 2.11. The tests are carried out in MATLAB using the LTE System Toolbox for generation of baseband preamble signals as well as the detection after white Gaussian noise is added in order to obtain the desired SNR.

In the scenario with only one UE, a random preamble is chosen out of the 64 available as would normally be the case if the eNodeB only allows group A preambles; the resulting signal is generated, channel noise is added, and detection of the preamble is attempted; this is carried out N_{trials} times.

(a) 1 UE transmitting at a given RAO.



(b) 5 UEs transmitting at the same RAO.



(c) Combined plot of preamble detection simulation with 1 to 30 UEs transmitting at the same RAO.

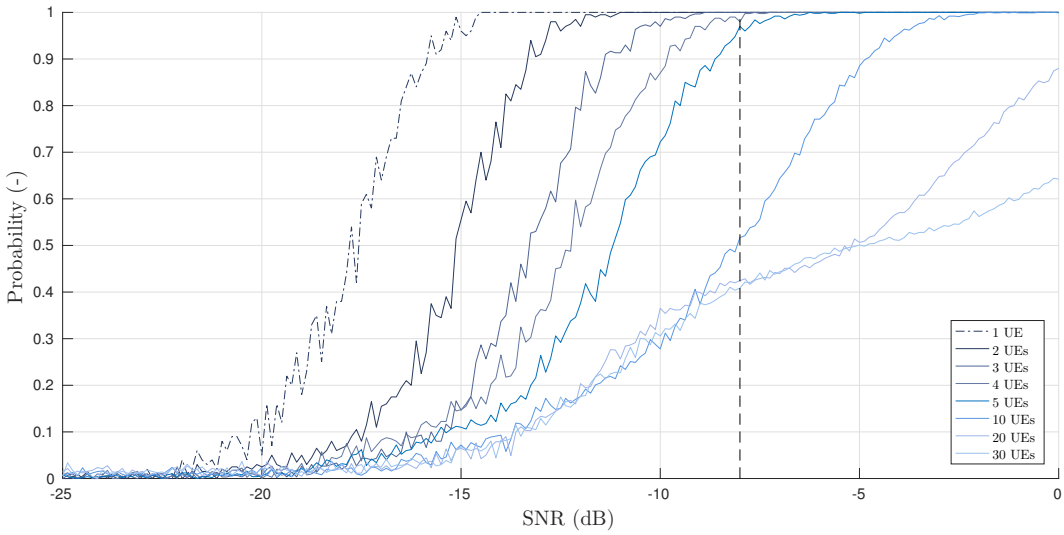


FIGURE 2.11: *Simulated preamble detection results.*

The probability of successful detection, p_d, is estimated as follows:

$$\widetilde{p}_d = \frac{N_{\text{det,tx}}}{N_{\text{trials}}} , \tag{2.5}$$

where $N_{\text{det,tx}}$ is the number of preambles detected given that the preamble has been transmitted. In addition, the probability of false positives, p_{fp} is also found:

$$\widetilde{p}_{\text{fp}} = \frac{N_{\text{det,ntx}}}{63 N_{\text{trials}}} , \tag{2.6}$$

where $N_{\text{det,ntx}}$ is the number of preambles falsely detected, i.e. given that the preamble has not been transmitted. The division of 63 is due to that there is 64 preambles to chose from, one of which is sent, i.e. there is 63 possibilities of a false positive. The p_{fp} is then the probability of preamble with index i is falsely detected, which is assumed equal for all of the residual preambles.

When multiple preambles are sent in the same RAO, N_{sent} preambles are chosen randomly to be transmitted. The following is used to calculate the probability of detection:

$$\widetilde{p}_d = \frac{N_{\text{det,tx}}}{N_{\text{sent}} N_{\text{trials}}} . \tag{2.7}$$

And likewise for the probability of false positives:

$$\widetilde{p}_{\text{fp}} = \frac{N_{\text{det,ntx}}}{(64 - N_{\text{sent}}) N_{\text{trials}}} . \tag{2.8}$$

Also, for the simulations with multiple UEs transmitting a preamble, the probability of receiving all preambles sent in a PRACH slot is calculated as well; this is denoted "total preamble detection" in the figures. The -8 dB limit is added, as the probability of detection shall be equal to or exceed 99 % for burst format zero (i.e. preamble format) at -8 dB SNR (for single UE transmission), as specified in [37]. Figure 2.11(c) illustrates what happens when an increasing number of UEs attempts to connect to the eNodeB in the same RAO, in order to give an overview of the performance of preamble detection for a varying amount of UEs. Additional plots from the analysis can be found in appendix D.

It is seen from the simulations that when five UEs tries to use the same RAO, only 80 % (at the -8 dB SNR level) of these will be detected. This is a bottleneck for a systems with a large amount of devices; this is also in a best case scenario without collisions. The probabilities for successful detection will worsen taking collisions into account, which is more likely given that some preambles are usually reserved for contention-free access (handovers).

3. System setup: MassiveMTC test bed

The massiveMTC device is to be developed on an SDR platform connected to an eNodeB emulator. The connection between eNodeB and the massiveMTC device is made through cable so that interference is not introduced in the licensed spectrum.

Both the massiveMTC and the eNodeB are to be implemented using USRP B210 boards that feature Xilinx Spartan 6 FPGAs, as LTE implementations are known to be made on such boards. The open source project srsUE [38], developed by Software Radio Systems, is used as framework to perform the UE/MTC device protocol stack functionality; this is discussed in section 3.1. For the evolved UMTS terrestrial radio access network (E-UTRAN) and core network (CN), the open source project OpenAirInterface is chosen, this is discussed in section 3.2. Both the srsUE project and the OpenAirInterface have been verified to be supported on the USRP B210 board.

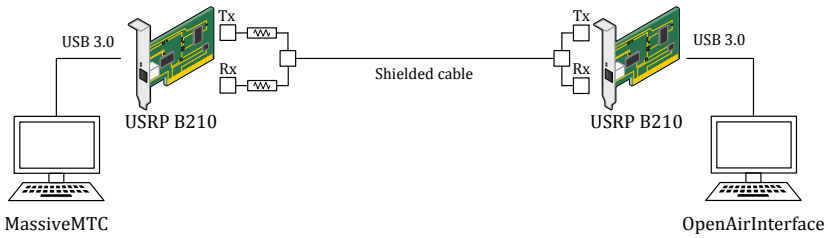


FIGURE 3.1: *MassiveMTC test bed setup.*

Figure 3.1 illustrates the setup, it consists of two B210 boards as explained above. Two ports on each of the boards are connected to splitters/combiners, this is required since the board transmits and receives on different ports. Additionally, 30 dB attenuators are introduced in order to attenuate the signal so that the maximum input power is not exceeded due to the low path loss of the cable as opposed to a wireless link. The boards are connected the PCs running massiveMTC and OpenAirInterface, respectively, with USB 3.0. Further discussion about the implications of the setup described in this section can be found in section 4.2.

3.1 SDR implementation of MTC device

As stated, the srsUE implementation is used as basic framework of the UE/MTC device protocol stack. It is a modular C++ implementation of the device side protocol stack using LTE functionality provided by srsLTE, which is an open source C library providing core LTE functions. The protocol stack in srsUE is defined as illustrated in figure 3.2. The defined radio layer makes use of the srsLTE API, defined in srsLTE, to send commands to the USRP hardware driver (UHD) [39], which is the driver used for accessing the USRP board. All of the srsUE modules and srsLTE functions are run on the PC, sending baseband signals and required metadata to the SDR board through the UHD.

The virtual network interface (VNI) is implemented as a Linux “TUN device”, which is a tunnelling interface that relays IP packets; in this way, applications can access the UE gateway and make use of the LTE connection.

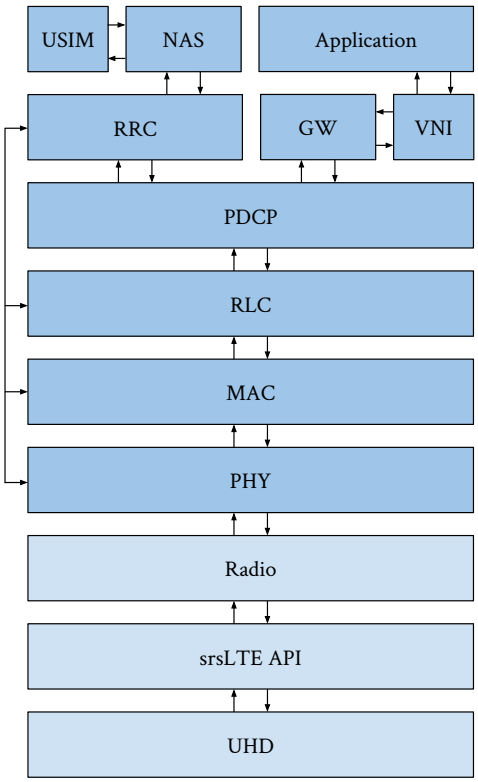


Figure 3.2: *LTE protocol stack as implemented in srsUE.*

It should be noted that srsUE is implemented in such a way that while all protocol layers are initiated at start-up, most of them wait before they start processing on a subframe basis. The only layer active from the beginning is the PHY layer, while the MAC layer simply waits for synchronisation is reached. Layers above the MAC perform no operations before first steps in the RAP has occurred. This means that the CPU load of srsUE is dependent on how far the UE is in the attach procedure.

The MAC and PHY layer are both split into several modules. The PHY layer consists of the following modules:

- (General) PHY module
- PRACH module
- PHY channel receiver module
- PHY channel worker module
- PHY channel common (worker) module

The general PHY initiates the workers and generates a pool of these, as well as create instances of the PRACH and common worker modules. After initiation, it is used as an interface to the PHY modules for the upper layers. The PRACH module handles all PRACH procedures, including generating the preamble signals. The PHY channel receiver handles cell search, cell synchronisation, and reception of each subframe by parsing subframe data to a channel worker for decoding after calculating carrier

frequency offset (CFO) and sampling frequency offset (SFO), tracking time, and more. Essentially, the PHY channel receiver is the backbone of the PHY layer, inasmuch as its thread provides bookkeeping of subframes for all layers and initiates decoding for each received subframe. The channel worker performs the decoding of subframes; see appendix F for detailed trace of down- and uplink processes. Several are instantiated, and for each subframe an idle worker is given the data from a subframe to process. In a similar fashion, they are used for encoding a subframe for transmission. The common channel worker handles PRACH, which is prepared before a RAO, and the decoding of RAR; it is also where channel properties are saved, such as estimated path loss and information calculated from PSS/SSS (such as CFO and SFO). In this way, the common channel worker is the interface between the individual workers.

The MAC layer consists of the following modules:

- (General) MAC module
- Multiplexing (MUX) module
- Demultiplexing (DEMUX) module
- Downlink HARQ module
- Uplink HARQ module
- PCAP[1] module
- Protocol data unit (PDU) module
- Procedure modules for:
  - Buffer status reporting (BSR)
  - Power headroom reporting (PHR)
  - Random access (RA)
  - Scheduling request (SR)

The investigation of the implementation has mostly been of the procedures for RA and SR. It is the SR procedure that signals the RA procedure instance to commence. The flow of initiation of PRACH and transmission of the preamble is displayed in figure 3.3.

The uplink processing in srsUE is always four subframes ahead of the downlink reception, in the sense that data is prepared four subframes before it is actually transmitted. This makes it possible to stream the data over USB 3.0 to the buffers on the SDR before it should be transmitted without this being time critical; transmission time is handled using metadata through the UHD. The reason that it is exactly four subframes ahead is that srsUE must read ACKs/NACKs at the current subframe, as eventual retransmissions is transmitted four subframes after a NACK is received [29].

[1] "Packet capture", file output for tracking in network protocol analysers such as Wireshark.

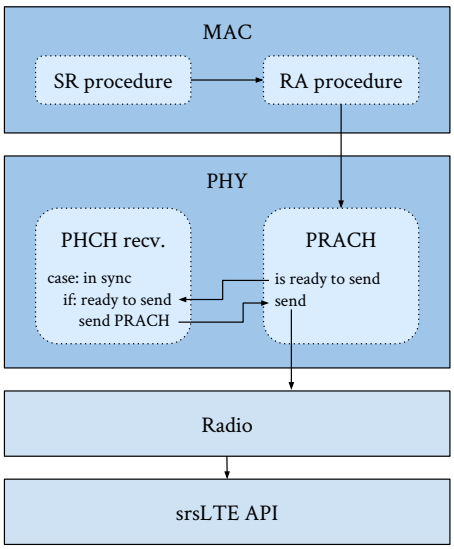


Figure 3.3: *Simplified PRACH transmission trace in MAC/PHY layer.*

3.2 LTE system architecture

The network architecture of LTE is composed of three main parts: user equipment (UE), evolved UMTS terrestrial radio access network (E-UTRAN), and core network (CN) which specifically for LTE is more widely known as evolved packet core (EPC). The combination of E-UTRAN and EPC are known as the evolved packet system (EPS); the setup is illustrated in figure 3.4. E-UTRAN handles communication between UE and EPC and is composed by eNodeBs in a flat network architecture. It should be noted that an eNodeB is a logical node, and not a physical construction, e.g. an eNodeB can be a three-sector site, pico eNodeB controlling multiple pico cells, or a home eNodeB (HeNB) controlling a femtocell. [29, 30]

The eNodeB is responsible for scheduling, transmitting, and receiving data to and from UEs. eNodeBs is capable of handover to neighbouring eNodeBs, which together with load and interference information is the main purpose of the X2 interface between eNodeBs. [29]

The EPC is responsible for the overall control of all UEs and establishment of EPS bearers, which are "communication paths" from UEs to the internet [29]. The main logical nodes of the EPC are:

- PDN gateway (P-GW)
- Serving gateway (S-GW)
- Mobility management entity (MME)
- Home subscriber service (HSS)
- Policy control and charging rules function (PCRF)

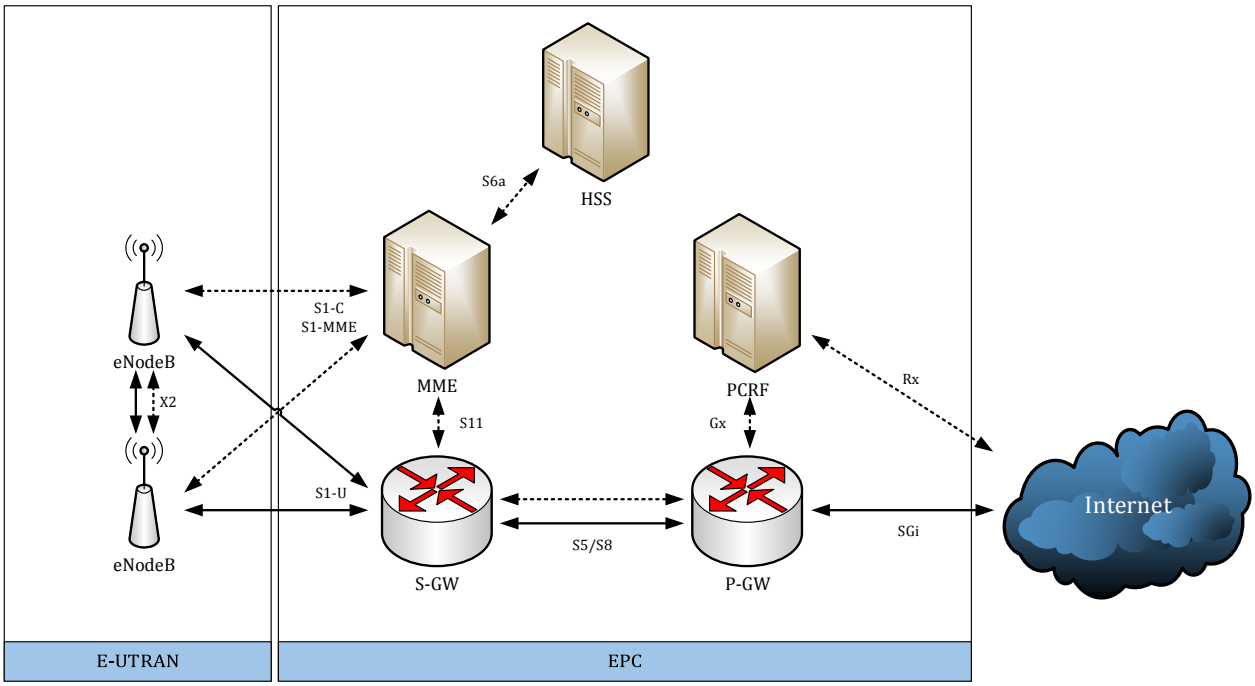


FIGURE 3.4: *LTE system architecture with interface names. Dotted lines represent control information and full lines is data.*

The EPC can be split up into two planes, namely user and control plane. Both S-GW and P-GW operates on the user plane, as they transport IP data traffic between UE (via eNodeB) and the external network (the internet). S-GW is the gateway between eNodeB and EPC, while P-GW is the gateway between EPC and the external network, communication between P-GW. The gateways may be seen as independent logical nodes, but are often combined in a single node, known as the system architecture evolution (SAE) gateway. [30]

An MME operates on the control plane, and as such handles signalling related to mobility and security for E-UTRAN. The MME processes information between UEs and the EPC, for which the NAS protocol is used. The HSS is a central database which contains information about subscribers on the network, such as telephone number and universal subscriber identity module (USIM) data. PCRF is responsible for charging functionalities and ensures that this is in accordance with the users subscription profile. [29]

The NAS layer can be seen as the equivalent of the network layer in the OSI model, and is used for managing UEs on the network. This includes handling establishment, handover, and release of bearers, while also handling the establishment of connections and security between the network and UEs. The access stratum (AS) protocols are the joint designation of the protocols/layers that handles communication between the UE and eNodeB, i.e. MAC, RLC, and PDCP layers. It can be seen as the equivalent of the data link layer in the OSI-model. Some of the most important tasks for the AS layer is radio resource allocation in both uplink and downlink, header compression, security, and position estimation of the UE. [29]

3.2.1 OpenAirInterface implementation

OpenAirInterface is a software alliance which offers an open source software implementation of UE, eNodeB, and EPC (excluding PCRF) [40]. The implementation used to emulate the eNodeB and EPC is an "all in one" setup, where all nodes are deployed on the same PC, as seen in figure 3.5.

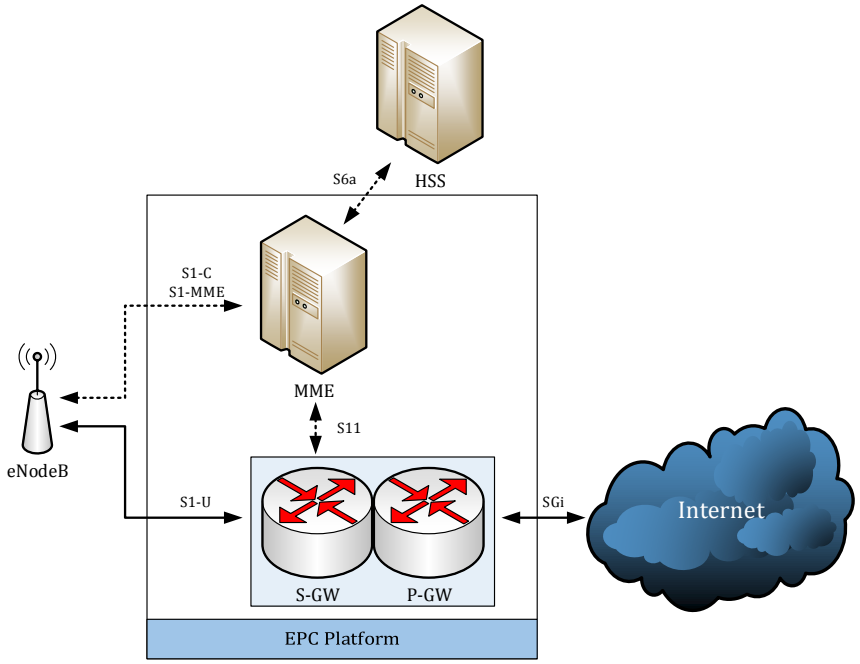


Figure 3.5: *OpenAirInterface E-UTRAN/EPS setup.*

It can be seen from figure 3.5 that S-GW and P-GW are merged together such that there is no interface between the two nodes. The interface between MME and S-GW is virtual, meaning that the signals are not sent through a network stack.

The HSS is deployed on the same PC as the EPC. The HSS is implemented using an SQL database in order to store the information for all users; more info on how subscriber data is added to the HSS database can be found in appendix C.4.

4. System development

The development of the massiveMTC device is split into several stages; the major steps are documented in this chapter. First, it is sought to devise a development principle to enable "parallel" devices and use this to develop a device that enables emulation of independent MTC devices. Initial development of the principle and prototype implementation is completed with two devices in parallel as a proof of concept. Subsequently, with a core principle well-defined and a two-device prototype, development continued to focus on scaling and improving reliability.

