

Network of Excellence

NEWCOM#

Network of Excellence in Wireless Communications#

FP7 Contract Number: 318306



WP2.3 – Flexible communication terminals and networks

D23.1

Open Lab access and experimental setups EuWin@CNRS/Eurecom and preliminary plan of activities

Contractual Delivery Date:	April 30, 2013
Actual Delivery Date:	April 30, 2013
Responsible Beneficiary:	CNRS/Eurecom
Contributing Beneficiaries:	CNRS/Eurecom, CTTC, CNIT/UniBo
Estimated Person Months:	8
Dissemination Level:	Public
Nature:	Report
Version:	1.0

PROPRIETARY RIGHTS STATEMENT

This document contains information, which is proprietary to the NEWCOM# Consortium.

This page is left blank intentionally

Document Information

Document ID:	D23.1
Version Date:	May 7, 2013
Total Number of Pages:	65
Abstract:	This deliverable introduces EuWin, the European Laboratory of Wireless Communications for the Future Internet, which is one of the main activities of Newcom#. In particular this deliverable describes the facilities established at EuWin@Eurecom and how they can be accessed in situ or remotely. Last but not least we present a preliminary plan of activities and policies for EuWin.
Keywords:	Wireless communications, real-time testbed, LTE, open-source, radio experimentation, prototyping, software defined radio, OpenAirInterface

Authors

IMPORTANT: The information in the following two tables will be directly used for the MPA (Monitoring Partner Activity) procedure. Upon finalisation of the deliverable, please, ensure it is accurate. Use multiple pages if needed. Besides, please, adhere to the following rules:

- **Beneficiary/Organisation:** For multi-party beneficiaries (CNIT) and beneficiaries with Third Parties (CNRS and CTTC), please, indicate beneficiary *and* organisation (e.g., CNIT/Pisa, CNRS/Supelec).
- **Role:** Please, specify: Overall Editor / Section Editor / Contributor.

Full Name	Beneficiary / Organisation	e-mail	Role
Florian Kaltenberger	CNRS/Eurecom	Florian.kaltenberger@eurecom.fr	Editor
Raymond Knopp	CNRS/Eurecom	Raymond.knopp@eurecom.fr	Contributor
Navid Nikaein	CNRS/Eurecom	Navid.Nikaein@eurecom.fr	Contributor
Roberto Verdone	CNIT/Bologna	roberto.verdone@unibo.it	Contributor
Miquel Payaro	CTTC	miquel.payaro@cttc.es	Contributor
Davide Dardari	CNIT/Bologna	davide.dardari@unibo.it	Contributor

Reviewers

Full Name	Beneficiary / Organisation	e-mail	Date
Roberto Verdone	CNIT/Bologna	roberto.verdone@unibo.it	April 29, 2013

Version history

Issue	Date of Issue	Comments
0.1	4.4.2013	First version
0.2	10.4.2013	Integrated with common parts from Track Leader
0.4	12.4.2013	Added input for remote access and details about Open-AirEmu
0.9	15.4.2013	Final draft to be delivered to the Newcom# board for revision
1.0	29.4.2013	Final version after internal revision.

Executive Summary

This document provides the status of the EuWin experimental facilities at the onset of the NoE and, in particular, detailed information on the EURECOM facilities. An overview of the three EuWin sites is given which focuses on the global objectives of EuWin, in terms of methodologies adopted for measurements, performance evaluation and dissemination. Specific attention is given to industrial relations established by the three sites in order to guarantee dissemination of key ideas to industry partners inside and external to the NoE. The initial EuWin website content and services are presented.

The EURECOM facility is described starting with the laboratory equipment at the disposal of the network (test and measurement equipment, vehicles for measurements, cell-site base-station equipment, etc.). The software libraries provided through OpenAirInterface.org (OAI) which will be enhanced through EuWin are outlined along with the software radio architecture. The two main experimental targets, namely 3GPP LTE and 802.11p, are described along with the different ways the platform can be instantiated, namely real-time MODEMs, unitary simulation in C or through MATLAB/OCTAVE interfaces, system simulator targets (oaisim). Description of remote-access capabilities using the ONELAB framework is also described.

Preliminary plans for experimentation and policies for accessing the facility are described. The most important outcome that stems from the contents of this deliverable is that EuWin@Eurecom is now ready to start the experimental activities of WP2.3.

Table of Contents

1. Introduction	8
1.1 Glossary	9
2. EuWIn Scope and Activities	12
2.1 Introduction	12
2.2 EuWIn Goals and Lines of Action	13
2.2.1 Real World Measurements	14
2.2.2 Real World Performance Evaluation	14
2.2.3 Links with Industries	14
2.2.4 Dissemination	14
2.2.5 Durable Integration	14
2.2.6 World Contests	15
2.3 EuWIn Activities	15
2.3.1 A Unique Portal	15
2.3.2 Web Meetings	15
2.3.3 Workshops	15
2.3.4 Training Schools	16
2.3.5 Industry Liaisons	16
2.3.6 Demonstration Activities	16
2.3.7 Experimental Tours	16
2.4 The Three EuWIn Sites	16
2.4.1 EuWIn@CTTC	16
2.4.2 EuWIn@UNIBO	17
2.4.3 EuWIn@EURECOM	18
2.5 Liaisons with Industries	18
2.5.1 EuWIn@CTTC	18
2.5.2 EuWIn@UniBO	19
2.5.3 EuWIn@EURECOM	19
2.6 The EuWIn Website	19
3. EuWIn Technical Facilities and at CNRS/EURECOM	21
3.1 Available facilities @ Eurecom	21
3.1.1 High-power multi-antenna cell site	21
3.1.2 Experimental vehicles	23
3.1.3 Lab equipment	24
3.1.4 Software radio platforms	24
3.1.5 OpenAirInterface LTE software modem	32
3.1.6 OpenAirInterface 802.11p modem	35
3.1.7 Collaborative tools	36
3.2 Interfaces	37
3.2.1 OpenAirInterface Matlab interface	37
3.2.2 OpenAirInterface unitary simulators	38
3.2.3 OpenAirInterface system level simulator	38
3.2.4 Remote access	39
4. Preliminary Plans of Activities and Policies	45
4.1 Introduction	45
4.2 Steps Followed	45
4.3 Plans of Activities	45

4.4	Access Policy	47
5.	Conclusions	49
6.	Appendix	50
6.1	ExpressMIMO2 Software Framework	50
6.1.1	Architecture	50
6.1.2	Driver and firmware update	50
6.1.3	C API	51
6.1.4	Octave API wrappers.....	51
6.2	Real-time operation of OpenAirInterface LTE software modem on ExpressMIMO2.....	52
6.2.1	ExpressMIMO data acquisition	52
6.2.2	Synchronization of the software modem	53
6.2.3	Compilation and usage	53
6.3	OpenAirInterface system level emulator.....	54
6.3.1	Experiment Design Workflow	54
6.3.2	Emulation Kernel	56
6.3.3	Software Architecture	57
6.3.4	Experiment 1: Analyze the control plane	59
6.3.5	Experiment 2: Analyze the Data plane	61

1. Introduction

EuWIn (the European Laboratory of Wireless Communications for the Future Internet) intends to become one of the durable and most successful outcomes of Newcom#. EuWIn fosters excellence in research in the field of wireless communications, and it aims at training a new generation of scientists prone to do research both from the theoretical and the experimental viewpoint.