As stated in chapter 3, the device protocol stack functionality will be derived from the srsUE implementation [38], which implements the full protocol stack along with the srsLTE API to the UHD, enabling transmission over the USRP B210.

4.1 Development of core principle

To have independent MTC devices running on the same device using the same RF front-end, several concerns must be taken into account. The two main objectives are to receive the (downlink) signals and parse these to each individual protocol stack, meanwhile for uplink, the signals for transmission from each MTC device must be combined appropriately before transmission using the SDR.

The implementation principle devised is performed exclusively on the PHY and radio layer, while the upper layers are duplicated. Duplicating the higher layers is deemed a technically viable approach, due to the fact that the most computational heavy processes are placed in the physical layer functions, specifically in the synchronisation and turbo coding processes; also individual higher layers are important for the devices to act independently, which is necessary for the emulation to be realistic.

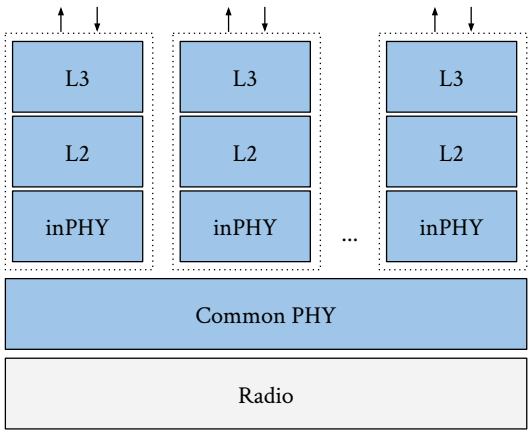


FIGURE 4.1: *Core principle in development strategy: radio layer is shared, on top is the PHY layer (layer one) split into a common part and an individual part.*

Thereby, an approach with the following implementation modules is chosen:

- A common physical layer (coPHY), where calculation heavy and common functionality should be computed once for all MTC devices.
- Individual physical layers (inPHYs), where individual decisions are made.
- Extra functionality in the radio layer in order to enable simultaneous transmission of signals from all MTC devices.
- All layers above the PHY are completely duplicated for each device, enabling full individual functionality for these layers.

The design philosophy is illustrated in figure 4.1.

4.1.1 Individual and common physical layers

The coPHY is handling all steps with regard to getting received signals from the SDR; it has three major responsibilities:

- Cell search
- Cell synchronisation
- Synchronous reception, parsing control and data information to each inPHY.

Cell search is deemed impractical to handle individually, as this requires changing the SDR frequency (sampling frequency), which practically cannot be handled in parallel. By that, access to different cells will not be supported by the massiveMTC device. Synchronisation is also managed in the coPHY, as this is identical between the MTC devices since they have the same physical access to the channel, i.e. the shared antenna front-end; the tracking of subframe number (SFN) is thereby occurring here and is shared between the individual MTC devices. As a result from this, PSS/SSS as well as all MIB and SIB decoding occur in the coPHY. When the coPHY is synchronised with an eNodeB, it will signal the inPHYs and start passing received signals each subframe.

The inPHY handles both the down- and uplink processing that subsequently occur when receiving synchronously from the eNodeB. Decoding of downlink signals is vital, as all uplink transmissions are scheduled and granted by the eNodeB. The up- and downlink grants are given individually to the MTC devices, and must therefore be handled by the inPHYs. Thereby, the main purpose of the inPHY (in downlink) is to decode all PDSCH or PDCCH allocated for the given device, along with information from PHICH in which ACK/NACK messages are transmitted. In regards to uplink, the inPHY is responsible for decoding and handling uplink grants (given in DCI format 0 messages on the PDCCH) and subsequent encoding of PUSCH or PUCCH, as well as generating SRS.

Upon transmission, the inPHY writes to two buffers in the radio layer: a buffer with transmission control information, and a buffer with the transmission samples; these buffers are described further in section 4.1.2. In addition to the buffers, a flag is set to signal subframe processing has finished. This is illustrated in figure 4.2.

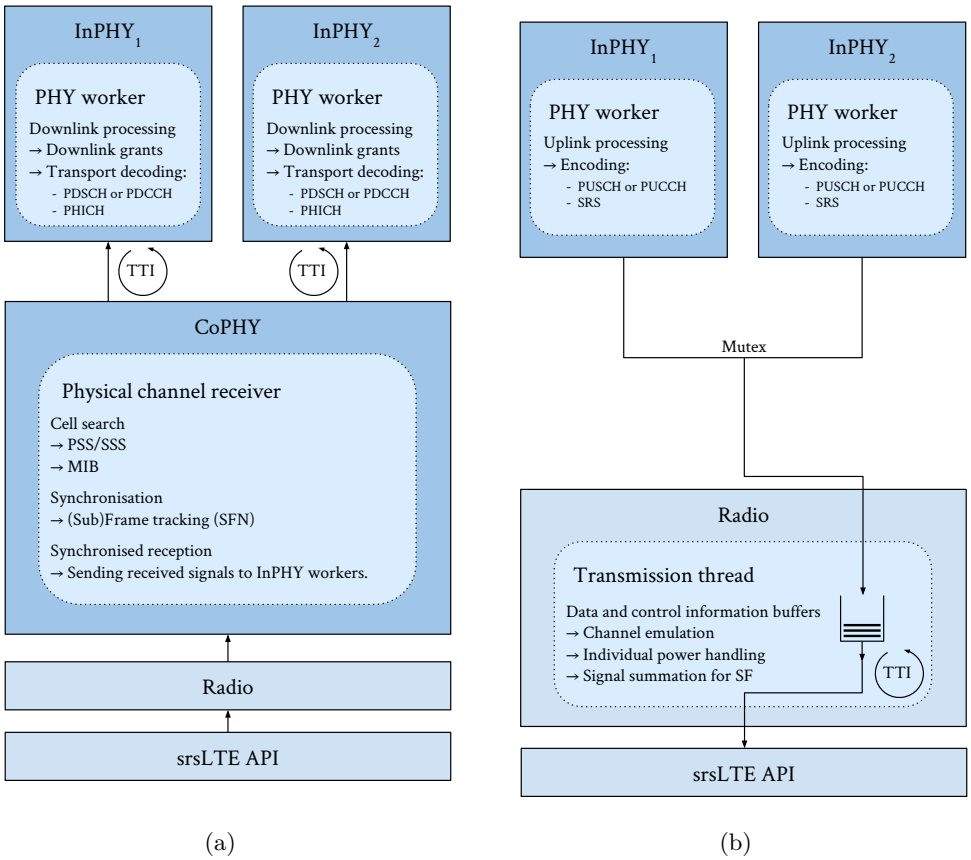


FIGURE 4.2: *Implementation of coPHY and inPHY for two MTC devices in down- and uplink, respectively. (a) Downlink. (b) Uplink.*

4.1.2 Radio layer modifications and uplink power control

Samples from those MTC devices who have data to send are combined in the radio layer, before the samples for the given subframe are scheduled for transmission in the SDR through the srsLTE API. The signals are scaled such that they are each sent with their desired power. The combined baseband signal, s'_{tx}, is then:

$$s'_{\text{tx}}(\tau) = \sum_{i=0}^{N_{\text{MTC}}-1} \frac{G_{\text{tx},i}}{G_{\text{tx,max}}} s_i(\tau) \,, \tag{4.1}$$

where τ is the discrete time samples, N_{MTC} is the amount of MTC devices, $G_{\text{tx},i}$ is the transmission gain of the ith MTC device, $G_{\text{tx,max}}$ is $\max(G_{\text{tx},0}, G_{\text{tx},1}, \ldots, G_{\text{tx},N_{\text{MTC}}-1})$, and s_i is the baseband transmission signals from the ith MTC device.

The combination of signals results in higher powers and amplitude. The maximum amplitude of the signal from one MTC device (amplitude of the baseband complex float handled in the radio layer) is varying between approximately 0.6 and 1.2; for the combination of signals, both maximum amplitude and average power scales with amount of devices, illustrated in figure 4.3. To avoid clipping in the power amplifier, the combined signal, i.e. $s'_{\text{tx}}(\tau)$, is normalised if the maximum peak of the combined signal exceeds 1.2, and the total baseband signal to be transmitted is then:

$$s_{\text{tx}}(\tau) = \frac{1}{\sigma} s'_{\text{tx}}(\tau) \,, \tag{4.2}$$

where $\sigma = \frac{\max_\tau \left(s'_{\text{tx}}(\tau)\right)}{1.1}$ is the scaling factor, which then results in a peak normalisation to 1.1 ($\sigma = 1$ when no scaling is performed). The transmission gain in the SDR is then set to $G_{\text{tx}} = \sigma G_{\text{tx,max}}$. In the implementation, some of the operations are carried out in logarithmic domain (decibel).

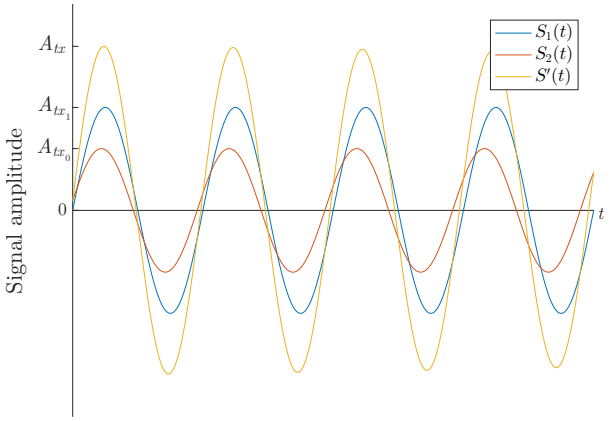


Figure 4.3: *Illustration of the signal summation and associated amplitudes.*

As previously stated, two buffers are introduced in the radio layer: the transmission control buffer and the transmission samples buffer. The transmission samples buffer contains the individual signal samples for the subframe from the UEs, i.e. s_i in equation (4.1); the buffer size is dependent on the bandwidth of the cell, so the initiation and allocation are not carried out until after cell search. The transmission control buffer contains both the amount of samples as well as timing information for transmission.

Also, two mutexes are implemented: a "radio access" and a "transmission" mutex. A mutex, short for "mutual exclusion", is a method that enables halting a thread, such that it does not enter a critical parts of code, e.g. access to memory shared between threads. The radio access mutex is used when accessing the radio layer, e.g. for setting the individual transmission power or for getting received signal strength indicator (RSSI) values. The transmission mutex is used by the inPHYs in order to be able to write to their respective transmission buffers, which are allocated in the same array. In addition to the mutexes, a conditional variable is implemented which signals the radio layer that the radio layer is to carry out transmission processing. Flags are used for each MTC device to signal whether they have finished uplink processing for the given subframe.

4.2 Uplink power budget

As previously stated, this thesis is utilising USRP B210 boards developed by Ettus Research. In order to emulate individual distances from the eNodeB to the MTC devices, path loss has to be implemented. There is however a limit as to how many individual signals can be transmitted and their relative power levels due to the physical limitations of the radio front-end of the SDR; this is subject to investigation in this section.

The output of the SDR is the individual signals combined at the same point in time and place – the individual signals from the MTC devices will therefore theoretically have equal power as they generally aim for the same received power at the eNodeB; there will however still be some difference in

the signal amplitude of the individual signals due to imperfections in LTE power control and frequency dependent noise estimations. Additionally, the reference received power is not the same on all physical channels, e.g. PRACH and PUSCH does usually not have the same received power, so differences will occur in output power between devices. Path loss for the individual MTC device is subtracted from their gain levels in the inPHY; this gain is then used in the radio layer when combining the signals and setting the total gain of the SDR for a given subframe. This is discussed in section 4.1.2. It is then up to the inPHY LTE power control to make up for the difference in output power (by tuning its set gain for the SDR higher) so that the received power at the eNodeB is close to the reference/aimed power.

This implementation setup mitigates the so-called "near-far" problem, where the DAC cannot produce the small changes in a low power signal when the dynamic range is used to generate a high power signal simultaneously. However, the problem is not completely eliminated – an analysis of what performance can be expected of the USRP B210 boards are therefore made.

The transmit gain of the B210 board is configurable, and can be set anywhere between 0 dB to 89.8 dB in steps of 0.2 dB by controlling a programmable gain amplifier (PGA) using the UHD [41]. The B210 board is equipped with a 12-bit DAC [42]. Output transmit power varies between −70 dBm and 13 dBm at 2.65 GHz using the whole gain spectrum, the relation between gain and transmit power scales almost linearly [43].

It must first be determined how many quantisation values are required to accurately describe a signal, in order to determine the maximum number of MTC devices which can transmit at the same time without clipping the signal on the radio front-end; figure 4.4(a) shows the root mean square error (RMSE) between a sine wave created using a range of N uniformly distributed quantizer levels up to the entire range of the 12 bit DAC ($2^{12} = 4096$ levels). Figure 4.4(b) shows examples of the decimation process to display the quantisation effect.

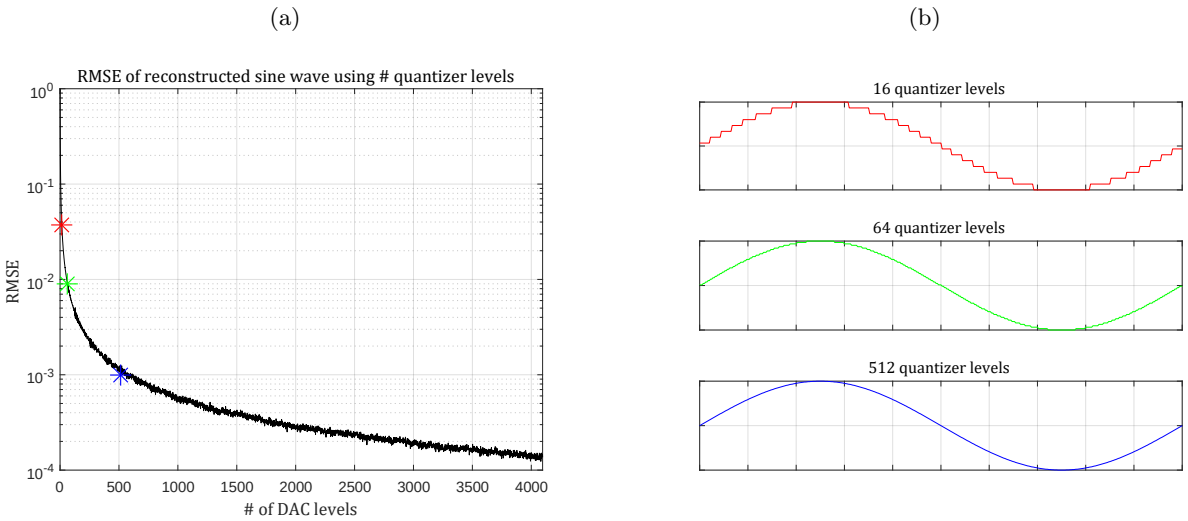


FIGURE 4.4: *Decimation factor impact on RMSE for a sine wave. (b) shows examples of the resulting output for different amount of quantizer levels, marked on (a).*

A threshold RMSE of 10^{-2} is chosen as an acceptable error, this corresponds to a sine wave created with a minimum of 57 quantizer levels. In order to allow for some headroom, as well as having an integer divisor of 2^{12} for simplicity, 64 is chosen as required number of quantizer levels for signal generation. Dividing the total number of quantizer levels with the now set requirement of 64 yields the maximum number of MTC devices that are allowed to transmit at the same time:

$$MTC_{\max} = \frac{4096}{64} = 64\,. \tag{4.3}$$

Note that this assumes that all MTC devices have identical transmit power. This is in other words the maximum number of MTC devices which can transmit in the same subframe and can be considered "best-case".

In order to determine the "worst case", e.g. the case where the least amount of devices can transmit in the same subframe, a case consisting of two MTC devices is considered, where the MTC devices will not generate the same output of the SDR. According to the LTE standard, the minimum transmit power of a device is −40 dBm [44]. "Worst-case" is then when the signal from the MTC device far away from the eNodeB, transmitting at maximum power (23 dBm for device category 0 [7]), arrives at the eNodeB with a very low power compared to the device close to the eNodeB. This is illustrated in figure 4.5. A maximum difference in path loss can then be defined, i.e. a kind of "maximum emulated distance" between the two devices.

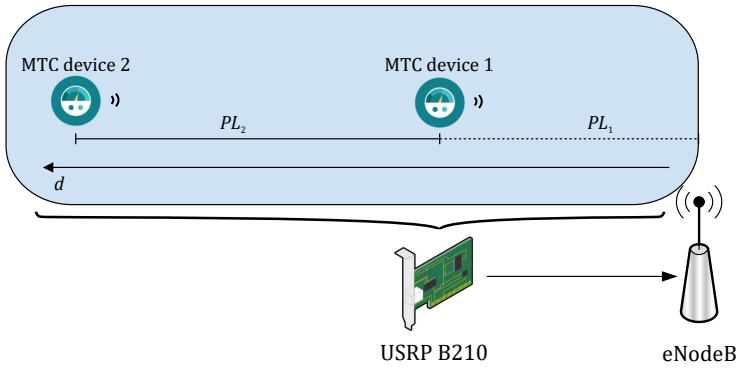


Figure 4.5: *Illustration of path loss in the system, PL_2 is the emulated path loss and PL_1 is the signal path loss due to physical attenuators and not emulated by the implementation.*

Recall that with the accepted level of quantisation error (arbitrarily chosen to 10^{-2} RMSE for a sine wave in order to have a reference), at least 64 levels should be used to generate the amplitude span. This then sets a lower limit of the output power relative to the high power signal of the closest device.

The output of the SDR is set to be at the device closest to the eNodeB, henceforth called device one. The "far away" device is called device two. Any desired path loss from device one to the eNodeB should be emulated using physical attenuators at the conducted connection. The output signal of the SDR is, therefore:

$$s_{\text{SDR}}(t) = P_{\text{tx},1}s_1(t) + (P_{\text{tx},2} - PL_2)s_2(t)\,, \tag{4.4}$$

where $P_{\text{tx},1}$ and $P_{\text{tx},2}$ are the individual set output powers with total output power $P_{\text{tx}} \approx P_{\text{tx},1} + P_{\text{tx},2} - PL_2$, PL_2 is the path loss from device two to device one, and $s_1(t)$ and $s_2(t)$ are the normalised signals of device one and two, respectively.

Now, at least 64 quantizer levels should be within the amplitude span of device two signal after path loss. The power of device two after path loss should then not subceed the equivalent of 64 least significant bit (LSB) flips, and the following relation is thereby obtained:

$$P_{\text{tx},2} - PL_2 \geq \frac{64}{4096-64} P_{\text{tx},1} \,, \tag{4.5}$$

where $P_{\text{tx},2} - PL_2$ is the SDR output for device two. Equation (4.5) then translates into:

$$PL_2 \leq P_{\text{tx},2} - \frac{64}{4032} P_{\text{tx},1} \,. \tag{4.6}$$

In the likely case that the two devices are transmitting with maximum and minimum specified power, respectively, this is then equivalent to:

$$PL_2 \leq 23\,\text{dBm} - (-40\,\text{dBm} - 17.99\,\text{dB}) \approx 81\,\text{dB} \,. \tag{4.7}$$

This means that if there is more than 81 dB difference between the output power of the SDR set by the MTC devices, the low signal power device will suffer higher quantisation noise than the accepted threshold. What 81 dB path loss relates to in relative emulated distance depends on the attenuation between the SDR and the eNodeB due to the power law nature of path loss.

Figure 4.6 shows the relation between physically added attenuation to the combined signals in dB and maximum distance according to Friis' free space path loss between the MTC devices in km. An attenuation of 60 dB corresponds to a distance of 10 m at 2.65 GHz using free space path loss, i.e. with the closest device being 10 m from the eNodeB the maximum cell size of 100 km [29] can be emulated. It should be noted that a signal of $23\,\text{dBm} - 60\,\text{dB} - 81\,\text{dB} = -118\,\text{dBm}$ will however not be decodable by the OpenAirInterface emulating the eNodeB in the test bed.

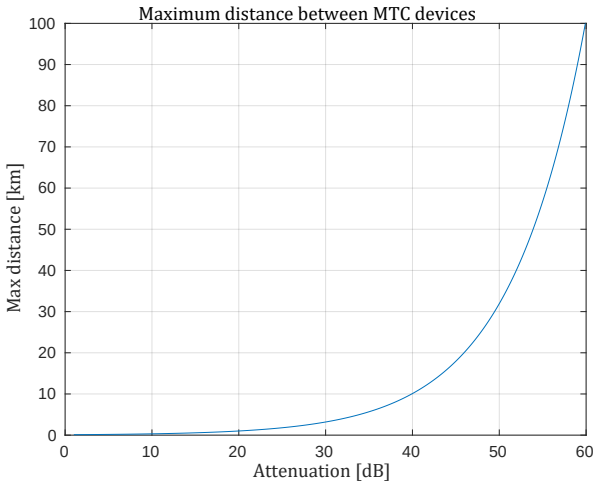


FIGURE 4.6: *Maximum possible distance between MTC devices as a function of attenuation of the radio link assuming free space path loss.*

The calculations performed in equation (4.7) are based on the minimum power requirement for an LTE MTC device, it is however possible to lower the power further as the SDR board is capable of transmitting with a power of −70 dBm [43]. By that, 30 dB attenuation can be added by gain adjusting the SDR instead of adding physical attenuators.

It should be noted that these cases describe the two limiting cases, and cannot be applied to any arbitrary configuration of MTC devices.

4.3 Downlink path loss emulation

To make the multiple devices experience individual channel conditions, some channel emulation is implemented in the downlink on the massiveMTC device. A simple attenuation of the sampled baseband signal will not change the SNR, because the attenuation of the information signal will be equal to that of the noise. Instead, the noise level must be increased relatively to the information signal level. In the massiveMTC implementation, a "static" decrease in the SNR is developed to emulate path loss. The downlink path loss emulation is implemented in the coPHY just before passing the signals to the inPHYs; this means that the CFO and SFO estimation, as well as PSS and SSS tracking are the same between devices, and made directly on the sampled signal (without any channel emulation). Placing the channel emulation before the processing in the coPHY would prove impractical, as these processes are by design carried out on behalf of all the devices. Alternatively, it could be placed in the inPHYs, which would yield the same result (if placed in the beginning of the processing); the reasoning behind choosing the coPHY is that it keeps the inPHYs "unaware" of the emulation, and it makes it easier to manage the given path losses, as they are used in only one place and therefore the implementation does not require further interfacing.

To obtain the decrease in the SNR level, channel estimation of the noise level (or the SNR itself, which generally requires noise level estimation as well) must be performed, otherwise noise cannot be added at a known level relative to the signal and the reduction in SNR will be unknown. As the channel emulation is only performed when synchronised (as cell search and synchronisation is performed by the coPHY), estimation can be performed using the downlink RSs. The channel estimation is also carried out in the inPHYs to be able to do CQI reporting, however only for their allocated RB depending on CQI reporting mode set by eNodeB. The noise estimation is based on subtracting the known RS form from the received PSS.

Noise is generated as follows:

$$N = \kappa \sqrt{\frac{P_{PL} P_N}{2}}\, N_{IQ}\,, \qquad N_{IQ} = N_I + jN_Q\,, \tag{4.8}$$

where P_{PL} is the "path loss attenuation" (emulated as an increase in noise level, but results in the same result with regard to SNR), P_N is the current noise level, and $N_I, N_Q \overset{i.i.d.}{\sim} \mathcal{N}(0,1)$. The κ is the digital scaling factor used to take the receiver gain into account, as it is the ratio between the received power and the "digital power" of the baseband signal.

The digital scaling factor is estimated using the difference between the RSSI value estimated in the baseband processing in the masiveMTC implementation and the on-board RSSI estimate found using the B210 ADC. This is a method also applied by srsUE in other functions; however, it has been found not to yield perfect results and requires offset scaling that needs to be calibrated.

The generated noise is then added to the samples and passed to the inPHY; in this way, the different devices will then be experiencing different channels for decoding downlink control information and data, as well as for CQI reporting. The noise samples, i.e. N_{IQ}, are pre-generated when initiating the software, as real-time generation has proven too slow. A large array is therefore pre-generated in the memory, so that the random number generation is not required in the time sensitive process.

As a final remark, it should however be noted that while downlink channel emulation is possible, the implementation is not reliant on individual downlink channels. Meaning that regardless of the measured channel, each MTC device is able to report a predetermined CQI to the eNodeB. This implementation strategy is based on that if it is the objective to test the load on a network, little is gained by adding additional path loss in downlink, which adds calculation heavy processes in the time critical parts of the software.

4.4 Scalability

To expand the implementation in order to handle more devices, several tweaks are required; most notably additional USIM details are incorporated in the OpenAirInterface HSS and in the massiveMTC device, more advanced device initiation processes are developed and optimised, as well as additional signalling for less dependency between individual devices. The latter is a key feature to prevent that the collective performance is not severely affected by the performance of the individual devices, i.e. the performance should not be limited by the slowest device. This is especially relevant in uplink processing, where the transmission should be executed when the deadline for transmission is reached, even if all devices have not finished processing; in this case, the packets from the late device(s) will simply be discarded. This is however preferable to the alternative, where all packet transmissions are delayed (and probably discarded by the eNodeB).

A detailed illustration of the down- and uplink processing in, and signalling between, the radio layer and physical layer is displayed in figure 4.7. The transmission thread is signalled to start once every subframe, either by the last inPHY that is ready or by the coPHY when reaching the transmission deadline. Because of this, the coPHY and radio are always synchronised, and thereby transmission (i.e. uplink) is always synchronous with the downlink, as is required for LTE. This provides resilience to errors or other events slowing individual devices down.

The function calls performed in the inPHY and radio layer is shown in figure 4.8. For each TTI (subframe), the worker in the inPHY will signal the radio layer through function calls in the general PHY module: it will either write the transmission data (I/Q samples in complex float format) and appropriate signals/metadata (`tx_buffer()`) or simply signal that no data is to be transmitted (`nothing_to_send()`).

The two function calls for signalling the radio are essentially equal in their structure, which is why they are collectively described in figure 4.8. When nothing is to be sent, time information for the given TTI is written to the transmission control buffer and the flag signalling that uplink processing has finished is set. When transmission is to occur, the same functions are carried out and in addition to this, the signal samples are written to the transmission buffer; the number of samples written to the transmission samples buffer is indicated in the information written to the transmission control buffer. Both procedures will finish by checking whether all devices have finished processing, in which case uplink transmission process in the radio layer will be signalled through the conditional variable.

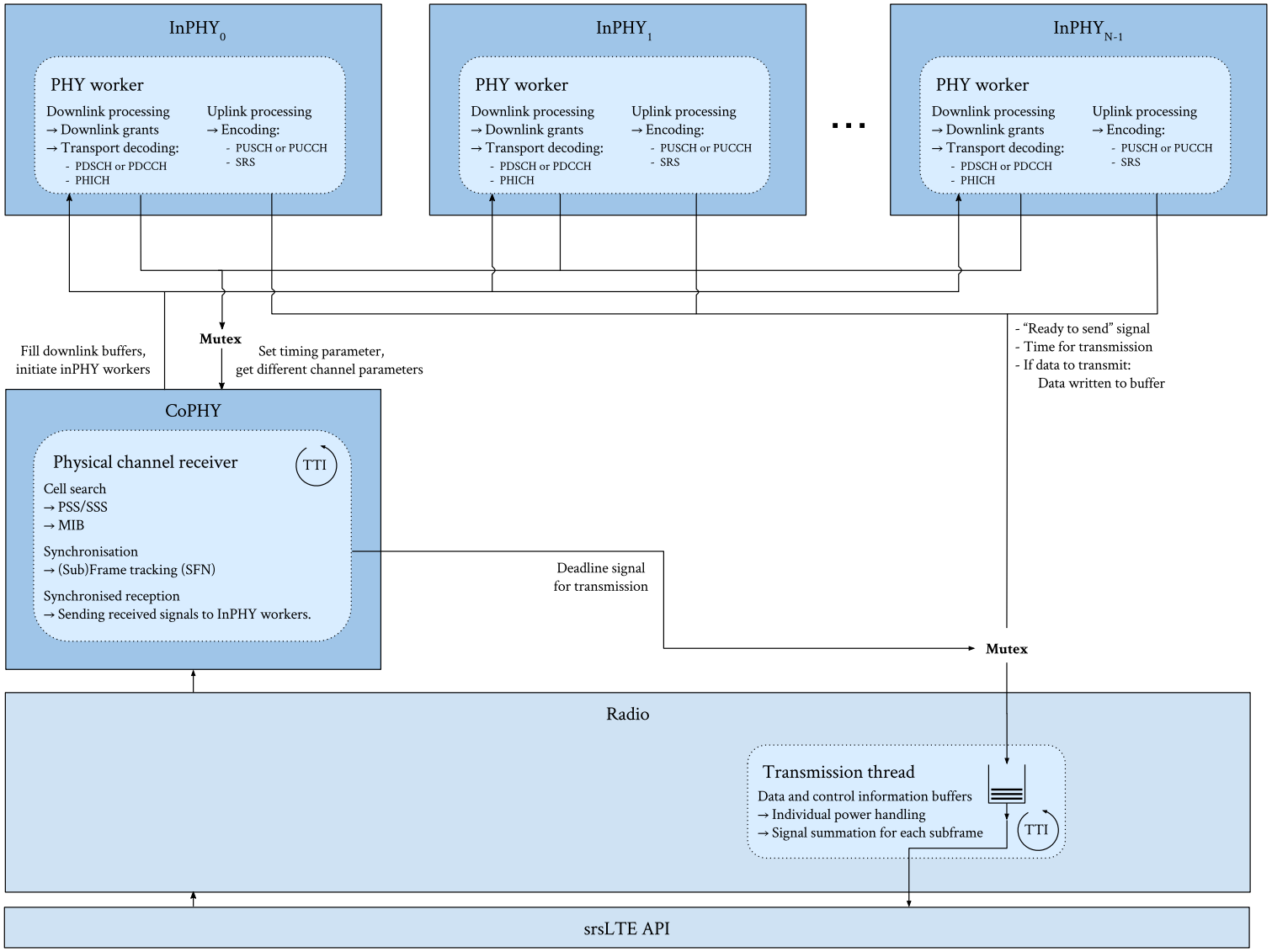


Figure 4.7: *Elaborate illustration of down- and uplink processing and signalling on the radio and PHY layers.*

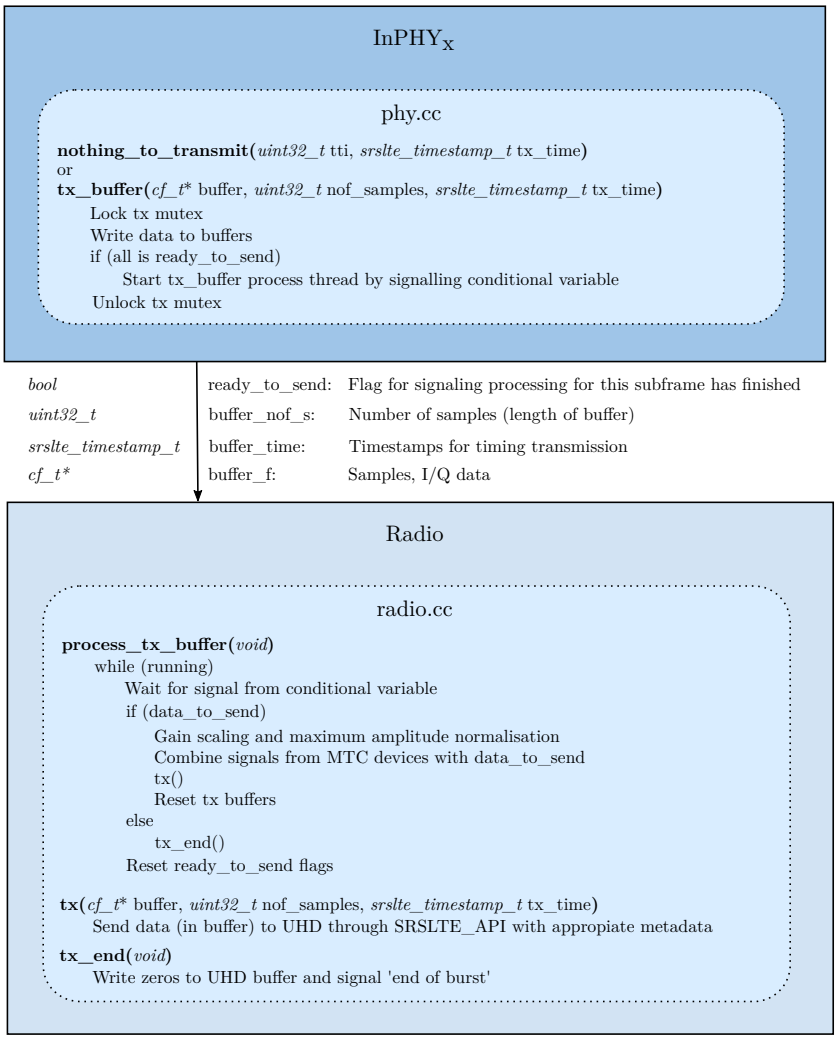


FIGURE 4.8: *InPHY uplink data and control signalling to radio layer.*

In the radio layer, the transmission process (`process_tx_buffer()`) will check whether any of the devices have data to be transmitted. If any, signals from the devices will be combined, power control as described in section 4.1.2 is carried out, before the combined signals are sent to the SDR using the `tx()` function call.

As previously discussed, and displayed in figure 4.7, additional signalling occurs from the coPHY when the deadline for transmission is reached. This function call uses the same mutex, but applies another function in the radio layer. The function will check, whether all devices have finished their processing, so that transmission for the active TTI has occurred or is occurring. If not, the transmission process will be initiated for the devices which have finished processing such that transmission will occur within the deadline.

4.5 Initiation: The device handler

The instantiation of the devices is handled from a `main` module, as is customary for C/C++ software. A configuration file with RF settings is loaded inside the `main` module, (such as downlink and uplink frequencies), SDR device arguments, output/logging settings, USIM data, as well as some "expert" settings (amongst others, specific PRACH transmission gain and number of PHY layer workers/threads). When each device is initiated it is registered in a *device handler*, which is developed for handling the multiple device instances as well as the coPHY. Handles to the instances and their individual layers are stored in the device handler, required handles is passed to the coPHY so that it can pass subframe data and other signalling to the devices.

The additional path losses for the individual device signals are set in the device handler; as per current development, it is hard-coded in the device handler software module, but it could be integrated in the configuration file.

Error messages from the radio layer, starting the coPHY when the software is finished initiating, as well as the termination of the device handles and coPHY when shutting down are also handled by the device handler.

4.6 Outputs of software

During the initiation of the software modules, messages are written to the console such that progress can be tracked. When initiated, only important events will be notified through the console, such as initial decoding of MIB and SIB1/2 messages. When MIB has been found, cell settings will be setup for each device, such that arrays can be allocated and workers instantiated; this is called cell initiation and messages for this are also written to the console. When synchronised, messages will be sent to the console upon PRACH transmission, as well as if the RAP is successful and RRC connected state is reached. If authentication goes well and the RRC reconfiguration complete message is received, the VNI, i.e. the TUN device, will be generated; messages for this and the given IP address will also be written to the console. Upon successful connection, only eventual synchronisation issues will be written to the console; in the event that connection re-establishment is required, additional PRACH transmissions and RRC connected messages (if successful) will be displayed.

Additional outputs can be written to a log file, the verbosity of the output is set through the configuration file. Debug messages are written so that software progress can be tracked; examples of these are messages describing synchronisation in coPHY, TBs between MAC and inPHY, when signals are sent to the transmission buffer in the radio layer, as well as transmission of baseband signals to the SDR. The log messages cannot be tracked in real time and is therefore a debugging tool. With the current development status, outputs must be written to the console if they are required in real time.

5. Performance evaluation

The massiveMTC prototype implementation is tested in order to evaluate the performance of the implementation. This chapter describes the test setups used before providing the data from the tests. All tests are carried out with the massiveMTC software running on a desktop PC. The specifications of the desktop PC, acting as the host PC, are shown in table 5.1.

TABLE 5.1: *MassiveMTC host PC specifications.*

PC	**Dell Optiplex 9020 SFF**
Distribution	Ubuntu 16.04 LTS
Kernel	4.4.0-67-lowlatency
CPU	Intel Core i7-4790 @ 3.60 GHz
Memory	24 GB 1600 MHz DDR3L RAM

In order to test the massiveMTC device implementation, an eNodeB (or eNodeB emulator) is required, as the code will otherwise not run due to the downlink synchronisation in the LTE protocol; all messages must also be handled correctly in order to verify successful attachment to a network. Three different eNodeB setups are used, each with their own advantages:

- Dedicated test bed using OpenAirInterface
- UXM wireless test set (E7515A)
- Nokia eNodeB at Telenor Aalborg test facility

In addition to these, successful attachment has been carried out on an Anite 9000 network simulator, but no further test has been conducted using this setup.

OpenAirInterface has been the most used platform, it being the eNodeB in the test bed for the massiveMTC device development. More information about OpenAirInterface setup is given in section 3.2.1. It has high degree of reconfigurability, especially as specific functionality can be changed directly in the source code. It can however be difficult to modify the OpenAirInterface code due to the low transparency of the implementation. The OpenAirInterface eNodeB can handle up to three devices before becoming unreliable according to the developers; especially that the MAC scheduling is not optimised causes some issues for more than two devices. It proved difficult to connect to the OpenAirInterface EPC with more than two devices due to seldom success in either the authentication or the creation of a DRB. A major downside of the OpenAirInterface is the power control; it does not have strict control on the output power, both total and relative between physical channels, as it uses calibrated values from their own setup. A consequence of this is that downlink path loss estimation, and thereby uplink power control, does not work as intended and uplink power must therefore be manually set as a constant on the massiveMTC device. In addition, the OpenAirInterface EPC has not proven capable of correctly routing IP traffic, making data transfer tests impossible on this setup.

Through the collaboration with Keysight Laboratories in Aalborg, tests has been carried out on their E7515A UXM wireless test set, which is a signalling test device designed for RF design validation of LTE devices by emulating an eNodeB. The UXM is capable of changing parameters using a GUI which makes it possible to test use cases within the LTE protocol, as well as dissecting protocol messages for debugging. It is however only capable of testing one UE at a time (on the same cell). A picture of the UXM and massiveMTC during testing is shown in figure 5.1. The UXM has an internal EPC emulator, but also supports connecting to an external EPC. Tests has been conducted both for the internal EPC emulator as well as with an external emulator, namely the Polaris EPC emulator, also supplied by Keysight Laboratories.

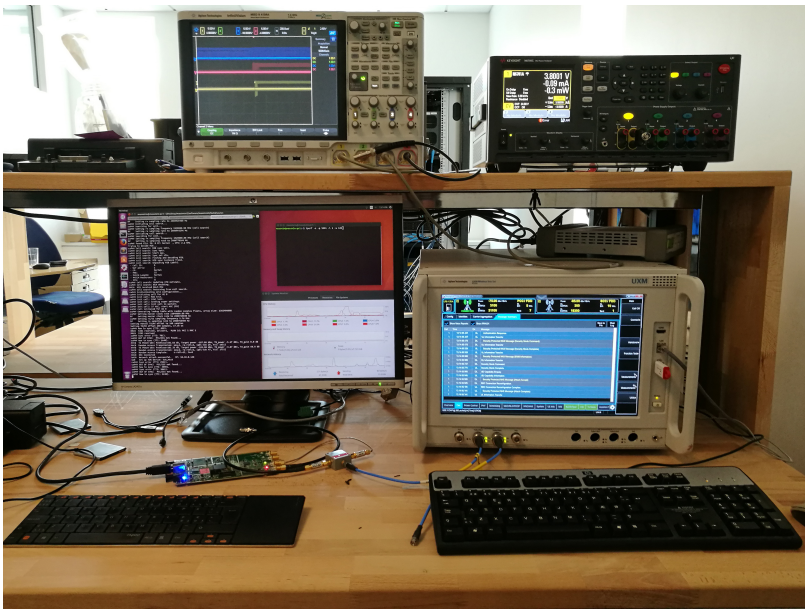


Figure 5.1: *The massiveMTC device host PC and B210 board (lower left) connected to the UXM (lower right) at Keysight Laboratories in Aalborg. The oscilloscope and power supply is not used during massiveMTC device testing.*

The mobile operator Telenor granted access to test the massiveMTC device on a commercial Nokia eNodeB at their testing facility in Aalborg. This being a commercial eNodeB allows for a greater number of devices to be connected at the same time. In the test facility, the test eNodeB is connected to the live Telenor EPC, which makes it impossible to get authenticated to the network without a real SIM card. Unfortunately, this means that the implementation is not able to successfully connect to the network as physical SIM information is not supported and injecting false USIM data into Telenor's live HSS is not viable. Tests are instead carried out on the Nokia eNodeB connected to the Polaris EPC emulator provided by Keysight Laboratories; connection using this EPC did however also prove unsuccessful, as the NAS authentication request message (shown in figure 2.10) issued by the MME within the Polaris EPC is not detected. It is observed that with the Polaris EPC, the attachment procedure consistently reaches the authentication request step, meaning that the massiveMTC devices reach the RRC connected state and issue the RRC connection setup complete message, including the NAS attach request message for the MME, which is received correctly at the MME. The response (authentication request message) is however lost somewhere between the MME and massiveMTC device implementation.