The general theme of EuWIn is “**Fundamental Research Through Experimentation**”. This motto emphasizes that the laboratory is not targeted to demonstration activities (though they might be considered sometimes); rather, the laboratory aims at making theoretical and fundamental research closer to the real world, in terms of adherence to the true problems that any new technology has to face at time of implementation, and of realism of model assumptions. As additional scope, EuWIn will represent a sort of gateway among the community of scientists doing fundamental research in the field of wireless communications, and industry.

Despite the intention to become a long-lasting initiative, EuWIn is described in this document only in terms of its plan of activities to be performed within the Newcom# course. EuWIn represents the experimental Track (Track #2) of Newcom#. EuWIn activities are devised as facilitators for the research performed within Track #1. Some of the EuWIn facilities will support the Joint Research Activities (JRAs) conducted within the Track #1 WPs. Inter-Track JRAs will be defined accordingly. On the other hand, EuWIn aims at creating a durable and self-sustainable environment that will survive after the end of Newcom#; therefore, all means to establish a structured integrated laboratory will be pursued and are described in this Deliverable.

EuWIn is composed of three sites, targeting at separate technologies and fields of experimentation: the laboratories of the research center CTTC of Barcelona (Spain), of CNIT/University of Bologna (Italy), and of the EURECOM institute of Sophia-Antipolis (France). The three institutions had developed experimental facilities in the context of other projects in the past years, and have committed to make them available to other Newcom# partners, through an integrated and open framework.

Additional Newcom# institutions, besides the three that committed towards the creation of an open laboratory context, have expressed the intention to contribute through some local experimental facilities to EuWIn: Bilkent, CNRS, PUT, UCL and UPC (third party of CTTC).

The three EuWIn sites cover aspects related to radio interfaces (mainly based on MIMO and PHY-layer algorithms) and localization techniques, at CTTC, flexible radio technologies over MIMO platforms, at CNRS/EURECOM and Internet of Things (IoT) and Smart City applications, at CNIT/UniBO. Overall, accounting for all institutions participating to Track #2, the experimental activities will deal with aspects of modulation, coding, signal processing, localization, radio channel characterization, routing, topology control.

The main scopes of EuWIn, the tools used to achieve integration within the sites, and the facilities available at each of the three sites, are discussed in Section 2 of this Deliverable. The Section also includes a short presentation of the EuWIn website, which is described in more detail in D36.1. In Section 3 the lab equipment made available at EURECOM is presented, and the objectives of the testbeds are clearly identified; two additional parallel Deliverables report on the details related to the CTTC and CNIT/University of Bologna sites. The general parts of the three Deliverables are common, to emphasize the unique scope and the integrated strategy, while Section 3 is different for each of the three Deliverables. Section 4 shows the plan of EuWIn activities. It is detailed for the next six months, while the plans for the following two years will be described in the Deliverables due at M12. The Section also

presents the policy for access to the lab facilities. The conclusions shortly discuss on the potential role of EuWin within the scientific community and for the European industry context.

1.1 Glossary

3GPP	Third generation Partnership
ACK	Acknowledgement
ACLR	Adjacent channel leakage ratio
ADC	Analogue digital convertor
AM	Acknowledged Mode
AMBA	Advanced Microcontroller Bus Architecture
ASIC	Application Specific Integrated Circuit
AVCI	Advanced VCI
BB	Baseband
BCH	Broadcast Channel
CCCH	Common Control Channel
CFI	Control Format Indicator
CPU	Central Processing Unit
CQI	Channel Quality Indicator
CRC	Cyclic Redundancy Code
CTS	Clear to send
DAB	Digital Audio Broadcast
DAC	Digital analogue convertor
DCCH	Dedicated Control Channel
DCI	Downlink Control Information
DD	Digital Dividend
DDR	Double data rate
DLSCH	Downlink Shared Channel
DMA	Direct memory access
DoW	Description of Work
DSP	Digital Signal Processor
DTCH	Dedicated Transport Channel
EC	European commission
eNB	Enhanced Node B
EuWin	European Laboratory of Wireless Communications for the Future Internet
FDD	Frequency division duplex
FE	Front-end
FFT	Fast Fourier Transform
FPGA	Field programmable gate array
GPIO	General Purpose Input Output
GSM	Global System for Mobile Communications
HARQ	Hybrid Automated Repeat Request
HI	Hybrid ARQ indicator
IEEE	Institute of Electrical and Electronics Engineers
IoT	Internet of Things

IP	Internet Protocol
JRA	Joint Research Activities
LNA	Low noise amplifier
LTE	Long term evolution
LXRT	Linux Real Time
MAC	Medium access control
MIMO	Multiple-Input Multiple-Output
NAK	Non-Acknowledgement
NAS	Non Access Stratum
NFS	Network file system
NITOS	Network Implementation Testbed using Open Source code
OAI	OpenAirInterface
OEDL	OMF Experiment Description Language
OFDM	Orthogonal Frequency division multiplexing
OMF	cControl and Management Framework
OS	Operating System
OSD	OpenAir Scenario Descriptor
PA	Power amplifier
PBCH	Physical Broadcast Channel
PC	Personal Computer
PCFICH	Physical Contrl Format Indicator Channel
PCI	Peripheral Component Interconnect
PCIe	PCI express
PDCCH	Physical Downlink Control Channel
PDCP	Packet Data Convergence Protocol
PDSCH	Physical downlink Shared Channel
PDU	Packed Data Unit
PHICH	Physical Hybrid-ARQ Indicator Channel
PHP	Hypertext Preprocessor
PHY	Physical
PMI	Precoding Matrix Indicator
PRACH	Physical Random Access Channel
PSS	Primary Synchronization Signal
PUSCH	Physical Uplink Shared Channel
RA	Random Access
RB	Resource Block
RF	Radio frequency
RI	Rank Indicator
RLC	Radio Link Controller
RRC	Radio Resource Controller
RRM	Radio Resource Manager
RTAI	Real-time Application Interface
RTS	Ready to send
RX	Receiver

SIMD	Single Instruction Multiple Data
SMA	SubMiniature version A
SNR	Signal to Noise Ratio
SRS	Sounding Reference Signal
SSS	Secondary Synchronization Signal
SVN	Subversion
TB	Transport Block
TDD	Time division duplex
TM	Transmission Mode
TVWS	Television white space
TX	Transmitter
UCI	Uplink Control Information
UE	User Equipment
ULSCH	Uplink Shared Channel
UM	Unacknowledged Mode
UMTS	Universal Mobile Telecommunications system
VCI	Virtual Component Interface
WCDMA	Wideband Code division multiple access
WiMAX	Worldwide Interoperability for Microwave Access

2. EuWIn Scope and Activities

This Section describes the general goals of EuWIn, the types of activities planned and the policy for accessing its facilities. Short introduction to the EuWIn website structure is also given.