All tests in the following sections are carried out in band seven in LTE, which corresponds to 2.6 GHz. Tests for support of the bandwidths in LTE are shown in table 5.2.

TABLE 5.2: *Test of supported bandwidts of massiveMTC prototype implementation.*

	OAI	**UXM**	**Nokia**
1.4 MHz	*N/A*	**x***	*N/A*
3 MHz	*N/A*	-	*N/A*
5 MHz	✓	✓	✓
10 MHz	✓	✓	✓
15 MHz	✓	✓	✓
20 MHz	(✓)	(✓)	(✓)

*Cell search and synchronisation working, issue observed with RAP MSG3.

✓	Working.
(✓)	Working, but unstable.
x	Not working.
N/A	Not supported by eNodeB.
-	Not tested.

The Nokia eNodeB at Telenor does not support bandwidths lower than 5 MHz, OpenAirInterface claims that it supports all bandwidths; it is however not working at 1.4 nor 3 MHz on the test bed. 1.4 MHz is tested on the UXM, where neither the original srsUE nor massiveMTC device are able to connect due to some error in the decoding of the MSG3 of the ARP.

5.1 PRACH detection analysis

As only a limited amount of simultaneously attached devices are supported by OpenAirInterface, evaluation of the attachment procedure of many devices are therefore not possible on this platform. Instead, only the preamble detection is tested to prove that several device instances are generated, and that these are able to transmit in the same RAO. OpenAirInterface, in its original version, will only detect one preamble (above a given threshold), as it only supports starting one attachment procedure per RAO; the preamble detection module will therefore by default only return the preamble with the highest detected energy. However, to test the preamble detection and not the full attachment procedure, this is modified such that all preamble above the detection threshold are reported, so that preamble transmission and signal addition is verified.

The detection rate of the preambles are displayed in table 5.3, where up to ten devices transmit in the same RAO and the subsequent detection is noted on the output of the OpenAirInterface host PC for 30 RAOs. The massiveMTC is transmitting with a constant gain of 55. The test is performed with PRACH configuration index 0 having a root sequence index of 0, zero correlation zone configuration value 11, and frequency offset parameter value 2; these are the default settings in OpenAirInterface.

TABLE 5.3: *Detection of preambles using OpenAirInterface eNodeB.*

MTC devices	**Detection rate**
1	100 %
2	98.3 %
3	98.8 %
4	99.1 %
5	96.6 %
10	96.4 %

Detection performance is seen to be excellent also for many devices, which is in accordance with that the conducted channel conditions are good, even though 30 dB attenuation is used on the signal (as described in chapter 3). Generally, no false positives are observed, though such has been observed during the development and generally when the PRACH transmission power is increased from the reference received power, as will be discussed in section 5.3.

It should be noted that it is difficult to use the OpenAirInterface setup to investigate the preamble detection rate in a realistic scenario, as the detection threshold within is empirically set and does not by default take estimated delay into account as part of the threshold. Further detailed investigation with regard to other power levels is not performed, as the actual output power of the B210 board is not calibrated as well as with the current development status, the testing sweep is not automated, making the process slow and cumbersome.

It is however observed that the detection is independent of whether the transmitted preamble indices are in sequence, meaning that it does not seem to prompt for the signals to result in neither additional missed detections nor false positives if e.g. preambles with indices 0, 1, and 2 are transmitted.

5.2 CPU load analysis

The tests conducted at the Telenor test facilities involve connecting an increasing number of MTC devices to a network in order to measure the CPU load while the devices are connected and in synchronisation with a Nokia eNodeB, in order to establish an idea of how many devices can be connected at the same time. The CPU load has been measured using the Ubuntu system monitor. As a result of the issues with the EPC at the Telenor test facility, the tests performed shows the CPU load of the host PC when the devices are connected and reaching RRC connected; when the authentication request is not received, the devices stay synchronised until they are released by the Nokia eNodeB. Tests are conducted with the Nokia eNodeB running at 5 and 10 MHz bandwidth. It should be noted that during all tests, the initial spike in CPU load is due to initialisation of software, followed by a spike for cell search and initial synchronisation as displayed in figure 5.2(a).

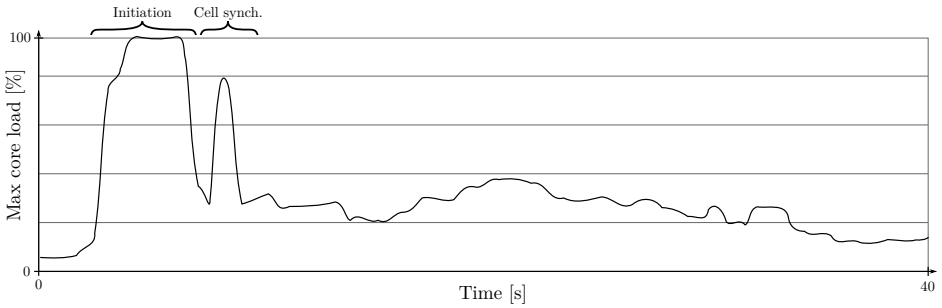
The average CPU load of the host PC observed during the tests conducted both using the Nokia eNodeB and OpenAirInterface is shown in table 5.4; envelope of maximum core load is illustrated in figure 5.2. Additional test results can be found in appendix E.

Table 5.4: *Average CPU load (in percentages) with N MTC devices using Nokia and OpenAirInterface (OAI) eNodeB.*

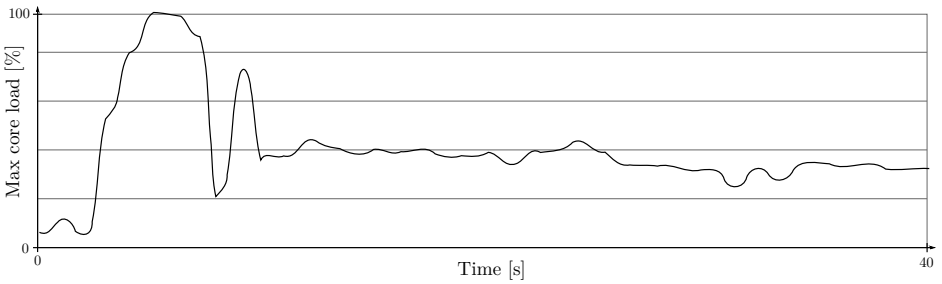
	Nokia		**OAI**	
N	5 MHz	10 MHz	5 MHz	10 MHz
1		5	8	11
2		10	14	16
4	17	11		
6	15	20		
8	20	48		
10	35	55		
12	42			
14	52			
15	57			

The CPU load generally increases with the number of connected devices as expected. Maximum memory usage is registered just above 20 %, corresponding to 4.8 GB for the case where 15 MTC devices is connected. The average core load of the CPU increases with around 10 % for every five MTC devices. It should be noted that that performance for more than 15 MTC devices at 5 MHz proved unstable, resulting in synchronisation issues.

(a) 4 MTC devices.



(b) 10 MTC devices.



(c) 15 MTC devices.

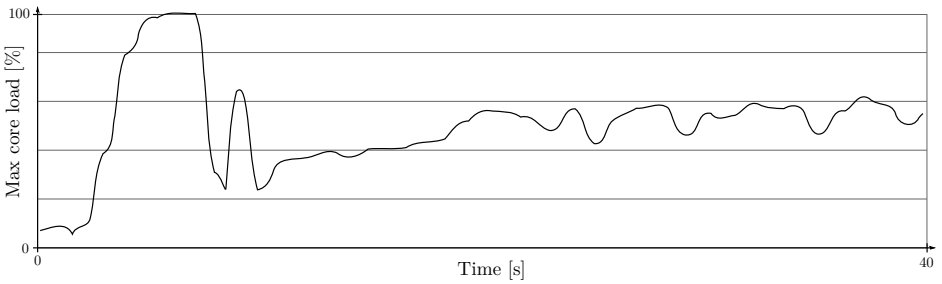


FIGURE 5.2: *Envelope of maximum CPU core load, i.e. the maximum load on any of the cores, for Nokia eNodeB tests at 5 MHz.*

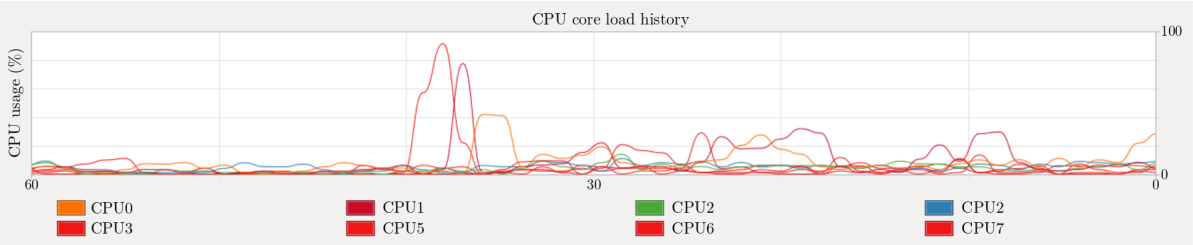
The required memory for eight MTC devices at 10 MHz case is also about 5 GB. More than eight MTC devices at 10 MHz proved unstable, similarly to when having more than 15 devices at 5 MHz.

The same test has been organised using the OpenAirInterface eNodeB and EPC, in order to investigate the influence on the CPU load if successful authentication is possible. Due to performance constraints of this platform, only up to two MTC devices could be connected at the same time.

It is seen that an increase in CPU usage per device is apparent, which is expected as all protocol layers are running. It seems that halving the number of MTC devices yields approximately the same CPU usage when comparing the Nokia eNodeB with OpenAirInterface.

CPU usage of srsUE and the massiveMTC implementation has been compared for a single user in order to investigate the effect of the added overhead, this can be seen in figure 5.3. MassiveMTC has similar CPU usage for one device compared to srsUE. This shows that the added multi-thread overhead has negligible effect on the performance. It is by definition not possible to test the performance of srsUE with more than one UE on the same device, at least not using the same SDR. However, making parallel srsUE instances running using separate SDR boards should result in a CPU load increase equal to the load of a single device.

(a) 1 srsUE UE device connected to OpenAirInterface eNodeB at 5 MHz.



(b) 1 device in massiveMTC connected to OpenAirInterface eNodeB at 5 MHz.

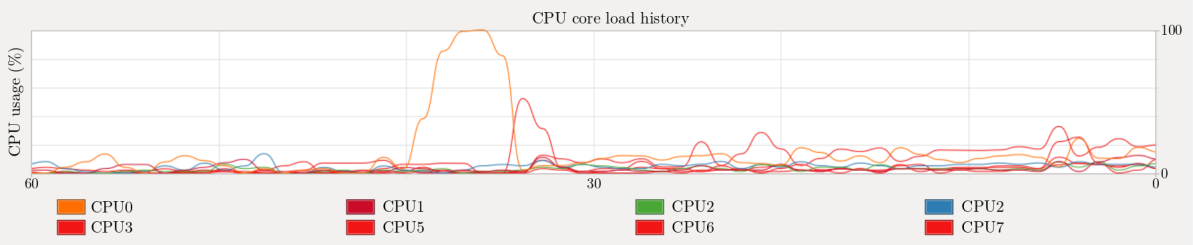


Figure 5.3: *Comparison of CPU load at 5 MHz.*

Table 5.5 summarises the maximum amount of MTC devices which are able to connect and maintaining a stable connection.

Table 5.5: *Maximum number of stable MTC devices.*

Bandwidth	Nokia	OAI
5 MHz	15	2
10 MHz	8	2

5.3 Power control

Uplink: Preamble power

The received power of preambles sent by multiple devices in the same RAO is compared, in order to test whether the output signal scaling of the uplink power control described in section 4.1.2. The results are displayed in table 5.6. The preambles sent are chosen randomly, causing occasional duplicates/collisions; the received powers of colliding preambles are not taken into account.

The estimated received power measurement by OpenAirInterface is performed using an arbitrary logarithmic scale, so it cannot be translated directly into received power in dBm; however, a change in the expected direction of the received power is observed.

TABLE 5.6: *Preamble power detection trials using OpenAirInterface, with estimated power of preamble detections. Values are in a arbitrary logarithmic scale used by OpenAirInterface, the gain value is the transmission gain as set by the individual devices, μ is the mean for all the trial runs.*

		Trials													
	Gain	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	
MTC0	57	76.7	*dup*	76.7	76.7	76.7	72.0	*dup*	76.7	76.7	76.7	60.0	*dup*	75.0	
MTC1	55	70.8	*dup*	71.6	72.0	72.0	70.5	72.0	70.9	70.6	71.6	55.7	*dup*	71.7	
MTC2	53	67.2	68.5	67.1	68.3	55.7	65.7	66.5	70.0	66.4	66.9	57.0	66.5	69.7	
MTC3	51	66.6	65.9	65.4	64.2	*nd*	62.3	61.0	64.6	63.7	63.9	*nd*	57.0	64.1	
MTC4	49	63.0	63.9	62.5	61.6	56.4	58.4	*dup*	60.0	63.4	61.6	60.5	61.4	58.4	
MTC5	47	57.0	*nd*	60.5	56.4	72.0	56.4	60.7	58.4	58.7	58.7	*nd*	59.4	55.7	
Frame		290	294	298	302	352	356	360	364	368	372	406	410	414	
		Trials													
	Gain	**14**	**15**	**16**	**17**	**18**	**19**	**20**	**21**	**22**	**23**	**24**	**25**	**26**	$\boldsymbol{\mu}$
MTC0	57	75.0	75.0	75.0	76.7	75.0	75.0	75.0	75.0	*dup*	76.7	75.0	75.0	75.0	74.9
MTC1	55	72.0	71.4	71.5	72.0	72.0	72.0	72.0	71.5	*dup*	72.0	72.0	72.0	71.9	70.9
MTC2	53	69.4	67.6	67.4	68.8	68.5	71.0	68.2	67.0	68.1	68.2	67.3	69.4	67.7	67.1
MTC3	51	64.3	63.5	63.3	65.4	63.0	66.5	63.0	65.7	65.5	65.6	64.9	63.9	64.5	64.1
MTC4	49	59.4	57.0	58.4	62.4	58.0	60.7	72.0	59.7	62.7	63.1	61.9	60.5	61.4	61.1
MTC5	47	55.7	58.4	57.0	56.4	55.7	71.0	58.4	57.0	58.7	*nd*	*nd*	57.0	*nd*	59.0
Frame		418	422	430	434	438	442	446	450	454	458	462	466	470	

nd Not detected by OpenAirInterface.
dup Duplicate/collision, i.e. same preamble is sent multiple times in the same RAO.

Having a received power up to 10 dB above the reference received power results in a large number of false positives detected by the OpenAirInterface. These are ignored in the analysis.

It is seen that the received powers are individual for each device, displaying that the output signal scaling works. The estimated difference between devices is approximately 2 dB to 4 dB, where the gain difference between them is set to 2 dB on the transmitter side.

Downlink: CQI analysis

In order to test the implemented downlink power control, the CQI is analysed. The CQI reporting in srsUE, which is also the one used in the massiveMTC device, is based on the work in [35]. The CQI is implemented as a look-up table based on the SNR in order to ensure that the current AMC guarantees a transport block error rate (BLER) below 10 %. The paper found that during single input single output (SISO) transmission, an approximate 10 dB increase in SNR corresponds to an increase of 5 in CQI value.

The test is performed on the UXM at 5 MHz with cell power −60 dBm and reference signal power (in the SIB2 message) at 18 dBm, resulting in a simulated path loss of 78 dB. These values are chosen such that a high CQI will be reported in the case of no additional path loss, this can be seen in figure 5.4. An additional 5 dB path loss is then added until the reported CQI is close to 1. The reported CQI for all cases is displayed in figure 5.4 and the average CQI can be found in table 5.7.

Table 5.7: *Average CQI with added path loss.*

PL	Mean CQI
0 dB	14.66
5 dB	13.47
10 dB	12.13
15 dB	5.85
20 dB	4.12
25 dB	1.76

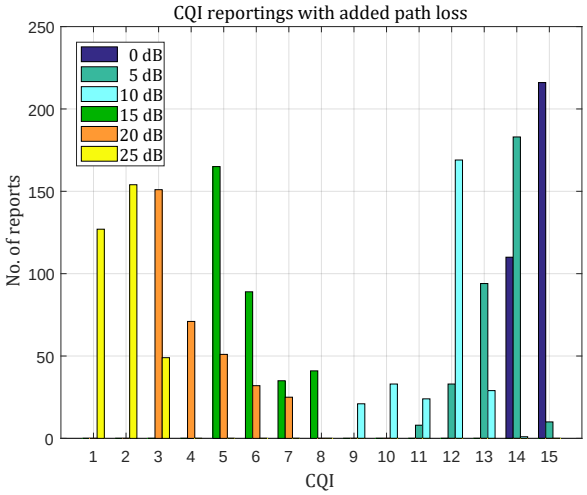


Figure 5.4: *CQI at varying SNR levels measured over approximately 6.5 s, note that a 2 s settling time has been removed from the data in order to improve the noise estimation.*

It can be seen that while the CQI drops for each case, it is not the linear behaviour described in [35] where a 10 dB decrease in SNR yields a CQI value of 5, while the behaviour is satisfactory at the edges of the CQI-scale.

5.4 Data transfer

The ability to transfer data through one of the MTC devices created in the massiveMTC emulator has been verified, the throughput is measured using the UXM wireless test set. The measured IP and over-the-air (OTA) throughputs are shown in figure 5.5 using massiveMTC and the original version of srsUE. The data transfer is generated using the `iperf` tool, where the MTC device acts as server and the UXM acts as client. The server is set up as shown in listing 5.1.

Listing 5.1: *Server initiation command.*

```
1    $ iperf -s -p 5001 -i 1 -w 64k
```

The `-s` flag sets up the server side, `-p` is used to set the port number, `-i` sets reporting interval, and `-w` sets the TCP window size. The client is set up as shown in listing 5.2, where `-c` sets it up as client, which is followed by the IP of the server. The `-t` flag is used to set transmit time in seconds and `-r` reverses the roles of client and server after t seconds.

LISTING 5.2: *Client initiation command.*

```
$ iperf -c 192.168.3.128 -t 10 -r
```

First, the eNodeB acts as a client while the MTC device is the server, so that downlink traffic throughput is tested. After 10 s the direction is reversed, and data is then transmitted in uplink. MCS setting is set to 8 for both down- and uplink, as this proved to perform the best on the UXM; this corresponds to QPSK modulation and a transport block size of 3496 bit per subframe.

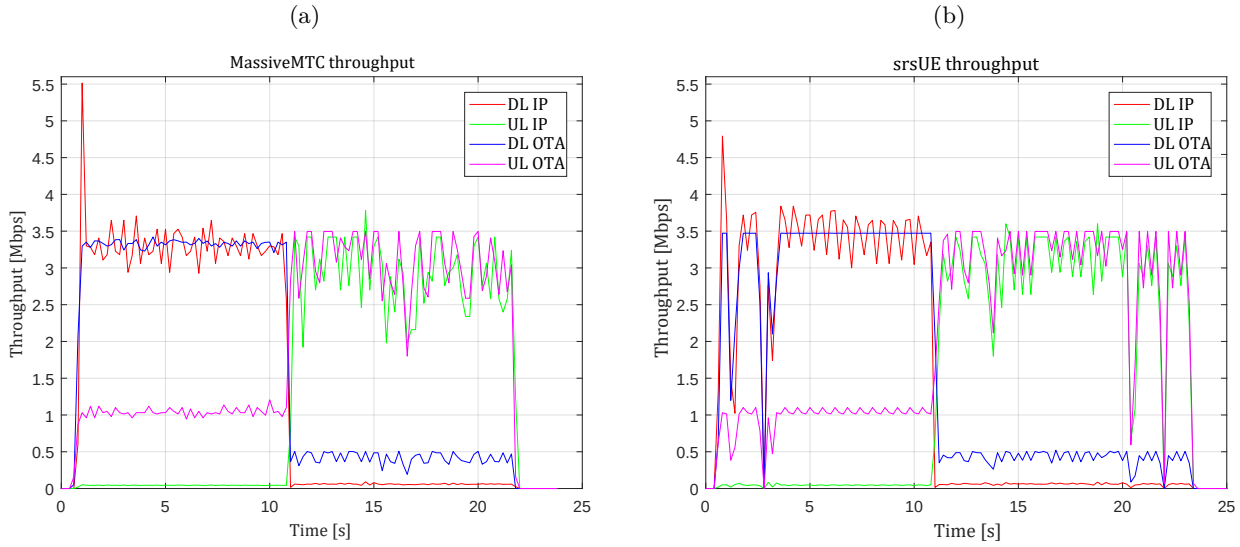


FIGURE 5.5: *IP and OTA throughput at 5 MHz for both up- and downlink with MCS set to 8.*

It is seen that throughput using massiveMTC is close to unchanged in both up- and downlink compared to srsUE, and that the maximum data rate for both implementations is approximately 3.3 Mbps. This corresponds well with the theoretical maximum throughput of 3.496 Mb/s at 5 MHz when the MCS is 8[1].