2.1 Introduction

As mentioned in Section 1, the three EuWIn sites deal with separate technical and scientific topics: Radio Interfaces (CTTC), IoT (UniBo) and Flexible Communication Terminals (EURECOM). However, the scope of EuWIn is to achieve an integrated laboratory able to address under a common environment the various topics of wireless communication technologies for the future Internet. To reach this goal, on the one hand the sites have been identified based on the complementarity of their competencies and facilities; on the other, some technical topics are common to two EuWIn sites, thus facilitating the integration of competencies and procedures: localization techniques are a common topic for CTTC and UniBO; MIMO based platforms for testing purposes are available at both EURECOM and CTTC; heterogeneous architectures and technologies for the IoT are considered both at UniBO and EURECOM sites.

All Newcom# institutions will be allowed to take advantage of this integrated (yet distributed) laboratory, through an open access policy described below. The procedures for the coordinated access to the lab facilities are also introduced in this Deliverable, and implemented through the EuWIn website. Newcom# associate partners will also be allowed to have access under agreed conditions, while institutions external to the Newcom# environment will undergo specific and selective agreements as described below.

The vision of EuWIn can be represented through the spheres shown in Figure 2-1.

The inner sphere includes the three EuWIn sites. The integration of facilities and coordination of activities is under the responsibility of the EuWIn Director, Roberto Verdone (CNIT/UniBO). The three institutions, as detailed later, have been working during the first semester of Newcom# to extend and integrate their pre-existing facilities, creating the tools for their use by the rest of the Network of Excellence.

The second “internal” sphere includes all Newcom# institutions (the figure emphasizes those who, at time of DoW preparation, expressed commitment towards EuWIn. During the three years of Newcom# the set of these institutions might change). All Newcom# institutions have access to the EuWIn facilities, according to the open access policy defined later, free of charge.

The third “internal” sphere includes the associate partners of Newcom#, both those defined at time of DoW preparation (mainly industries) and those accepted by the consortium after project kick off. These institutions have access free of charge to the EuWIn facilities based on a selective process/policy described later.

Finally, the “external” sphere includes all institutions interested in having access from outside Newcom# to the EuWIn facilities. This sphere is based on a case-by-case consideration of the access conditions, which will depend on the type of activity foreseen, the facility/site of interest, etc.

At time of delivery of this document, the following Newcom# partners are involved in potential JRAs targeted at the exploitation of the EuWIn facilities: AAU, Bilkent, CNRS, IASA, PUT, UCL, VUT, Inov, Moreover, the following additional institutions from outside Newcom# have declared strong interest into the EuWIn WPs: ACTIX, Agilent, AVEA, U-Blox, NEC Labs, Orange Labs, Renesas Mobile, Samsung, Telecom Italia, Telefonica I+D, Thales Communications.

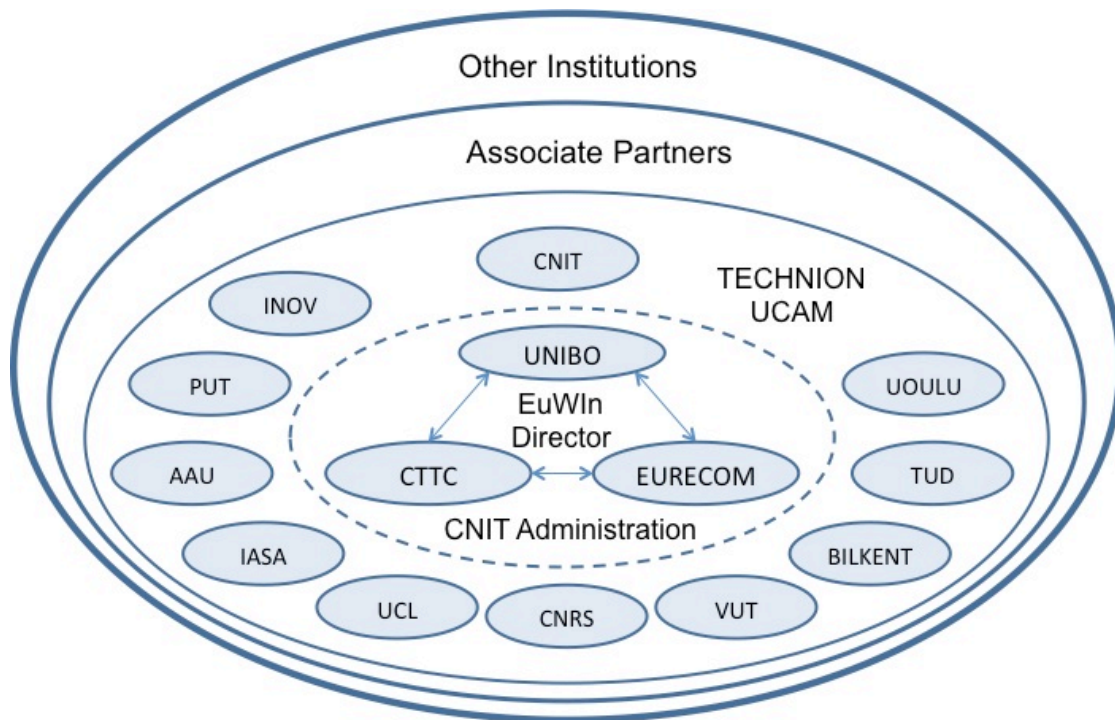


Figure 2-1: The Newcom# Spheres.

The EuWIn website will specify and publicize for each sphere the access policy and the procedures for a coordinated exploitation of the laboratory facilities.