It is apparent that drops in the throughput is present in the downlink case of srsUE in figure 5.5(b), this is due to the device losing synchronisation with the cell, and that it therefore has to find PSS again; this also occurs in the massiveMTC implementation. For higher transport sizes it is seen that the implementation (srsUE and inherently also massiveMTC) has trouble keeping up the throughput, when it has to use 16QAM.

[1]Theoretical throughput is calculated by finding the TBS index and size using table 7.1.7.1-1 and 7.1.7.2.1-1 in [36].

6. Conclusion

Given the recent and continuing rise in MTC devices due to high growth in the popularity of IoT, new communication patterns are introduced to wide-area networks featuring packets with smaller payloads from a vast amount of subscribers. Actions towards improving the existing networks are in development in order to be able to accommodate the larger amount of active devices on the network. This is also the case for cellular networks, where operators want to compete with dedicated LPWAN systems. New protocols which are able to coexist with LTE, such as NB-IoT and LTE-M, emerged in Release 13 to accommodate these new demands. Implementations for deployment are in development, currently being in the pilot project phase.

A design principle and a prototype are sought to be developed in order to evaluate the potential improvements that the new MTC protocols bring, and to evaluate bottlenecks in the core network when massive amounts of devices are accessing the network. This new device, called the massiveMTC device, is capable of emulating multiple devices on the network, such that eNodeB and core network can be evaluated, as well as new features, to determine whether they can sustain the desired amount of devices in a cell.

By emulating the entire protocol stack, the most realistic evaluation can be obtained, as the functionality of the entire protocol stack must be carried out in real time. By using an SDR as radio front-end, testing up against commercial eNodeBs is possible, as there is no significant difference between signals from commercial devices and the SDR generated signals.

A first step towards developing the massiveMTC device is to investigate whether it is possible to create a software emulator where several MTC devices are able to communicate using the same RF front-end in order to keep the cost of the emulator as low as possible. An existing software implementation of a single Release 8 compliant device protocol stack has therefore been used to develop a platform providing multiple MTC devices that all share the same RF front-end. Although Release 8 is deprecated, it still serves as a strong proof-of-concept as the overall structure of the protocol remains the same for all releases as non-MTC specific elements of LTE are backwards compatible. The implementation also provides proof-of-concept for NB-IoT and LTE-M, because they are based on the same underlying principles, though not directly backwards compatible.

The design principle enables several instances of the LTE protocol stack to access the same radio layer by splitting the PHY layer of the LTE protocol into a common part for all devices, known as the common physical layer (coPHY), and an individual part for each MTC device, named individual physical layers (inPHYs). This makes it possible to do calculations that are equivalent for all devices in order to cut down complexity and lower usage of computational powers, thereby gaining scalability.

Power control in both up- and downlink has been implemented such that it is individual for each MTC device, facilitating individual channel conditions to each MTC device. So far a simple difference in path loss between devices is supported; more advanced channel emulation can be added, however this must be investigated considering the trade-off between channel complexity and scalability, as adding more complexity will require the implementation to be less computational efficient, effectively reducing the number of devices that can be emulated.

Furthermore, the maximum number of MTC devices which can physically transmit at the time without degrading the resolution of the signal transmitted from the SDR has been investigated. It is found that 64 devices (see analysis in section 4.2) should be able to transmit at the same time without excessive quantisation noise being generated in the radio front-end, if the output powers of the signals are equal. The dynamic range is therefore not the limiting factor on the prototype.

The design principle of the massiveMTC prototype implementation has been verified, and its performance evaluated, using three different eNodeBs. Tests show that it is possible to successfully decode combined signals from the devices at the eNodeB, and successful attachment of multiple devices has been accomplished using an eNodeB emulator.

The overall CPU load of the massiveMTC platform has been investigated in order to determine the maximum number of MTC devices the platform is able to keep in synchronisation with an eNodeB. The tests show that lowering the bandwidth increases the number of MTC devices which can run at the same time, as required computational power scaled with the bandwidth. Successful authentication was not possible at the commercial Nokia eNodeB due to issues with receiving messages from the core network, the results do however indicate how the performance is for multiple MTC devices which are kept in RRC connected mode, and thereby in synchronisation with the cell. Testing on an eNodeB emulator (OpenAirInterface) on which successful authentication is possible, gives an indication of how much extra computational power is needed once the authentication step is complete. The results using the commercial eNodeB show that up to 8 and 15 devices can be realised for 5 and 10 MHz respectively, using the Intel i7 processor based PC, indicating that the maximum number of emulated MTC devices will be much larger using MTC specific protocols, given the smaller bandwidths of LTE-M and especially NB-IoT. The results with the eNodeB emulator indicates that performance with full attachment will increase required computational power and thereby decrease the amount of devices that can be realised.

Common for all massiveMTC configurations is that the application becomes unstable before 100 % usage of the CPU cores is reached. This suggests that better performance can be obtained by code optimisation and mitigation of bottlenecks in the implementation, without changing hardware platform to dedicated hardware or to more powerful general purpose hardware. A computational gain is achieved by splitting the physical layer into a common and an individual part, this gain scale with an increase in bandwidth as well as with an increase of connected MTC devices. This gain paves the way for implementations supporting massive amounts of MTC devices. The gain can be increased by moving further functionality into the coPHY.

Both up- and downlink channel emulation are tested. The test of the uplink channel emulation in uplink is carried out by estimating the power of received preambles from multiple devices transmitting with different power: while a difference in received power is apparent, the received power estimation used on the OpenAirInterface is not suitable to further evaluate the specific output powers of the devices. The method is however proven to work, while the precision of the scaling has not been verified due to limitations of the test bed. Downlink channel emulation is validated using CQI reporting: the tests show that the path loss implementation does decrease the SNR as desired, however, the behaviour does not seem to result in the expected linear change in CQI values reported. The implementation is based on two features: the noise estimation and the digital scaling. Further testing of other noise estimation techniques within srsLTE should be tested, such as noise estimation of cell-specific reference signals or estimation using the empty resource elements in the resource grid, to compare with actual noise measurements in order to determine the best noise estimation procedure.

Additionally, further work should be put into the digital scaling procedure to make it less dependent on input power and generally more precise.

Generally, it has been shown that the design principle devised in this thesis does enable live emulation of multiple devices using a single radio front-end. Full authentication as well as data transfer has successfully been carried out using the massiveMTC prototype implementation, showing that the full LTE protocol stack functionality is implemented. The devised design principle and its prototype implementation serves as a strong proof-of-concept for massive device emulators as well as a platform for further development of "massive devices" testing for LTE based protocols and network performance.

6.1 Further work

This section covers some areas which could be investigated further or optimised on the current prototype platform, as well as additional functionalities that are deemed natural to consider for further development.

The overall stability of massiveMTC should be improved, as it is currently not uncommon that the coPHY layer loses synchronisation with the eNodeB. After lost synchronisation coPHY has to search for PSS in order to re-establish synchronisation, before the individual MTC devices are able to continue communication; an example of this can be seen in figure 5.5(b), where the data transfer is temporarily interrupted. The example in the figure is for the native srsUE, and the issue is inherently also in the massiveMTC device implementation. As srsUE is code that is continuously being updated, a potential solution could be to use an updated version of srsUE. Migration to the newest version will require the massiveMTC device changes to be manually reinstated on the new srsUE branch. However, it does not seem that this issue has yet been mitigated, so for further development this should be something to investigate, as this poses to be a limiting factor when scaling.

As instability occurs before 100 % CPU usage is reached (for any core, not in average), significant improvements should be achievable through bottleneck mitigation; one bottleneck is probably found in the access to shared media where mutexes are used to control the access. This should be improvable with a more flexible multi-thread access solution. Some calculation heavy processes could potentially be moved to an FPGA in order to reduce the load on the CPU. Whether it should be implemented in the FPGA on the SDR or a different FPGA depends of the size of the implementation and if USB 3.0 can support the increased data rate. An example of such a process is the turbo-decoder, which according to one of the developers of srsUE is dominating the CPU load – in some data heavy cases up to 80 % of the load. Further investigation with profiling tools may be able to find additional functionalities which could benefit from being implemented on an FPGA.

It is currently not possible to control the MTC devices in real time when running the massiveMTC implementation, this means that every time the user would like to change the amount of devices, the up- or downlink path loss, or when the MTC devices should (re-)try to attach to the cell, massiveMTC has to be restarted and in some of the cases also recompiled. A control layer should be implemented, so that when the virtualisation environment and middleware (as described in section 1.3.1) can pass such control information to the device handler, in which then functionalities should be implemented to start, stop, and change path loss for the individual MTC devices based on information arriving from the application layer through the middleware.

An example of a finished user-interface is shown in figure 6.1, which should be running on top of the virtualisation environment; it is sought that real-time outputs of the protocol performance are displayed, e.g. of the ARP performance and current capacity status of the cell.

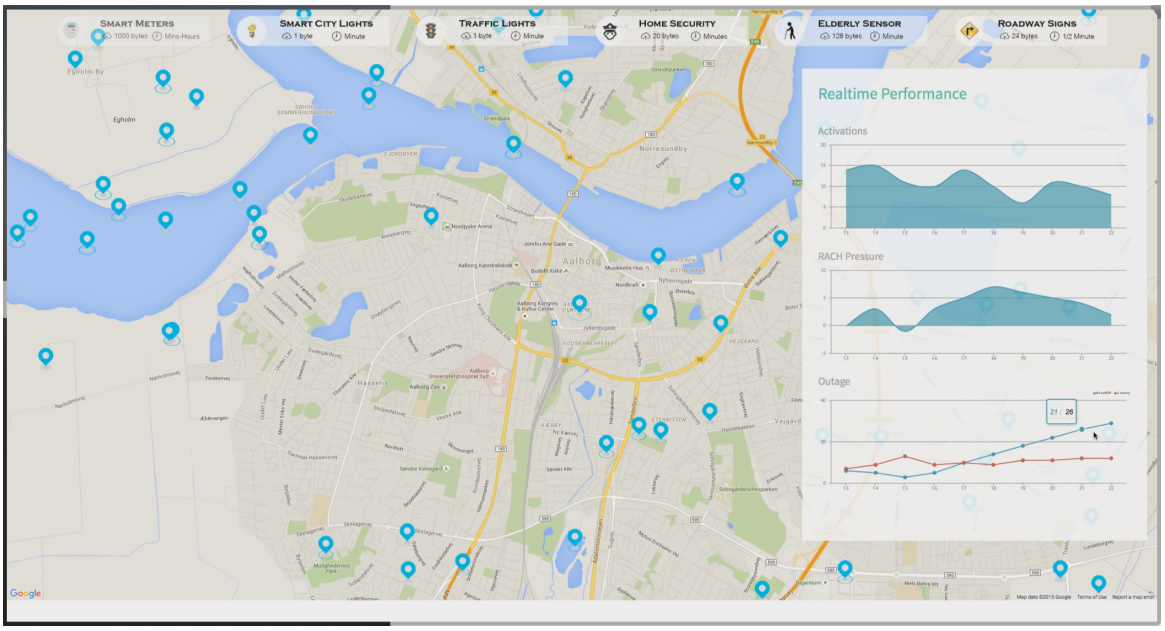


Figure 6.1: *Concept of finished product. Types of MTC devices with different data patterns can be added by clicking on a map, while real-time statistics on performance are displayed.*

The massiveMTC prototype implementation is developed on LTE Release 8 compatible source code. As development of implementations for LTE-M and NB-IoT advances, these are the obvious choice for further development to enable support of these protocols. Performance for these protocols should be expected to be much improved as they are simpler and operates lower bandwidths. The Software Radio Systems developers are currently working on a modified version of srsUE in order to support NB-IoT; migrating this (when available) into the massiveMTC implementation should not prove very demanding. Also, implementing LTE-M support should also be viable, as this in greater extend than NB-IoT resembles standard LTE. It is advised to update to the newest version of srsUE in the process before seeking to implement new physical layer functionalities.

The functionality for adding path loss is currently implemented in the form of a modified uplink power scaling and additional downlink noise generation. Several points of optimisation are available for this implementation: to enable precise uplink path loss, the scaling of the SDR gain to actual output power should be taken into account – the SDR gain does result in a linear output power change, however not a one-to-one change. For the downlink, as previously stated, the noise estimation process could be further investigated and the digital gain scaling should be improved. To obtain general improvements for the power control and channel emulation, it can be tried to manually calibrate the B210 board. The channel emulation development is kept on a show-case level being relatively simple, as this is not required to do core network and eNodeB load evaluation of massive devices. However, if it is deemed interesting and purposeful, further steps can be taken for the channel emulation: additional channel emulation on the baseband signals can be implemented in the form of a log-normal distribution emulating slow fading, as well as fast fading in the form of a Rayleigh or Rician distributed power.

Doppler effects can be added, which enables emulation of moving MTC devices; this can be applied to baseband signals in real time by shifting the baseband samples in a circle around the origin as a way of introducing a frequency offset. Adding more advanced channel emulation will result in a trade-off with performance, as this will add further load to the CPU. However, a solution to this is to implement the channel emulation on dedicated hardware in order to remove additional computations on the host PC.

For optimising the test bed consisting of the massiveMTC device and OpenAirInterface, the OpenAirInterface implementation should also be updated. OpenAirInterface is, as srsUE, under continuous development, while it appears that the developers tend to focus on upgrading the eNodeB towards newer releases, some improvements to the number of devices which can be connected at the same time has however taken place according to reports on the OpenAirInterface mailing list. The version of OpenAirInterface used for testing is unable to parse IP traffic coming from the internet back to the eNodeB; this is most likely tied together with the fact that the current setup has the eNodeB and core network running on the same PC. Reports from the mailing list indicate that people successfully solved this problem by running a newer version of the core network and eNodeB on separate PCs (it is unsure if running the core network on a virtual machine on the same PC as the eNodeB will solve the problem). Running the core network on a separate PC will also ease the reconfiguration if another core network should be used (such as the Polaris EPC emulator), as only the IP addresses used as interfaces between eNodeB and core network should be changed.

The OpenAirInterface core network is currently unable to release UEs from the network, it is therefore impossible to connect with the same SIM information twice without restarting the core network, as the core network will not accept the device a second time. The developers of OpenAirInterface has reported that an update supporting release of devices should be coming to the *develop* branch soon.

The authors believe that the work done in this thesis can serve as a strong starting point for further development of massive device testing platforms, enabling emulation of multi-device scenarios in future communication protocols.

Bibliography

[1] 3GPP. LTE releases. http://www.3gpp.org/specifications/67-releases. [Online, accessed June, 2017].

[2] Denise Lund, Carrie MacGillivray, Vernon Turner, and Mario Morales. Worldwide and regional internet of things (IoT) 2014–2020 forecast: A virtuous circle of proven value and demand. Technical report, International Data Corporation (IDC), 2014.

[3] Ye Yan, Yi Qian, Hamid Sharif, and David Tipper. A survey on smart grid communication infrastructures: Motivations, requirements and challenges. *IEEE Communications Surveys and Turtorials*, 15(1):5–20, 2013.

[4] Erfan Soltanmohammadi, Kamran Ghavami, and Mort Naraghi-Pour. A survey of traffic issues in machine-to-machine communications over LTE. *IEEE Internet of Things journal*, 3(6):1–21, 2015.

[5] Andrea Biral, Marco Centenaro, Andrea Zanella, Lorenz Vangelista, and Michele Zorzi. The challenges of M2M massive access in wireless cellular networks. *Digital Communications and Networks*, 1:1–19, 2015.

[6] M. Zubair Shafiq, Lusheng Ji, Alex X. Liu, Jeffrey Pang, and Jia Wang. A first look at cellular machine-to-machine traffic – large scale measurement and characterization. *IEEE/ACM Transactions on Networking*, 21(6):1960–1973, 2013.

[7] Nokia. LTE evolution for IoT connectivity. https://tools.ext.nokia.com/asset/200178, 2016. White paper.

[8] Ericsson. Cellular networks for massive IoT. https://www.ericsson.com/assets/local/publications/white-papers/wp_iot.pdf, 2016. White paper.

[9] Mads Lauridsen, István Kovács, Preben Elgaard Mogensen, Mads Sørensen, and Steffen Holst. Coverage and capacity analysis of LTE-M and NB-IoT in a rural area. *IEEE Vehicular Technology Conference*, (84), 2016.

[10] GSMA. 3GPP low power wide area technologies. http://www.gsma.com/iot/3gpp-low-power-wide-area-technologies-white-paper/, 2016. Mobile IoT white paper.

[11] Philippe Reininger. 3GPP Standards for the Internet of Things. http://www.3gpp.org/ftp/Information/presentations/presentations_2016/2016_11_3gpp_Standards_for_IoT.pdf. 3GPP presentation, Smart Summit Singapore November 2016. [Online, accessed June, 2017].

[12] 3GPP. Progress on 3GPP IoT. http://www.3gpp.org/news-events/3gpp-news/1766-iot_progress. [Online, accessed June, 2017].

[13] 3GPP. Standardization of NB-IOT completed. http://www.3gpp.org/news-events/3gpp-news/1785-nb_iot_complete. [Online, accessed June, 2017].

[14] Andres Laya, Luis Alonso, and Jesus Alonso-Zarate. Is the random access channel of LTE and LTE-A suitable for M2M communications? A survey of alternatives. *IEEE Communications Surveys and Turtorials*, 16(1):4–16, 2014.

[15] Carsten Bockelmann, Nuno Pratas, Hosein Nikopour, Kelvin Au, Tommy Svensson, Cedomir Stefanovic, Petar Popovski, and Armin Dekorsy. Massive machine-type communications in 5G: Physical and MAC-layer solutions. *IEEE Communications Magazine*, 54(9):59–65, September 2016.

[16] Nuno K. Pratas, Sarath Pattathil, Čedomir Stefanović, and Petar Popovski. Massive machine-type communication (mMTC) access with integrated authentication. *IEEE International Conference on Communications (ICC)*, 2017.

[17] Monica Alleven. KT, Samsung pursue NB-IoT pilot. `http://www.fiercewireless.com/wireless/kt-samsung-to-launch-nb-iot-pilot`. FireceWireless.com, April 26, 2017. [Online, accessed June, 2017].

[18] Iain Morris. Could LTE-M Torpedo NB-IoT? `http://www.lightreading.com/iot/nb-iot/could-lte-m-torpedo-nb-iot/a/d-id/732464`. LightReading.com, February 2, 2017. [Online, accessed June, 2017].

[19] Dino Flore. LTE evolution and 5G. `http://www.3gpp.org/ftp/Information/presentations/presentations_2016/2016_11_flore_LTE%20evolution%20and%205G.pdf`. 3GPP presentation, Broadband World Forum 2016. [Online, accessed June, 2017].

[20] Kaj Printz Madsen, Rune Willum Larsen, Jonas Sand Madsen, Rene Mejer Lauritsen, Casper Møller Bartholomæussen, and Carsten Vestergaard Risager. Kuiga Box: A test bed for large scale testing of virtualized IoT devices, 2016. Aalborg University, Department of Computer Science, student project, sw704e16.

[21] Anders Normann Poulsen and Gabriel Vasluianu. NTA – Network Test Application, 2017. Aalborg University, Department of Computer Science, student project, d807f17.

[22] Upamanyu Madhow. *Introduction to Communication Systems*. University of California, Santa Barbara, 2014.

[23] Di Pu and Alexander M. Wyglinski. *Digital Communication Systems Engineering with Software-Defined Radio*. Artech House, 2013.

[24] Eugene Grayver. *Implementing Software Defined Radio*. Springer, 2013.

[25] Xilinx Inc. ZYNQ boards, SoCs and MPSoCs. `https://www.xilinx.com/products/silicon-devices/soc.html`. [Online, accessed June, 2017].

[26] Ettus Research. FPGA utilization statistics. `https://kb.ettus.com/B200/B210/B200mini/B205mini#FPGA`. [Online, accessed June, 2017].

[27] Rodger H. Hosking. Software-Defined Radio Handbook. `http://www.pentek.com/pildocs/8363/techother/DGTLRCVRHBK43.PDF`. 12th edition, PENTEK. [Online, accessed June, 2017].

[28] Alan V. Oppenheim and Ronald W. Schafer. *Discrete-Time Signal Processing*. Pearson, third edition, 2014.

[29] Stefania Sesia, Issam Toufik, and Matthew Baker. *LTE - The UMTS Long Term Evolution: From theory to practice*. Wiley, second edition, 2011.

[30] Harri Holma and Antti Toskala. *LTE for UMTS: Evolution to LTE-Advanced*. Wiley, second edition, 2011.

[31] Erik Dahlman, Stefan Parkvall, and Johan Sköld. *4G - LTE/LTE-Advanced for Mobile Broadband*. Academic Press, second edition, 2014.

[32] 3GPP. Evolved Universal Terrestrial Radio Access (E-UTRA) and Evolved Universal Terrestrial Radio Access Network (E-UTRAN); Overall description; Stage 2, 2010. TS 33.300, version 8.12.0. Release 8. www.3gpp.org.

[33] 3GPP. Evolved Universal Terrestrial Radio Access (E-UTRA); Radio Resource Control (RRC); Protocol specification, 2014. TS 33.331, version 8.21.0. Release 8. www.3gpp.org.

[34] Germán C. Madueño, Jimmy J. Nielsen, Dong Min Kim, Nuno K. Pratas, Čedomir Stefanović, and Petar Popovski. Assessment of LTE wireless access for monitoring of energy distribution in the smart grid. *IEEE Journal on Selected Areas in Communications*, 34:675–688, 2016.

[35] Mohammad T. Kawser, Nafiz Imtiaz Bin Hamid, Md. Nayeemul Hasan, M. Shah Alam, and M.Musfiqur Rahman. Downlink SNR to CQI mapping for different multiple antenna techniques in LTE. *International Journal of Information and Electronics Engineering*, 2(5):757–760, 2012.

[36] 3GPP. Evolved Universal Terrestrial Radio Access (E-UTRA); Physical layer procedures, 2009. TS 36.213, version 8.8.0. Release 8. RTS/TSGR-0136213v880, www.3gpp.org.

[37] 3GPP. Evolved Universal Terrestrial Radio Access (E-UTRA); Base station radio transmission and reception, 2016. TS 36.104, version 8.14.0. Release 8. www.3gpp.org.

[38] Software Radio Systems (SRS). srsLTE/srsUE. https://github.com/srsLTE. [Online, accessed June, 2017].

[39] Ettus Research. USRP hardware driver and USRP manual. https://files.ettus.com/manual/. [Online, accessed June, 2017].

[40] OpenAirInterface Software Alliance (OSA). OpenAirInterface. http://www.openairinterface.org/?page_id=72. [Online, accessed June, 2017].

[41] Ettus Research. USRP B210 UHD FPGA image probing. Specifications returned by connected board when probing with uhd_usrp_probe.

[42] Ettus Research. USRP B200/B210 Bus Series Specification Sheet. https://www.ettus.com/content/files/b200-b210_spec_sheet.pdf.

[43] Ettus Research. B210 RF performance. https://kb.ettus.com/B200/B210/B200mini/B205mini#RF_Performance_Data. Page 235, [Online, accessed June, 2017].

[44] 3GPP. Evolved Universal Terrestrial Radio Access (E-UTRA); User Equipment (UE) radio transmission and reception, 2016. TS 36.101, version 8.27.0. Release 8. www.3gpp.org.

[45] Junyi Li, Xinzhou Wu, and Rajiv Laroia. *OFDMA Mobile Broadband Communications: A systems approach*. Cambridge University Press, 2013.

[46] Cubic metric in 3GPP-LTE. Technical report, Motorola, 2006. 3GPP TSG RAN WG1 LTE Adhoc - Tdoc R1-060023.

[47] 3GPP. 3GPP System Architecture Evolution (SAE); Security architecture, 2008. TS 33.401, version 8.1.1. Release 8. www.3gpp.org.

[48] 3GPP. Technical Specification Group Core Network and Terminals; Numbering, addressing and identification, 2009. TS 23.003, version 8.4.0. Release 8. www.3gpp.org.

Appendices

A. LTE protocol

The LTE protocol consists of multiple layers, this appendix will explain the major functionalities of the layers. The layers and their interconnectivity are displayed in figure A.1

A packet received by a layer is known as a service data unit (SDU), while a packet leaving a layer is known as a protocol data unit (PDU). This means that a PDCP PDU is the same as a RLC SDU as it is the same packet from the "point of view" of two different layers. The LTE protocol is split up into two planes; the control plane and the user plane. Only the NAS and RRC layer operates solely on the control plane, while the application layer communicates using IP packets is solely on the user plane. [29]

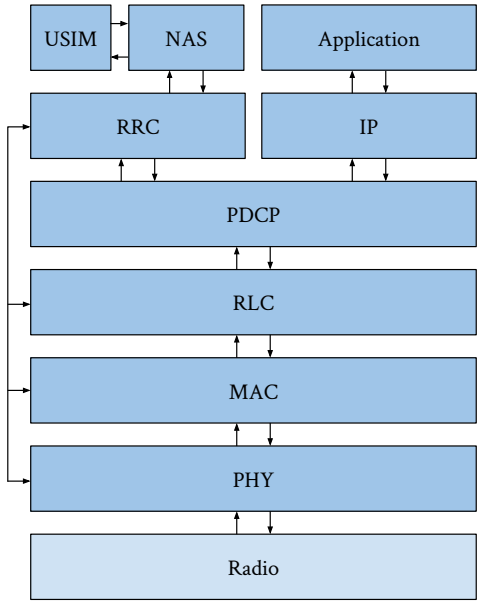


FIGURE A.1: *LTE protocol stack of a UE.*

A.1 LTE layers

NAS

The NAS layer is the highest layer in the control plane. It is responsible for communication between the UE and the MME. It handles mobility of the UE, as well as management and establishment of IP connectivity, as it transmits the USIM information to the MME during initial connection. [29]

RRC

The RRC layer is mainly responsible for transmission of system information from both NAS and AS, paging establishment, maintenance of RRC connections (connection between UE and E-UTRAN) and security functions in the AS. [29]

PDCP

The PDCP layer is responsible for handling messages to and from the RRC layer in the control plane and IP packets in the user plane. The main functionalities of the PDCP layer are IP header compression, as well as security, both in the form of ciphering and integrity protection. Reordering and duplicate detection of bearers during handover also takes place at the PDCP layer. [29]

Data above the PDCP is transported using bearers, there are multiple types of bearers in LTE. Between UE and eNodeB, there is two types of radio bearers, namely the signalling radio bearers (SRBs) and data radio bearers (DRBs). Below is a brief explanation of the roles of each bearer:

- SRB0 is responsible for initial RRC messages in both up- and downlink. SRB0 is transporting data on common control channel (CCCH) using RLC transparent mode.
- SRB1 handles amongst other things RRC connection reconfiguration and release, UE capability inquiries, and data information transfer. SRB1 is transmitted on dedicated control channel (DCCH) using RLC acknowledged mode.
- SRB2 is only used for user plane data, it should be noted that SRB2 has lower priority than SRB1, and is only configured after security has been established. SRB2 is transmitted on DCCH using RLC acknowledged mode.
- DRBs are configurable depending on which service is required; any RLC transmission mode can be applied, and they can be have prioritised bit rates in order to guarantee a better QoS. Multiple DRBs can be created for a single UE.

One RLC and PDCP entity is established for every active radio bearer. [29]

RLC

The main functions of RLC are segmentation and reassembly of packets. Retransmission on the RLC layer is possible depending on which transmission mode is chosen, of which there are three:

- Transparent mode (TM) – If TM is used, no functionality are performed to the packets that pass through the RLC entity. This is only used for system information (SI) (MIB and SIBs), paging messages, and RRC messages sent when only SRB0 is available. This entity is unidirectional, meaning that it has to be set up as either as a transmitting or a receiving entity.
- Unacknowledged mode (UM) – which is also unidirectional, used for data transfer of delay sensitive and error tolerant applications, such as voice over IP (VoIP). Using this mode allows for segmentation, concatenation, reassembly, reordering, and duplicate detection of packets.
- Acknowledged mode (AM) – is bidirectional and supports retransmission, such that an entity can both transmit and receive. This mode is mainly used by error sensitive but delay tolerant services, such as web browsing and file downloading.

One RLC entity is established for every active bearer. The data transferred from the RLC to the MAC layer is going through logical channels which are either control logical channels (for RRC signalling and other control data) and traffic logical channels (for user data). These, along with transport and physical channels is discussed in A.2. [29]

MAC

The MAC layer performs multiplexing and demultiplexing between the logical channels and the transport channels. As shown in figure A.2, there is only one MAC entity per UE, which combines the different RLC entities (channels). The MAC layer aims to uphold the desired QoS for each radio bearer. As a result of this, the MAC layer has to report the amount of buffered data left for transmission to the eNodeB; this is carried in BSRs. The MAC also handles random access control, HARQ signalling, timing advance estimation, and discontinuous reception (DRX). [29]

PHY

The main functions of the PHY layer are to handle TB mapping from the resource grid to and from the MAC layer, link adaptation, power control, and cell search/synchronisation. [29]

A.2 LTE channels

The purpose of the logical channels, along with which channels they are routed to, is described in this section; this together with the discussion in section 2.3.2 seek to provide a basic understanding of the channels in LTE. The channels are illustrated in figure A.2; multicast channels are included in the illustration for clarification, but are not implemented in LTE until Release 9, they were however included in the Release 8 standard due to forward compatibility [29].

The broadcast control channel (BCCH) is used in downlink to transmit SI (MIB and SIBs), MIB is transmitted using the PBCH while SIB is routed to PDSCH via downlink shared channel (DL-SCH). Paging control channel (PCCH) handles paging (incoming data to the UE), it routes to PDSCH. When no connection is established, the CCCH is used for control information in both up- and downlink, while the DCCH serves the same purpose when the UE is in RRC connected state with the eNodeB. It should be noted that CCCH operates in RLC TM mode, while DCCH operates in RLC AM mode. The dedicated traffic channel (DTCH) transmits, as the name implies, user traffic in both up- and downlink. [29]

Below is a list of all channels displayed in figure A.2.

Physical channels:

- Physical broadcast channel (PBCH)
- Physical downlink shared channel (PDSCH)
- Physical HARQ indicator channel (PHICH)
- Physical downlink control channel (PDCCH)
- Physical control format indicator channel (PCFICH)

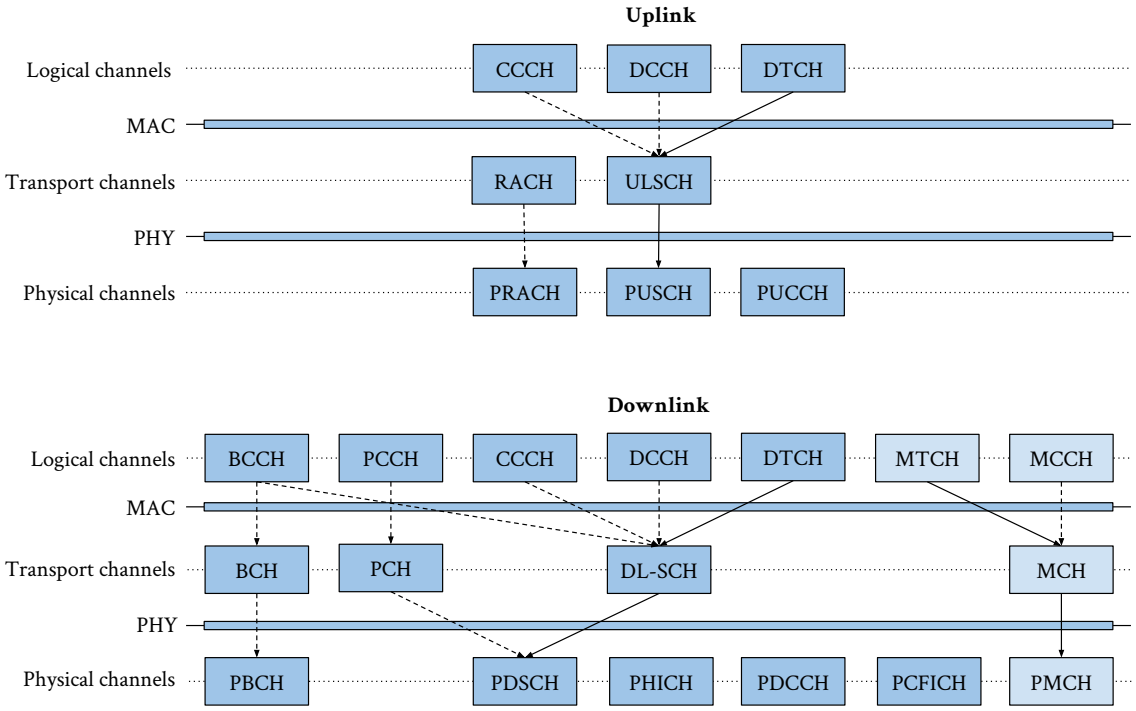


FIGURE A.2: *Logical, transport and physical channels in up- and downlink. Note that radio bearers and physical signal channels such as PSS and SRS are omitted. Light blue boxes are multicast channels. Dashed lines are routes purely for control information.*

- Physical multicast channel (PMCH)
- Physical random access channel (PRACH)
- Physical uplink shared channel (PUSCH)
- Physical uplink control channel (PUCCH)

Transport channels:

- Broadcast channel (BCH)
- Paging channel (PCH)
- Downlink shared channel (DL-SCH)
- Multicast channel (MCH)
- Random access channel (RACH)
- Uplink shared channel (ULSCH)

Logical channels:

- Broadcast control channel (BCCH)
- Paging control channel (PCCH)
- Common control channel (CCCH)

- Dedicated control channel (DCCH)
- Dedicated traffic channel (DTCH)
- Multicast traffic channel (MTCH)
- Multicast control channel (MCCH)

B. LTE multiple access techniques

The fundamental multiple access techniques of LTE is orthogonal frequency-division multiple access (OFDMA) and single carrier FDMA (SC-FDMA) for downlink and uplink, respectively. The principles of these two techniques are discussed in the following sections to give insight to the methods and how they give rise to the basics of the LTE resource grid and LTE in general.

B.1 Orthogonal frequency-division multiple access (OFDMA)

Orthogonal frequency-division multiplexing (OFDM) is the basis scheme of OFDMA. OFDM divides the available bandwidth into N_c subcarriers (or subchannels, also called "tones") that are equally spaced. These subcarriers then make up a series of non-frequency-selective (narrowband) channels; they are overlapping, but are orthogonal to each other, an example of this can be seen in figure B.1. This gives rise to high spectral efficiency compared to conventional FDMA, as no guard bands are required due to the orthogonality. It also enables low complexity on the receiver-side, which is desired for cheap, mobile or high-rate devices. [29, 45]

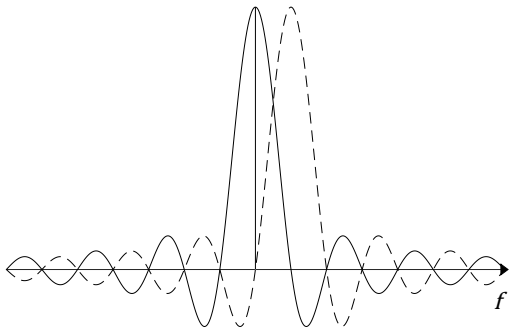


FIGURE B.1: *Frequency spectrum response, subcarriers in OFDM.*

The principle of OFDMA is to split up a high symbol rate (wideband) signal into parallel low rate signal on each of these sub-carriers. Doing so increases symbol durations, which is desired as the symbol duration, T_s, should be large compared to the delay spread in the channel, τ_s, decreasing the influence of inter symbol interference (ISI). The principle is illustrated in figure B.2 The long symbol duration is also important in relation to the coherence period, T_{coh}, as this effectively makes the channel a linear time-invariant system. This ensures that the signal orthogonality is preserved and by that there is no inter-carrier interference (ICI). [29, 45]

Though the symbol duration is long, ISI is not negligible; a guard period between symbols must therefore be implemented. Instead of a simple, empty guard period, a cyclic prefix (CP) is applied; this takes the last part of the symbol and duplicates it in front of the symbol, as illustrated in figure B.3. The duration of the CP, T_{cp}, should be as long as the longest (supported) channel impulse response to completely counter the ISI. The application of CP (in contrast to using a simple guard period) results in a symbol that appears cyclic/periodic, which makes for simpler processing using a discrete Fourier transform (DFT). [29, 30, 45]

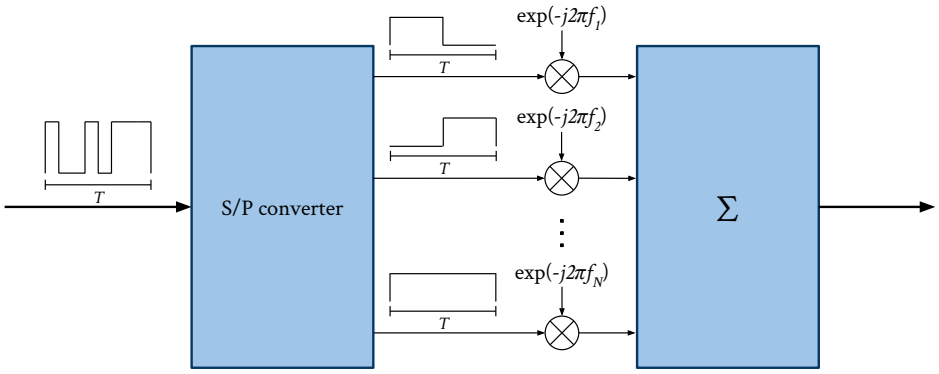


Figure B.2: *High rate signalling put on parallel (lower rate) data streams on the sub-carriers.*

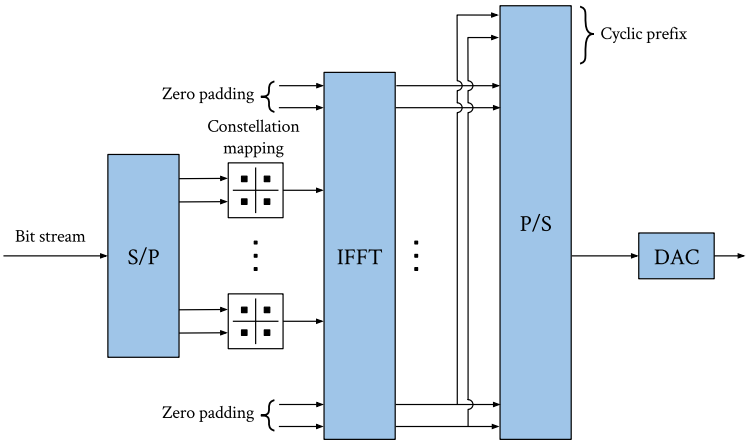


Figure B.3: *OFDM transmission chain.*

By definition, an OFDM symbol is a sum of N_c complex symbols modulated on the sub-carriers. This causes a Gaussian waveform (in time domain), which generally speaking has a high peak-to-average power ratio (PAPR), i.e. large power fluctuations over time; this is discussed further in [29, 46]. This poses a major drawback of the OFDM scheme: having large PAPR results in difficult requirements for the power amplifier, as it will be required to have a large dynamic range; otherwise the signal will suffer from non-linear distortion due to clipping, which will produce both in-band distortion of the signal as well as out-of-band noise. To counter this, a large back-off must be applied in the dynamic range, which in turn results in inefficient transmission. This is the main reason that OFDMA is not used in the uplink of LTE, as efficient performance is vital for mobile (battery powered) UEs. [29, 45]

As previously stated, ICI is ideally not apparent for OFDM systems. However, this is only true when the transmitter and the receiver are synchronised in frequency. This is difficult to obtain in real-life systems, especially as UE hardware is to be cheap and therefore often has issues as frequency drift as well as Doppler effects due to moving terminals. Therefore, in LTE systems the CFO is tracked and sought countered. [29]

OFDMA is the application of OFDM combined with frequency-division multiple access (FDMA) and/or time-division multiple access (TDMA). In LTE, both are implemented, as the downlink stream is divided in a resource grid in frequency and time. This is illustrated in figure B.4. The resource grid is composed of RBs, which are individually allocated to different users.