2.2 EuWIn Goals and Lines of Action

The uttermost goal of EuWIn is the creation of a new generation of scientists willing to perform fundamental research through experimentation. Additionally, EuWIn, being one of the Tracks of Newcom#, aims at an active role within the Network of Excellence as an instrument for dissemination and link with industries.

These goals can be reached in the long term by developing instruments and activities which will allow the involvement, at different levels, of the largest possible number of researchers, from inside and outside Newcom#. EuWIn will therefore pursue its goals through the implementation of several types of actions:

- 1) **Real world measurements** helpful for fundamental research activities.
- 2) **Real world performance evaluation** of fundamental research results.
- 3) **Links with industries**, either associated to NEWCOM# or not.
- 4) **Dissemination** of NEWCOM# research outcomes.
- 5) **Durable integration** of resources and activities provided by NEWCOM#.
- 6) **World contests** open to all researchers, within and outside NEWCOM#.

These lines of action are discussed briefly in the following subsections.

2.2.1 Real World Measurements

Real world measurements will be useful in order to extract from them models and parameters to be used in theoretical research. Radio channel characterisation, for instance, is of outmost relevance to scientists studying the link-level performance of wireless systems; human mobility modelling, link-level assessment of the bit error rate against SNR (signal-to-noise-ratio) performance, are examples of research activities which can take advantage of measurement campaigns and are useful to the evaluation of the performance of wireless networks. All the EuWIn sites (and other Newcom# institutions) have facilities which will allow the development of measurement campaigns aimed at the definition of suitable models to be used for the performance of wireless systems, either at link or network level, as specified later. This line of activity indeed will provide *inputs* to the theoretical study of link and network level performance of wireless systems, performed within Track #1, making them more realistic in terms of models used.

2.2.2 Real World Performance Evaluation

Real world performance evaluation is a task performed in research either through mathematical modelling, simulation, emulation or experimentation; when the system under analysis is complex, the latter approach is the only one that is feasible. It requires flexible experimental facilities allowing the implementation of the link-level or network-level techniques/algorithms under evaluation. The EuWIn sites have developed facilities that are flexible enough in order to allow implementation of different techniques/algorithms designed by scientists performing theoretical research. This line of action therefore, will take the *outputs* of the research performed within Track #1, and provide assessment of the performance through experimentation.

2.2.3 Links with Industries

EuWIn will represent the main Newcom# instrument to create links with industries; the lab facilities will be used as proofs-of-concept of the techniques/algorithms designed, studied and evaluated within Track #1. Representatives from industry will be invited at all EuWIn events, and the researchers involved in the JRAs performed within EuWIn will organise visits at the premises of companies. The industry-academia link, which was missing in its predecessor Newcom++, will be established within Newcom# thanks to the availability of the EuWIn facilities and testbeds.

2.2.4 Dissemination

All EuWIn dissemination events, either intended for industry or academia, will have at least one of the EuWIn sites present, showing the facilities, making them available remotely, and reporting on the latest achievements. EuWIn leaflets, posters and video trailers will be generated and made available to all EuWIn researchers with the aim of creating a repository of promotional material to be used for dissemination purposes.

2.2.5 Durable Integration

EuWIn intends to create a framework which will go beyond the duration of the Newcom# project. To this aim, all coordination, promotional and dissemination activities will be geared towards a longer horizon than the three years of expected duration of Newcom#. However,

durable integration requires financial sources which should be found outside the Newcom# budget. EuWIn will seek for additional sources of funds through the participation of the EuWIn institutions to competitive EC calls, and from the private sector (industries willing to access the EuWIn facilities).

2.2.6 World Contests

Some of the EuWIn sites (among them, the one at UniBO) will launch international contests to test over the lab facilities techniques/algorithms designed by institutions willing to participate. This will create further visibility to the laboratory, while representing a potential source of incomes for future activities after the end of Newcom#.

2.3 EuWIn Activities

The lines of action described in the previous subsection take the form of activities performed within Track #2 as it follows:

- 1) **A Unique Portal:** facilities, activities and repository of measurements;
- 2) **Web Meetings,** to coordinate and plan visits of researchers;
- 3) **Workshops** on experimental research;
- 4) **Training Schools** on experimental research;
- 5) **Industry Liaisons;**
- 6) **Demonstration activities:** at conferences, fairs, etc;
- 7) **Experimental Tours:** PhD students visiting the three sites in sequence.

Short description of the activities is given in the following subsections.

2.3.1 A Unique Portal

Under the domain euwin.org, a website has been created (properly linked to the Newcom# website). The scope of the portal is to create a unique access to the laboratory, by providing shared documents, description of procedures for getting access to the facilities, and a repository of those measurement data that are shared and can be useful to the researchers involved in EuWIn activities. The website will also provide information on the events generated or participated by EuWIn, the contacts had with industries, etc.

2.3.2 Web Meetings

Besides the usual web meetings held periodically by the three WP Leaders of Track2, EuWIn will organise web meetings open to all EuWIn participants in order to disseminate within the project information regarding the latest development of the lab facilities, to coordinate and schedule the access and the activities of the EuWIn-related JRAs. These web meetings will be called by the EuWIn Director depending on needs and requests, and the dates will be fixed through polls, informing all researchers at least 30 days before the meeting will take place.

2.3.3 Workshops

To foster sensibility towards experimentation, EuWIn will organise workshops, with the contribution of both Newcom# researchers and scientists participating to other projects, which will aim at discussing the benefits of experimental research and compare the approaches toward open access platforms.

2.3.4 Training Schools

One of the most important activities performed within EuWIn, apart from the experimental trials, is the organisation/contribution to training schools. Through the participation of EuWIn key researchers as lecturers and the organisation of experimental sessions at the schools, the attendees will be trained towards experimental research.

2.3.5 Industry Liaisons

Each of the EuWIn sites will establish liaisons with specific industrial labs and companies, identified based on the topic of interest of the site. The EuWIn sites will be then responsible for the organisation of periodic meetings with the industry representatives in order to create stable contacts that might bring to forms of support to the EuWIn activities in the long term.

2.3.6 Demonstration Activities

The achievements got within Newcom# and the development of new facilities, will be demonstrated at fairs, exhibitions and other events participated by many industry representatives.

2.3.7 Experimental Tours

The most challenging objective of EuWIn is the creation of a generation of researchers prone to fundamental research through experimentation, with competencies spanning over different layers of the protocol stack. This goal can be achieved by letting PhD students get involved into activities performed over the experimental facilities of different EuWIn sites, dealing with separate scientific topics. These activities will be denoted as “experimental tours”, as the students will visit different sites, performing experiments that will be properly coordinated in order to achieve a unique scientific goal through the integration of measurements achieved over separate testbeds.

2.4 The Three EuWIn Sites

2.4.1 EuWIn@CTTC

The CTTC site (www.cttc.es) features the following three high performance testbeds, which are the core of the EuWIn@CTTC assets:

a) GEDOMIS® (GEneric hardware DemOnstrator for MIMO Systems): GEDOMIS® is an experimental platform that comprises a complete set of high performance baseband prototyping boards (FPGA and DSP-based), signal generation equipment, high-end RF front-ends, signal analysis instruments, specialized software tools and APIs. A key instrument of the testbed is the EB PropSim C8 channel emulator, which can be configured to provide realistic mobility scenarios of both certified and user-designed channel models. The PropSim C8 operates in real-time and facilitates the verification and testing of system designs prior to field-trials. Hence, GEDOMIS® offers a lab environment that enables the prototyping of next generation broadband wireless communication systems, which feature bit-intensive OFDM PHY-layer techniques possibly combined, if desired, with multi-antenna schemes. Moreover, the prototyping boards of GEDOMIS® feature various energy consumption measurement blocks, which can be used to assess the energy-efficiency for a given hardware implementation.

b) GNSS-SDR (Global Navigation Satellite System - Software Defined Radio): GNSS-SDR is an open source GNSS software defined receiver that addresses the design and implementa-

tion of multi-constellation, multi-band GNSS receivers. According to its open nature, the source code is published under General Public License (GPL) v3, which secures practical usability, inspection, peer-review and continuous improvement by the research community, allowing the discussion based on tangible code and the analysis of results obtained with real signals. In addition to the source code, the project offers a development ecosystem (website, code repository, mailing list, bug tracker) that aims to build a community of researchers, developers and users around the project. The ultimate goal is to provide high-accuracy positioning for the masses, and for free. There are many ways for researchers to contribute to the project, e.g., programming new features (algorithms, receiver architectures, etc.), optimizing existing algorithms, reporting bugs or debugging existing features, among many others. This facility's equipment comprises an RF signal generator (Agilent's Vector Signal Generator E4438C equipped with GPS Personality), an assortment of GNSS antennas, RF measurement equipment (oscilloscope, network analyzer, spectrum analyzer), a set of Universal Software Radio Peripherals (V1 and V2) and USB data grabbers.

c) **OpenInLocation** (an open lab for research on indoor location): The main objectives of this lab is to drive R&D and innovation in indoor location by evaluating state-of-the-art and commercial technologies in challenging environments, improving them in terms of enabling High-Definition Situation-Aware (HDSA) applications and services, and, finally, fostering collaboration to contribute with new ideas for indoor positioning, to disseminate results and outreach to new audiences beyond Newcom#. This lab is composed of a set of extremely low-cost, programmable nodes that can act either as anchor node or as a mobile node, the wireless interfaces for such nodes (WiFi, Bluetooth, ZigBee, etc.), sensors (in IC form), and the corresponding software repository.

The common features to these three labs is that they promote collaborative research by offering an adequate framework and tools and they are all based on the principle of reproducible research, which enables practical discussions on tangible problems.

2.4.2 EuWin@UNIBO

The EuWin@UNIBO site provides more than 200 wireless nodes implementing different types of radio interfaces, and deployed according to the following platforms:

Flexible Topology Testbed (FLEXTOP) – 100 nodes equipped with IEEE 802.15.4 radios distributed inside the lab over a regular layout creating an open and flexible platform for emulating several types of network topologies, achieved through proper software setting of the inter-node losses. This platform is based on over-the-air (OTA) implementation of software, meaning that NEWCOM# partners will be given remote access to implement the software;

Data Sensing and Processing Testbed (DATASENS) – An infrastructure of approximately 100 nodes using IEEE 802.15.4 radios deployed inside one of the University buildings, in rooms and corridors, implementing sensors able to monitor/emulate physical instances like light intensity, temperature, current drain, equipped with small LCD screens that will ease the management and control of application tasks. In addition to the nodes composing the fixed infrastructure, approximately 50 battery-supplied mobile nodes (with 802.15.4 devices) carried by people moving around, are part of the testbeds (Roaming Nodes);

Localization Testbed (LOCTEST) - Approximately 50 additional nodes with high-accuracy localization capabilities, using IEEE 802.15.4a or IEEE 802.15.4f Ultra-Wide Bandwidth (UWB) radios, will be deployed in the same building depending on the future commercial availability of low-cost UWB-based devices;

The infrastructure also includes some network coordination nodes equipped with mini-PC boards and different types of gateways (WLAN, LAN) for proper interconnection to the Internet.

2.4.3 EuWin@EURECOM

EURECOM (www.eurecom.fr) is a research and teaching institute located in the Sophia Antipolis Technopole in the South of France. EURECOM's research activity traditionally focuses on 3 domains which have led to the creation of three research departments: Networking & Security, Multimedia Communications, and Mobile Communications. The Mobile Communications Department has its focus on digital signal processing for mobile communications, information theory, 4th generation and clean-slate cellular radio systems, wireless protocols, mobile ad hoc networks, software radios, and SW and HW prototyping. In addition to fundamental research, the department has strong expertise in open-architecture multi-way real-time radio platforms (www.openairinterface.org) for use in publicly-funded research projects aiming at demonstrating innovations at all protocols layers (RF to applications). The latter is provided to EuWIN for use within the network and afterwards. The OpenAirInterface.org (OAI) development and facilities primarily target cellular technologies (LTE/LTE-Advanced and beyond) and rapidly-deployable mesh/ad-hoc networks. The platform comprises both hardware and software components and can be used for simulation/emulation as well as real-time experimentation. It comprises the entire protocol stack from the physical to the networking layer. The objective of this platform is to provide methods for protocol validation, performance evaluation and pre-deployment system test.

2.5 Liaisons with Industries

2.5.1 EuWin@CTTC

Currently, the CTTC team involved in WP2.1 ("Radio interfaces for next-generation wireless systems") has links with many industry players, among which the most relevant follow:

AT4 Wireless (recently acquired by Agilent Technologies): This collaboration started in 2009, when the CTTC team started to develop, together with the AT4 Wireless team, the test platform E2010, which is designed to perform compatibility tests to LTE terminals. The experience gained through this collaboration has contributed to a knowledge transfer, which is foreseen to be very useful in Task 2.1.4 ("High spectrally-efficient radio interfaces").

AVIAT Networks: This collaboration started in 2012. CTTC is carrying out a series of projects for the Silicon Valley-based company AVIAT Networks. Among these projects, the most relevant for Newcom# is the project where CTTC is providing a baseband processing solution for the improvement of energy efficiency in new generation transmitters for wireless backhauling equipments. This collaboration can have a positive impact on CTTC's participation in Task 2.1.2 ("Low-energy-consumption and low-emission radio interfaces"). Moreover, within this same topic, CTTC has submitted a proposal in the Seventh Framework Programme together with the Slovenian branch of AVIAT Networks.