The smallest element in the resource grid is the RE, which denotes individual modulation symbols on the subcarriers. The modulation used depends on the current channel conditions, where good channel conditions allow for a more bits to be modulated in a single symbol, yielding a higher data rate. [29]

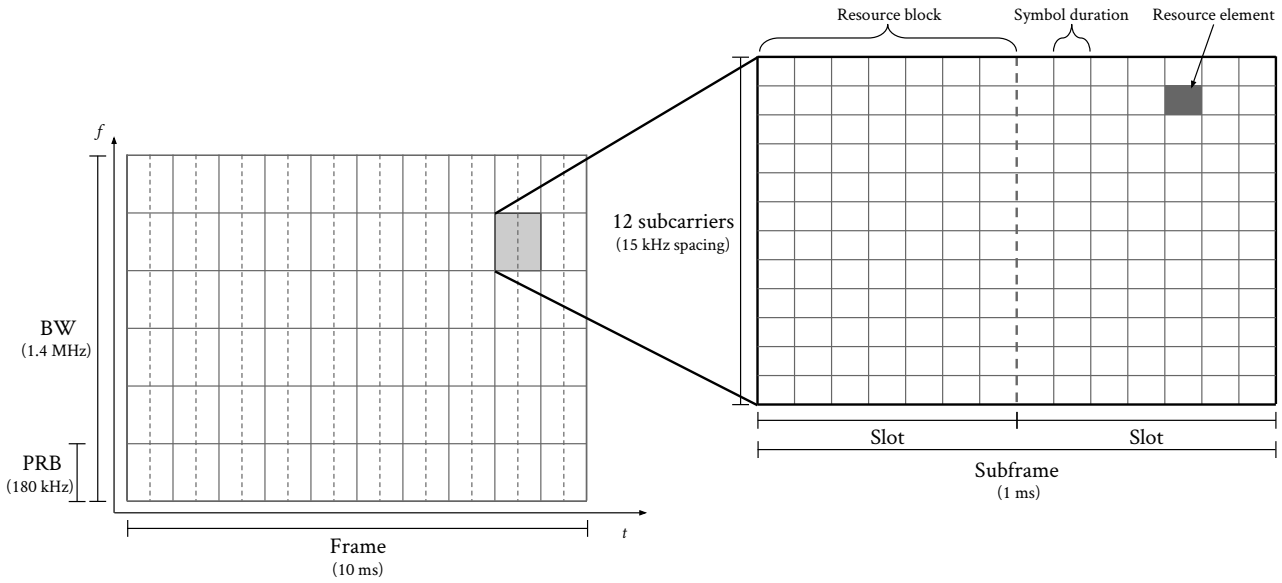


FIGURE B.4: *OFDMA/LTE resource grid, illustrating how frequency and time is "grouped" in blocks (example with lowest allowed bandwidth in LTE of 1.4 MHz).*

B.2 Single carrier FDMA (SC-FDMA)

Single carrier FDMA (SC-FDMA), also known as linearly precoded OFDMA (LP-OFDMA) or DFT-spread-OFDM (DFT-S-OFDM), is favourable compared to OFDMA in uplink due to the reduction in PAPR, effectively reducing the requirements for the power amplifier in the UE. The principle of SC-FDMA is similar to OFDMA: SC-FDMA splits the wideband channel into parallel, orthogonal subcarriers and also applies CP to counter ISI. The SC-FDMA in uplink LTE is divided into a resource grid with same dimensions as the downlink OFDMA resource grid, providing commonality between the up- and downlink methods. [29, 30]

Instead of directly modulating each subcarrier, SC-FDMA modulates a linear combination of the symbols to be transmitted at a given time on all allocated subcarriers. This results in that the signals on each subcarrier are not independent; this gives SC-FDMA its single carrier property and makes the waveform not to be Gaussian (as is the case for OFDM), which in turn reduces the PAPR. [29]

Generally, there are two ways of allocating the subcarriers in SC-FDMA: distributed and localised. When subcarriers are localised, a given device gets allocated a set of adjacent subcarriers for transmission while the rest of the bandwidth subcarriers components are set to zero. For distributed transmission, the set of allocated subcarriers is placed with a fixed amount of subcarriers between them; as for localised, unallocated subcarrier components are set to zero. In LTE, all uplink transmissions are of the localised kind[1], except for the transmission of SRS. Using the localised method, resource blocks can thereby be allocated for devices in a similar fashion as in downlink; allocated resource blocks for one device at a given time are simply required to be adjacent in frequency. [29, 45]

[1]Some changes have been made in LTE-A.

For localised transmission, frequency diversity can still be obtained by frequency hopping; in LTE this can occur between the slots in a resource block or simply between resource blocks. The distributed SRS provides the information required for the eNodeB to schedule effectively in order to cope with frequency-selective fading. [29]

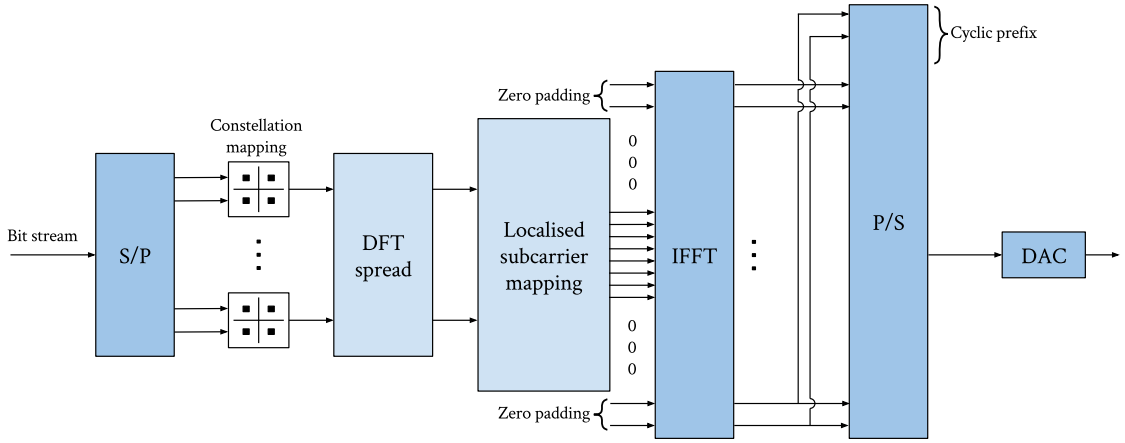


Figure B.5: *Transmission chain for SC-FDMA with localised subcarriers.*

While many of the virtues of OFDMA are kept, some trade-offs are apparent. SC-FDMA requires frequency domain equalisation at the receiver in order to combat ISI (happens when the channel is frequency selective) within an OFDM symbol, as now the transmission of modulation symbols occurs across multiple subcarriers (note that no ISI between OFDM symbols due to the CP). This adds some complexity to the receiver, which however is accepted as it still allows relative simple design on the transmitter side, i.e. in the UE. Also, uplink scheduling can be performed in a less flexible matter due to the requirements of localised subcarrier placement in SC-FDMA, which means that allocated RBs must be adjacent in frequency. [29, 45]

C. Security

The security aspect of LTE is discussed in the form of authentication procedure and security options upon successful authentication. In conclusion, values in the HSS implemented in OpenAirInterface are explained along with how they are derived.

C.1 Authentication

In GSM, authentication is uni-directional, meaning that it is only the network which has to authenticate the UE. In LTE and UMTS, the authentication is bi-directional, meaning that both the network and the UE has to authenticate each other; this is implemented to eliminate the security risk of fake base stations trying to trick UEs into thinking they are connected to an authentic network.

Upon provision of a new device, a secret key denoted K is placed on the USIM and in the HSS database. This is an unique key for all UEs and all integrity and ciphering keys are based on this number [47]. Going in detail with the derivation of all security keys is beyond the scope of this thesis, the most notable of the keys is however the key for the access security management entity (ASME), K_{ASME}, which is used to derive several other keys. A brief review of the authentication procedure is given in order to give insight how the key is used; the exchange of messages is displayed in figure C.1.

Upon successful random access, a UE transmits the RRC connection setup complete message which includes the "attach request" as part of the initial NAS message; see section 2.3.3.2. If the UE is not known to the network before this connection, or it is the first connection within some operator specified time, the MME will send a authentication information request (AIR). The HSS will then compute K_{ASME}, an authentication token $AUTN$, the expected response from the UE $XRES$ and a random number $RAND$ using K for the specific MTC device. An identity of the MTC has to be transmitted from the UE to the HSS; if it is the first time connecting to the network, international mobile subscriber identity (IMSI) is used as identification in the RRC connection setup complete message, otherwise a temporary identifier can be used if the device is known to the network beforehand. The IMSI is transmitted as rarely as possible in order to prevent spoofing of this number. [47]

The HSS transmits a response with an authentication information answer (AIA) to the MME, if the subscriber is found in the HSS database; the response contains $RAND$, $AUTN$, K_{ASME}, and $XRES$. The MME then transmits an authentication request message, including $RAND$, $AUTN$, and KSI_{ASME} (key selection identifier) to the UE. The UE will then read K from the SIM card and use the received numbers to compute K_{ASME} using KSI_{ASME}. In this way, K_{ASME} will not be exposed OTA, which is important as the authentication request is non-ciphered. [47]

The UE can then create an authentication result from K_{ASME} and K using the same algorithm as the HSS does. The UE will then send the RES value back to the MME in the authentication response message. Upon receiving the authentication response, the MME will compare RES with $XRES$ value from the HSS; if the two values match, the UE is authenticated, this will allow the attachment procedure to proceed; as the next steps is computing and applying integrity and ciphering keys. [47]

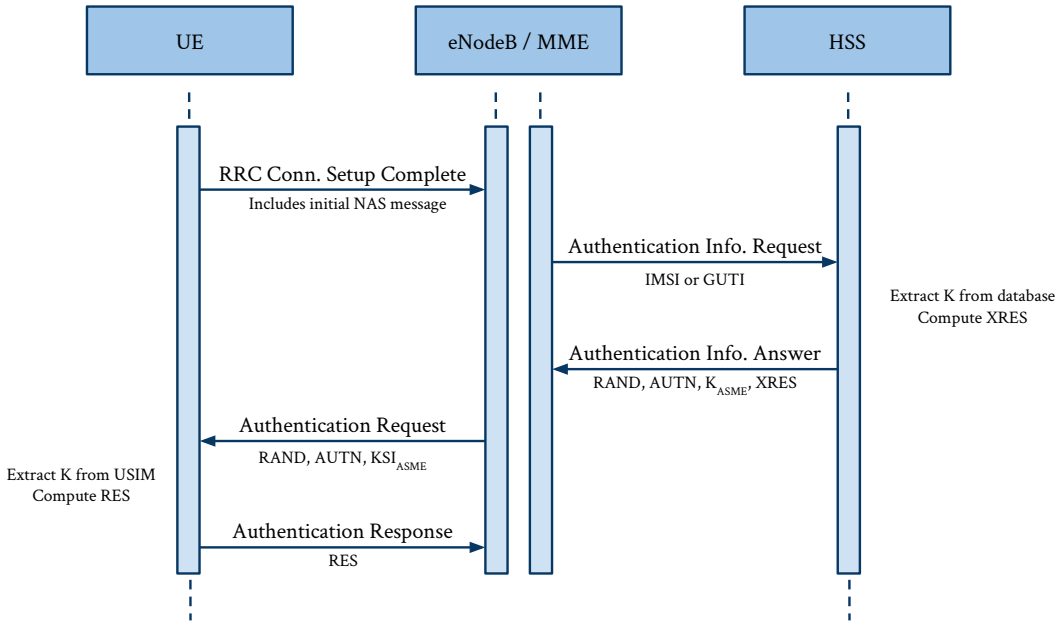


Figure C.1: *Flowchart of authentication procedure upon successful connection in LTE, note that communication between MME and eNodeB is omitted as it forwarding data in all cases except for XRES, which stays at the MME.*

C.2 Integrity protection and ciphering

The LTE protocol features both integrity protection and ciphering, this can be applied to both on the NAS (UE to MME communication) and AS (UE to eNodeB communication) layer [47]. The NAS security is capable of supporting both integrity protection and ciphering, ciphering is however optional. The NAS security keys are constituted by the integrity key, $K_{\text{NAS,int}}$, and ciphering (encryption) key $K_{\text{NAS,enc}}$; they are both derived from K_{ASME}.

All integrity protection and ciphering takes place on the PDCP layer. The AS security is similarly to the NAS security capable of supporting both integrity protection and ciphering. The keys used are $K_{\text{RRC,int}}$ and $K_{\text{RRC,enc}}$ for integrity and encryption of the control plane, respectively. The $K_{\text{UP,enc}}$ is used for for ciphering of the data plane (the actual IP packets); in this case, there is no integrity protection. These three keys are derived from K_{eNB}, which in turn is derived from K_{ASME}, while the UE has access to K_{ASME}, the eNodeB does not, it will therefore get K_{eNB} forwarded from the MME. [47]

Multiple encryption algorithms are available in LTE; the used algorithm is indicated by the EPS encryption algorithm (EEA). Three algorithms are supported, namely null ciphering, SNOW, and advanced encryption standard (AES). A fourth algorithm called ZUC was introduced in Release 11 (and onwards). The null ciphering algorithm is almost exclusively used for testing purposes and in some special cases such as emergency calls, where USIM is not available (or required). The same method as for EEA applies to integrity algorithms using the EPS integrity algorithm (EIA) identifier. Algorithms can be specified individually for NAS, RRC (AS control plane), and IP (AS data plane) integrity protection. [47]

C.3 Identifiers

This sections covers the most important identifiers used in LTE. As discussed in C.1, USIM is storing the pre-shared secret key, K, it is also storing the IMSI. The IMSI is unique, and is assigned to each subscriber. The IMSI is composed of three parts: the three digit mobile country code (MCC), which is an unique identifier of which country the mobile provider is based, the two or three digit mobile network code (MNC), which is an identifier for each provider, and lastly the mobile subscriber identification number (MSIN), which is an operator assigned number identifying each subscriber. MCC and MNC combined is known as the public land mobile network (PLMN); PLMN it is another way of identifying each provider. [48]

In order to call a subscriber, one would have to know the mobile station international subscriber directory number (MSINDN); this consists of country code (CC), national destination code (NDC), and subscriber number (SN), which is similar to IMSI. MSINDN is in other words the actual phone number, while IMSI is the number of the SIM card; this makes it possible to change phone number without changing SIM card. [48]

Another way of identifying subscribers on the network is by using international mobile equipment identity (IMEI); this is a unique number identifying a physical piece of equipment which can be used to identify or blacklist stolen phones regardless of which USIM card is used. [48]

C.4 OpenAirInterface HSS

In order to program the HSS database in OpenAirInterface, a number of values has to be added to the SQL database that the HSS has access to, some of these values has to be changed for every MTC device, these values are shown in table C.1, along with the initial values.

Table C.1: *Values used for the first device the in massiveMTC implementation.*

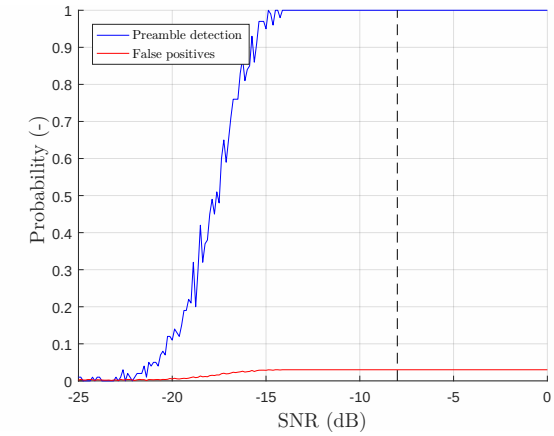
Name	**Initial value**
IMSI	208930000000008
MSINDN	88211005938
K	2DC204753BEA70DC8F010A4DFEDCEE33
OP	11111111111111111111111111111111

While the OP is the same for all devices, it is used to calculate OPc. The standard way is $OPc = AES_{128}(K, OP) \oplus OP$, this has to be carried out for every device. For every new MTC device, the IMSI, MSINDN and K values are incremented by 1 while OP stays the same, the OPC will however change. It can be see from the IMSI that the MCC is 208 and the MNC is 93.

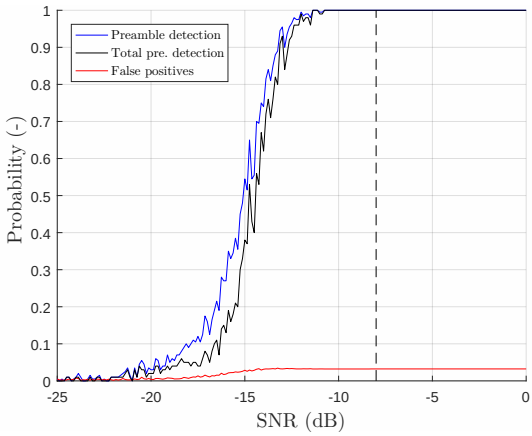
D. PRACH simulation results

Simulation results discussed in section 2.4.

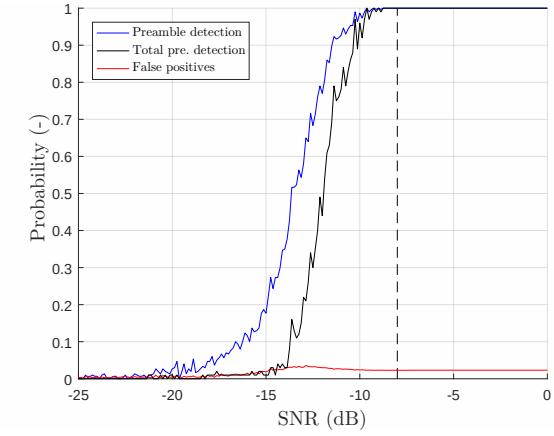
(a) 1 UE.



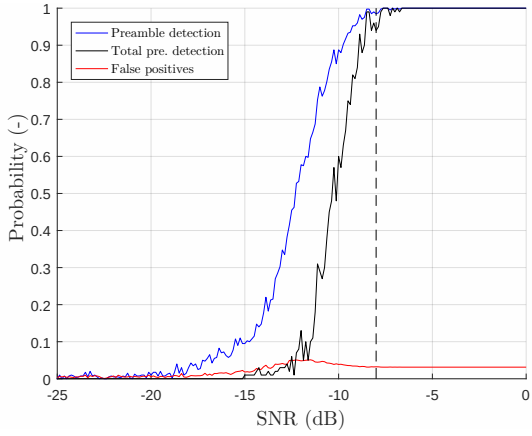
(b) 2 UEs.



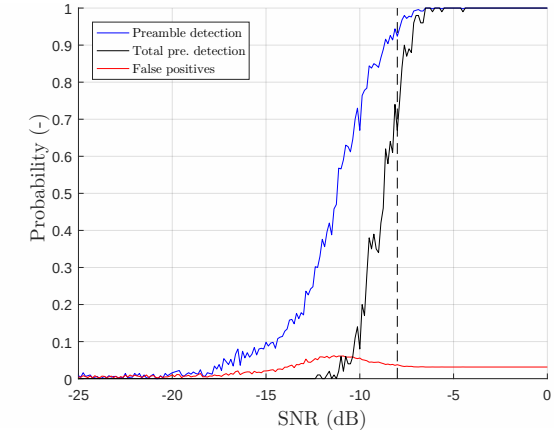
(c) 3 UEs.



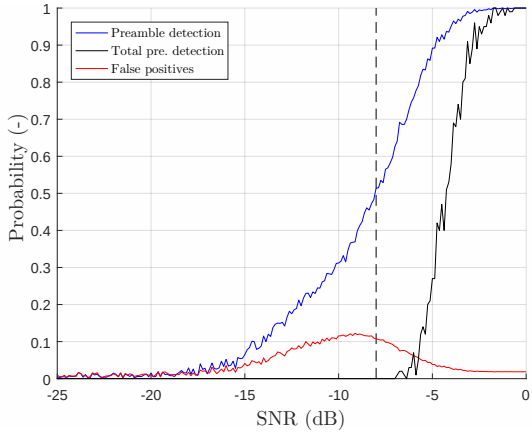
(d) 4 UEs.



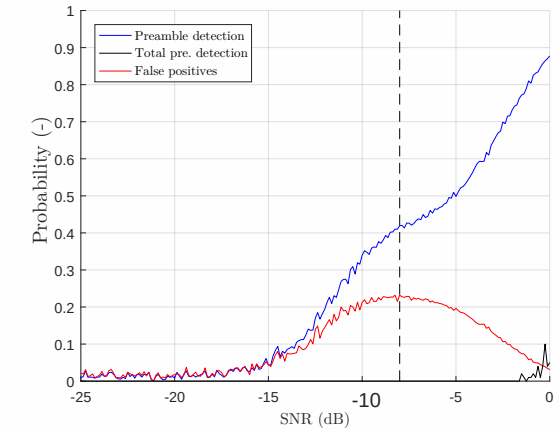
(e) 5 UEs.



(f) 10 UEs.



(g) 20 UEs.



(h) 30 UEs.