Nutaq (before Lyrtech RD): This collaboration started in 2012. Nutaq is currently commercializing a software IP that contains the physical layer of the WiMAX standard developed by CTTC, which is named BeMIMOmax (<http://nutaq.com/public/files/products/bemimomax/bemimomax.pdf>). The CTTC team is now adapting this IP to make it compatible with the LTE standard. If other IPs are obtained as a result of the experimental research performed within Newcom#, CTTC can study together with Nutaq and the other partners involved in the research the possibility of commercializing them.

2.5.2 EuWIn@UniBO

The EuWIn site in Bologna has links with many industries, owing to the participation to many EC projects and industrial contracts. The main industry links, related to the scopes of EuWIn, are mentioned here:

the FLEXTOP testbed developed within EuWIn is of interest to the research labs of Telecom Italia, with particular reference to IoT applications;

EuWIn@UniBo has a peculiar link with Embit srl, a small Italian company which is design centre for Texas Instruments and Freescale and the provider of all wireless devices implemented within the laboratory;

the EuWIn site in Bologna has a preferred link with Intel labs of Ireland, which also investigate smart city applications of wireless communications.

2.5.3 EuWIn@EURECOM

OAI has explicit links with industrial partners who contribute to its development both within the context of publicly-funded projects and out of mutual interest on specific research subjects. Currently, the main industrial collaborators, which are, incidentally, affiliate members of the NoE, include:

Alcatel Lucent Bell-Labs (Villarsaux, France) contribute to optimized real-time signal processing for SDR, innovative protocol extensions to LTE, emulation tools based on OAI, interfacing of OAI to industrial-grade RF platforms (remote radio-heads - RRH).

Orange Labs (Beijing, China, subdivision of Orange France) contribute to optimized real-time signal processing for SDR, interfacing of OAI to other hardware platforms (custom and USRP).

Agilent (Beijing, China) contribute to optimized real-time signal processing for SDR.

Thales Communications (Gennevilliers, France) contribute to development of multi-hop protocols and resource allocation strategies for rapidly-deployable mesh networks and innovative signal processing for improving RF amplifier efficiency in terminals.

2.6 The EuWIn Website

The EuWIn website has been designed using the same platform chosen for the Newcom# portal, i.e., Joomla. This makes their maintenance easier. It has been devised both as an instrument for facilitating cooperation among the institutions involved (through a private part used as repository of shared documents and measurement databases), and as a dissemination tool providing all the information needed to join the laboratory activities.

However, it is important to emphasise that the EuWIn website does not represent a duplication of the information reported in the Newcom# portal, and all official documents related to the project will be found only on the Newcom# portal. Therefore, deliverables and meeting minutes will be posted on the latter, whereas promotional material (leaflets, video trailers, etc.) and documents describing in detail the lab facilities, will be found on the EuWIn website.

Moreover, the EuWIn website is integrated in the Newcom# portal, as all user credentials made available to Newcom# researchers to access the project portal will be imported on the EuWIn site, so that all Newcom# researchers will be given access to the private area of the EuWIn website under the same credentials.

The structure devised for the EuWIn website is as follows.

HOME	it contains a short welcome message and describes the goals of EuWIn
STRUCTURE	description of the lab structure is given, using contents of this Deliverable
→ Sites	map, link to three institution websites
→ Director	short descript of role, personal welcome message
→ WP Leaders	short descript of roles, personal welcome messages
→ Partners	mention of Newcom# partners active within EuWIn
→ Liaisons	mention of the industries/labs/projects linked to EuWIn activities
FACILITIES	general description of the various platforms
→ Platform#1	(eg GEDOMIS) description
→ Platform#2	(eg FLEXTOP)
→ Platform#3	(eg DATASENS)
→ Platform#4	(eg LOCTEST)
→ ...	
ACCESS	the general access policy is described
→ Documents	link to public documents
→ Join	forms to be filled for requesting access
→ Visit	forms to be filled for proposing a visit to the laboratory sites
DEMOS	inaugural video and images
→ one per demo realised,	with some info (where, who, picture, what facility)
WORKSHOPS	description of the philosophy behind the organisation of workshops
→ one per workshop organised,	contributed (where, programme, what facility)
SCHOOLS	description of the philosophy behind the organisation of schools
→ one page per school organised,	contributed (where, programme, what facility)
REPOSITORY	a username/password form to be filled for restricted access
CONTACTS	

Concerning the repository, whose access will be restricted to Newcom# partners and affiliate institutions, it was agreed by the WP Leaders that it will contain the outcome of the measurement campaigns performed within the Newcom# JRAs, and the documents needed to coordinate/facilitate access to the lab facilities.

The first version of the website, released at M6, will replace the three pages DEMOS, WORKSHOPS, SCHOOLS, with one single page EVENTS. More details regarding the first version of the EuWIn website are found in D3.6.1.

3. EuWIn Technical Facilities and at CNRS/EURECOM

3.1 Available facilities @ Eurecom

Facilities at EURECOM that will be made available to the project include: (i) one fixed high-power multi-antenna cell site (800 MHz TDD/FDD, 1.9 GHz TDD, 2.6 GHz TDD/FDD) connected to the laboratory network. Other carrier frequencies will likely be made available during the course of the project because of regulatory constraints; (ii) two vehicles equipped with OpenAirInterface.org radio equipment which can be used either as secondary cell sites or terminals; (iii) high-end laboratory test and measurement equipment and design tools (LTE compliant signal/spectrum analyzers and signal generators, Xilinx/Mentor tool suites); (iv) OpenAirInterface.org software-radio platforms (3 ExpressMIMO and 10 Express-MIMO2); (v) Linux clusters with remote access possibilities for large-scale system emulation; and (vi) SVN repositories with external access.

3.1.1 High-power multi-antenna cell site

The following table illustrates the specifications for the eNodeB.