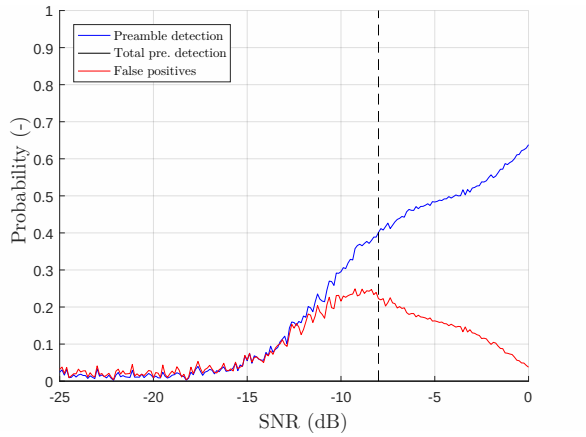


FIGURE D.1: *Simulated preamble detection with N UEs transmitting in the same RAO.*

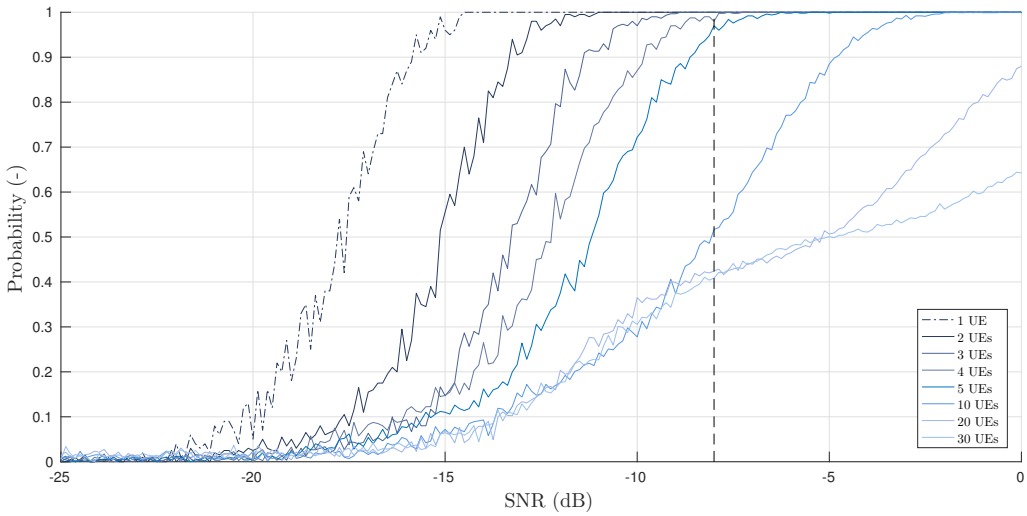
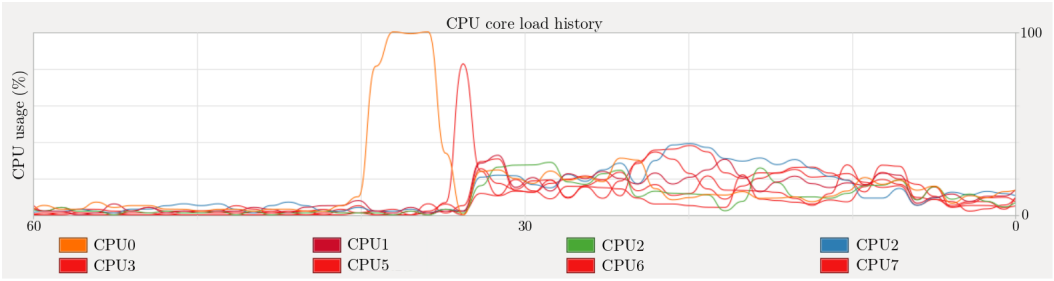


FIGURE D.2: *Combined plot of simulated preamble detection with 1 to 30 UEs transmitting in the same RAO.*

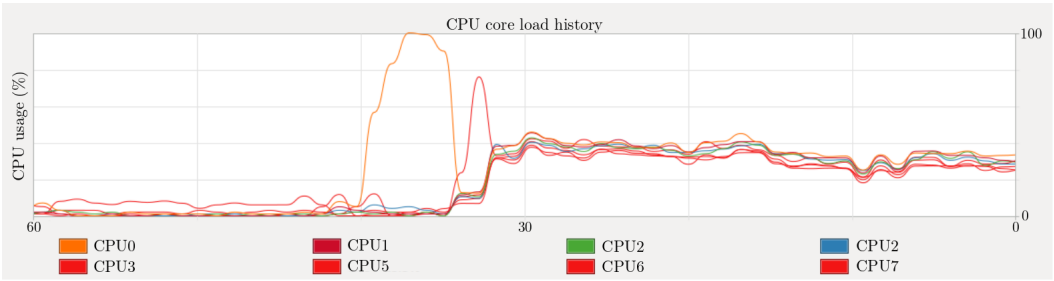
E. CPU load results

CPU load analysis plots from tests performed using Nokia and OpenAirInterface eNodeB.

(a) 4 MTC devices.



(b) 10 MTC devices.



(c) 15 MTC devices.

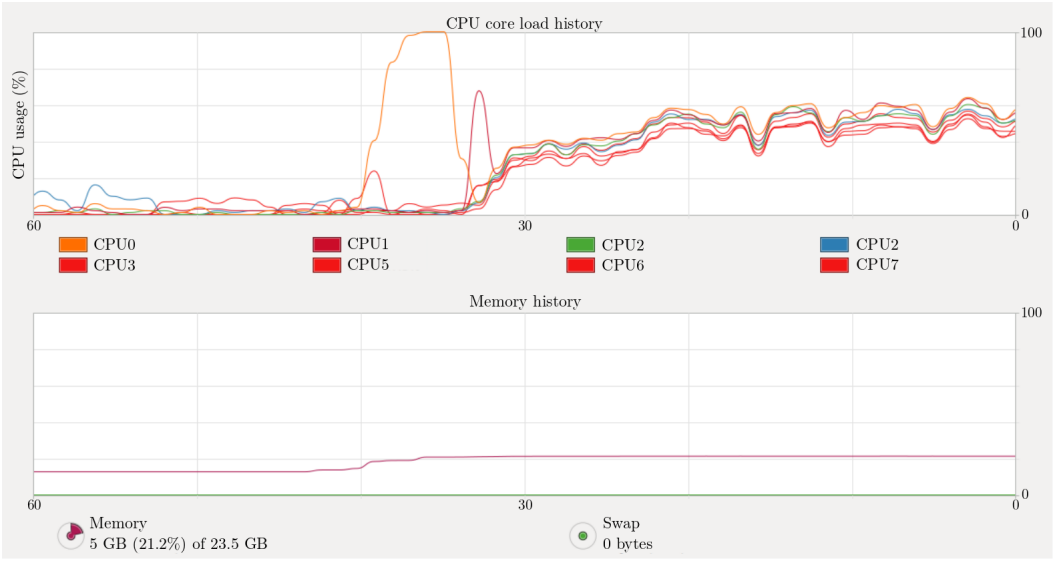
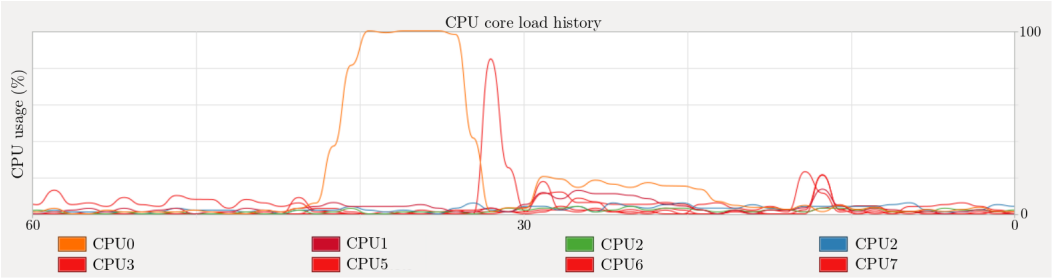
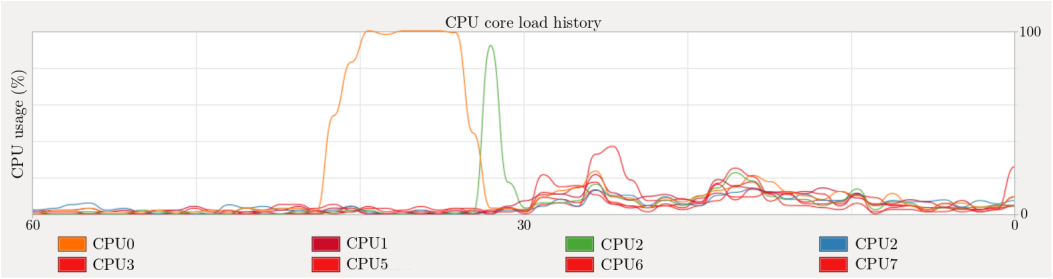


FIGURE E.1: *CPU load with N MTC devices in RRC connected state using the Nokia eNodeB at Telenor with a bandwidth of 5 MHz.*

(a) 2 MTC devices.



(b) 4 MTC devices.



(c) 8 MTC devices.

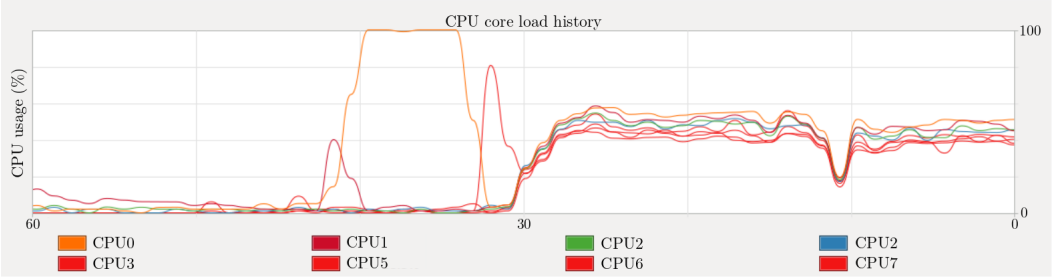
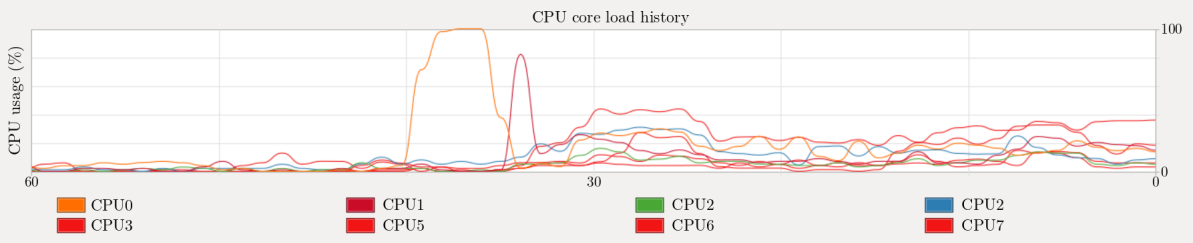


Figure E.2: *CPU load with N MTC devices in RRC connected state using the Nokia eNodeB at Telenor with a bandwidth of 10 MHz.*

(a) 2 MTC devices at 5 MHz.



(b) 2 MTC devices at 10 MHz.

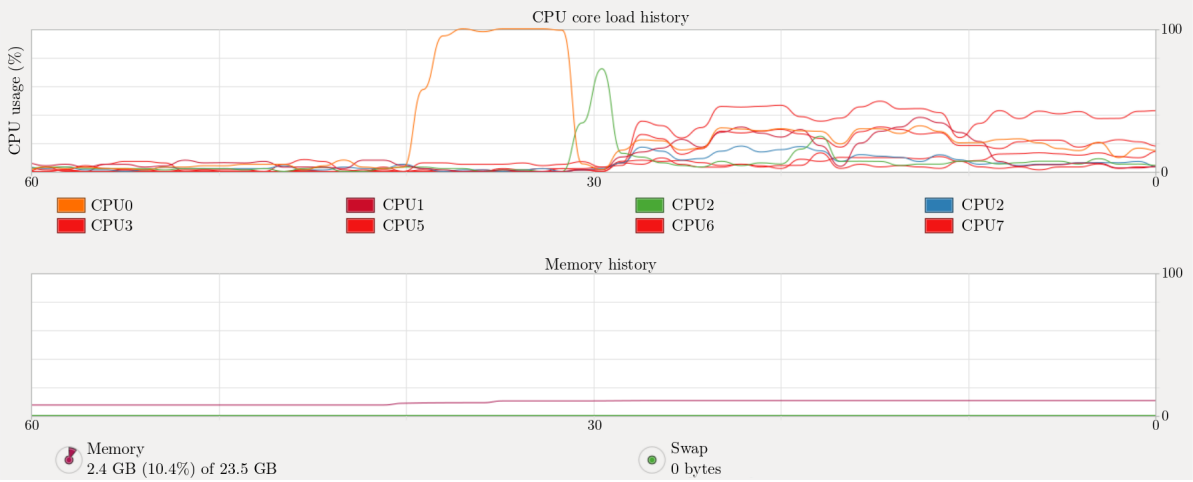


FIGURE E.3: *CPU load with 2 MTC devices connecting to OpenAirInterface eNodeB and EPC at 5 and 10 MHz, respectively.*

F. Physical layer tracing

The physical layer down- and uplink processes of the PHY layer of srsUE are traced by the order in which the functions are called in the physical channel worker each subframe (inPHY in the massiveMTC implementation); the trace is seen in figure F.1 and F.2 for down- and uplink, respectively. The most important function calls are traced through the srsLTE API to give an overview of the functionality of the software. The physical channel worker is responsible for both down- and uplink processing; an instance will start each subframe by decoding downlink information and pass data to the MAC layer, before performing the uplink processing. Downlink information must be decoded first, as uplink grants and ACKs or NACKs must be decoded before being able to decide on uplink transmission.

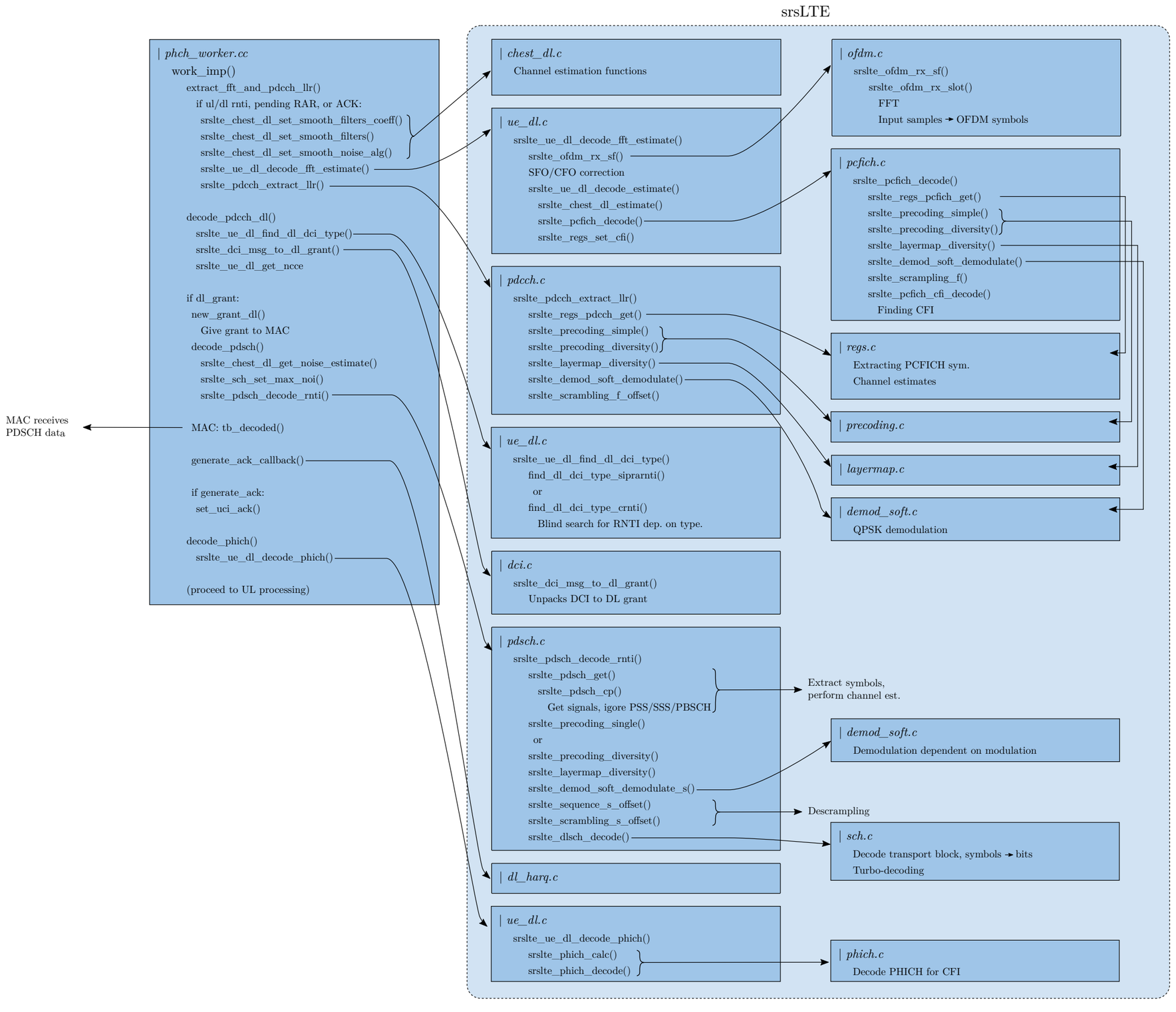


Figure F.1: *Downlink processing (PHY layer) function call tracing for srsUE.*

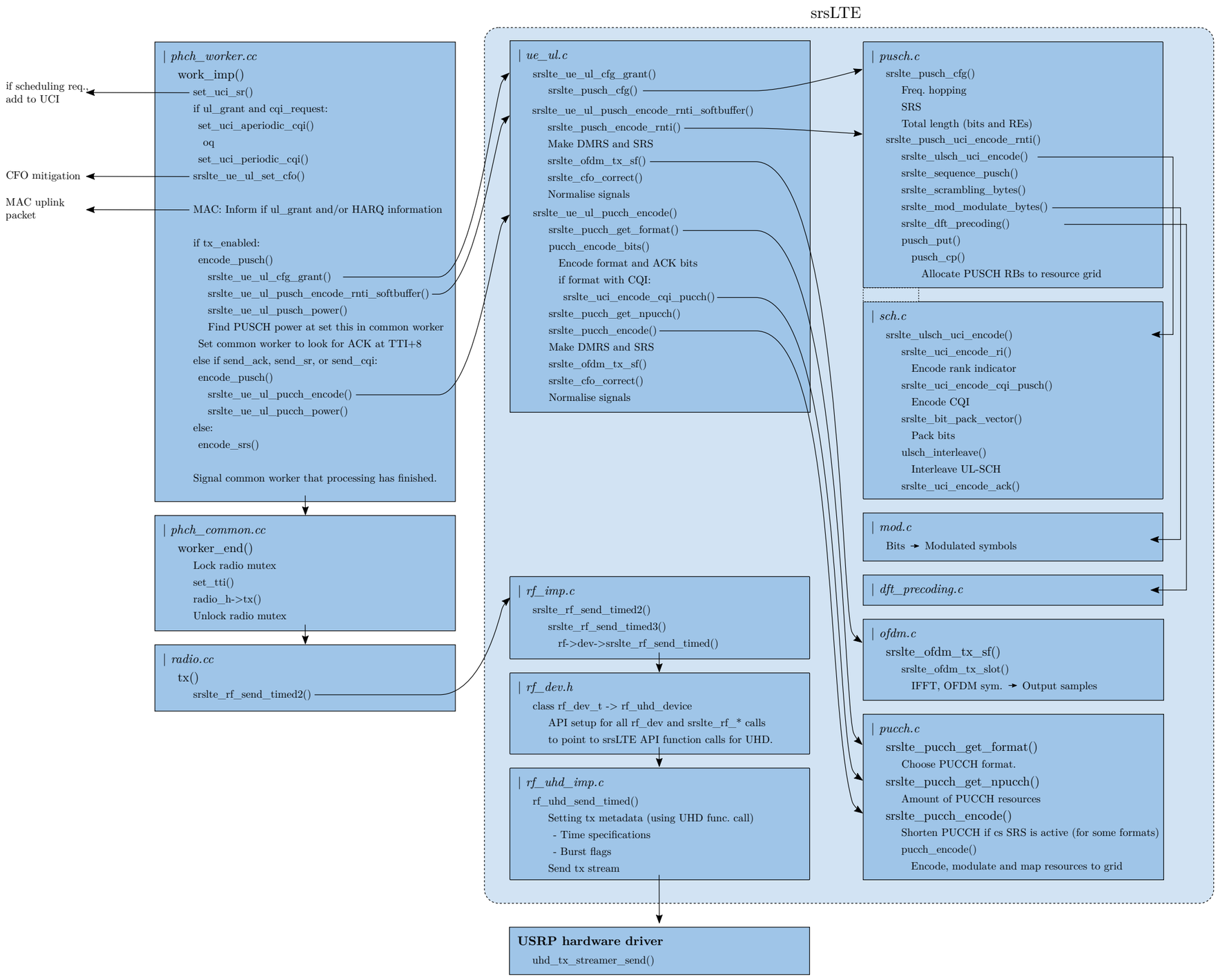


FIGURE F.2: *Uplink processing (PHY and radio layer) function call tracing for srsUE.*