TX transmitted power	+25 dBm (per path) in the TVWS (734 – 758 MHz) +30 dBm (per path) in the 1.9 GHz band +30 dBm (per path) in the 2.6 GHz band
Noise figure	6 dB
Operated TX frequency band	TVWS : 734 – 758 MHz (TDD) 1.9 GHz band: 1900 – 1920 MHz (TDD) 2.6 GHz band: 2500 – 2570 MHz (FDD)
Operated RX frequency band	TVWS : 734 – 758 MHz (TDD) 1.9 GHz band: 1900 – 1920 MHz (TDD) 2.6 GHz band: 2620 – 2690 MHz (FDD)
Maximum input power	-30dBm
Maximum attenuation between antenna and PA-LNA module (due to the 20 m cables)	3 dB

An overview of the eNodeB architecture is depicted in Figure 3-1. Here we show the combination of TVWS with the 2.6GHz band. Other combinations are however possible. In detail the eNodeB parts comprise:

- **3 antenna sub-systems**, one for the 1.9GHz band, one for the 2.6 GHz band and one for the TVWS band. The 2.6 GHz antenna sub-system comprises 1 antenna, the TVWS antenna sub-system comprises 2 antennas, the 1.9GHz antenna sub-system provides up to 4 antennas (currently only 2 antennas are useable). The antenna is located on the top of Eurecom's building as depicted in Figure 3-2.
- The antenna subsystems are connected through RF cables to an RF equipment to be installed in the control room beneath the mast, namely the **PA-LNA Subsystems**. The aim of those elements is to amplify and filter the signal of interest both on the RX and TX side. The overall hardware in the control room is depicted in Figure 3-3.
- The PA-LNA Subsystems are connected to the **RF/baseband Conversion** box. The aim of this part is to convert the RF signal into base band signal. The RF/baseband Conversion box is based on LIME LMS6002D evaluation boards. LMS6002D is a fully integrated multi-band, multi-standard single-chip RF transceiver for 3GPP (WCDMA/HSPA and LTE), 3GPP2 (CDMA2000) and WiMAX applications.
- The RF/baseband Conversion box is connected to a distribution board (which aims to put all the RF signals and controls into one single connector to ExpressMIMO) and to

ExpressMIMO. ExpressMIMO is a FPGA-based board dedicated to Base Band processing. The protocol stack of the eNodeB is done in a **host PC**, connected through a PCIeexpress interface.

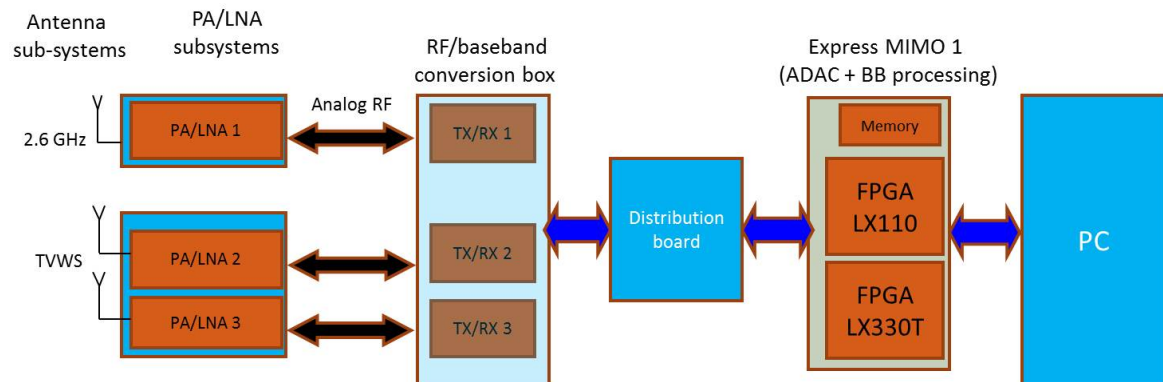


Figure 3-1: Block diagram of the eNodeB equipment

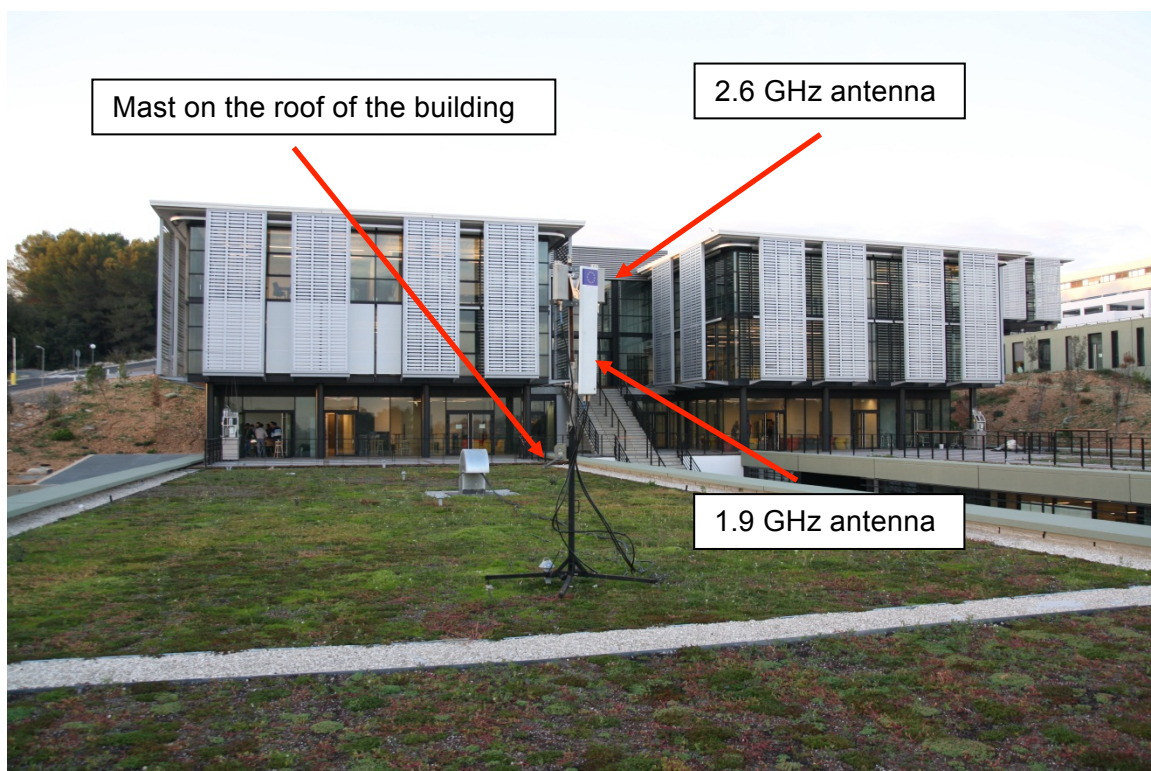


Figure 3-2: Mast on the Eurecom building, front view

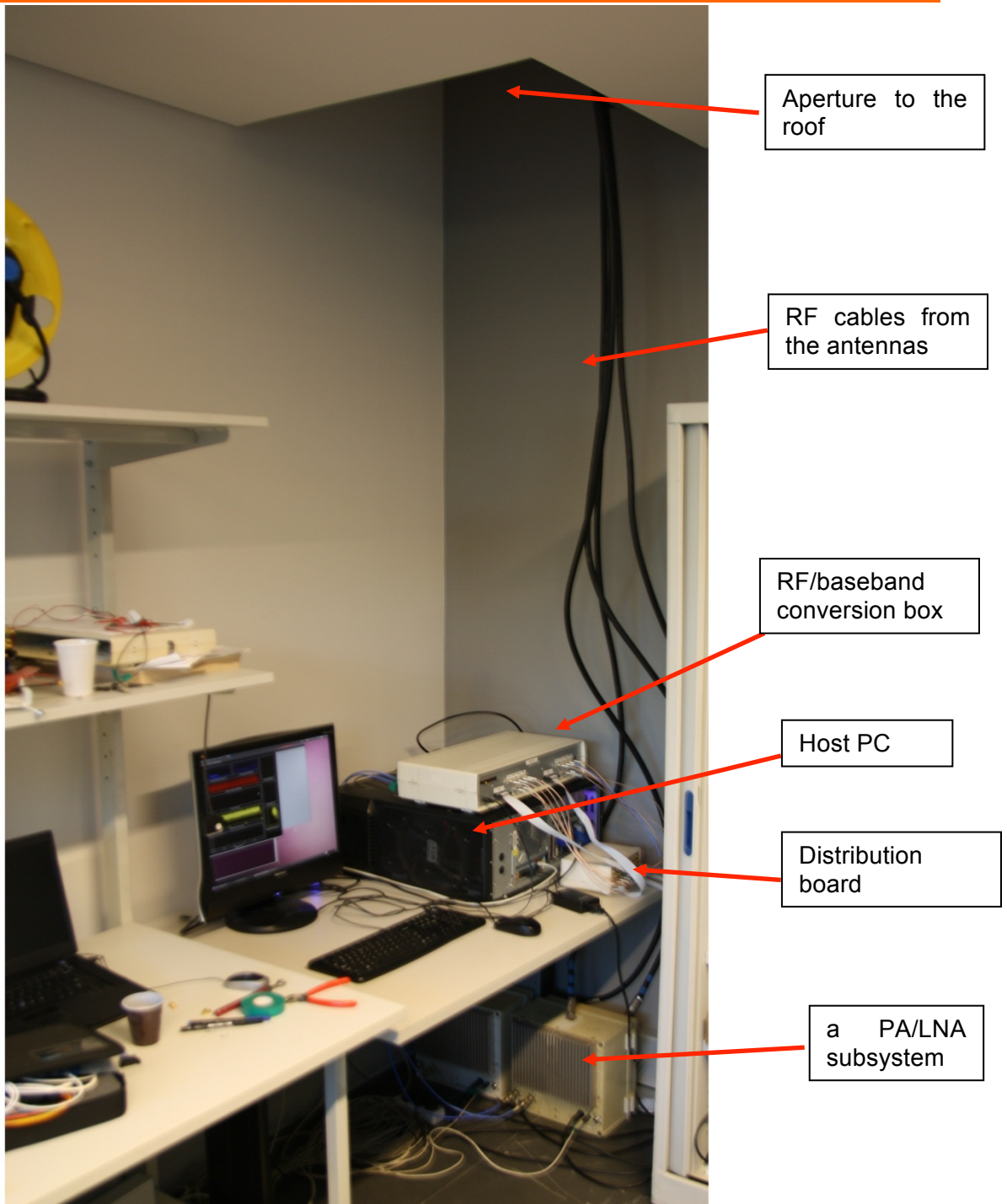


Figure 3-3: Overall hardware in the control room

3.1.2 Experimental vehicles

Eurecom has two experimental vehicles: a BMW X5 and a Renault Kangoo (see Figure 3-4). Both vehicles have DC/AC converters that allow to power all OpenAirInterface equipment directly from the car. The BMW X5 additionally has built-in antennas for 802.11p and DAB.



Figure 3-4: Experimental vehicles: BMW X5 and Renault Kangoo

3.1.3 Lab equipment

Eurecom currently has the following equipment available in their labs

- Agilent MXA N9020A Signal analyzer
- Agilent PSG E8257D Signal generator
- Lecroy Waverunner 6050 scope
- Rhode&Schartz SMT03 signal generator
- Rhode&Schwarz ZVB14 Network analyzer
- Rhode&Schwarz FSH6 Spectrum analyzer
- Rhode&Schwarz SMBV signal generator
- Rhode&Schwarz SMBV vector signal generator
- Tektronix Scope

3.1.4 Software radio platforms

3.1.4.1 ExpressMIMO1

The OpenAirInterface ExpressMIMO platform was developed jointly by Eurecom and Télécom ParisTech. Its hardware potentially supports a wide range of different standards like GSM, UMTS, 802.11, DAB, LTE as well as their multimodal processing and Time / Frequency Division Duplex (TDD / FDD) modes. It should be noted that to-date the only considered air interfaces for which a subset of the air-interface procedures have been implemented are LTE, 802.11p and DAB. The platform is used primarily for architecture exploration in SDR studies.

The platform is capable to process up to eight different channels simultaneously (four in reception, four in transmission) by reusing the same HW resources. As each channel may support a different wireless communication standard, the main design challenge is the synchronization of these resources by providing a maximum accuracy and by meeting all the real-time requirements. ExpressMIMO is used for experimental purposes only. Therefore the chosen target technology are FPGAs which come with a reduced design time, higher runtime flexibility, simple ease of use and lower costs for small quantities when compared to other solutions. Nevertheless ASICs are considered in a future version once the whole baseband design has been validated. In contrast to the previously presented solutions, the current design of the ExpressMIMO platform is split over two different FPGAs from Xilinx: (1) a Virtex 5 LX330 for the baseband processing and (2) a Virtex 5 LX110T for interfacing and control. To simplify testing on the platform, the two FPGAs can run stand-alone if required. Another difference is that the baseband processing which is responsible for the signal processing of the

transceiver is split over different DSP engines. The underlying hardware architecture further allows to process four receive and four transmit channels in parallel by using the same resources.

The interface and control FPGA transfers the signal coming from / going to the MAC layer and contains the main CPU (SPARC LEON3 processor) being responsible for the main control flow of the system. The two FPGAs are connected via an AMBA / AVCI DSP bridge while the different DSPs on the baseband side are connected via an AVCI crossbar. As only seven DSPs plus the VCI RAM and the main CPU are connected with each other, the performance of this crossbar is sufficient for the design of the ExpressMIMO platform.

The available memory space is distributed in a non-uniform way. Each DSP engine has its own memory space that is also mapped onto a global memory map. This global map is provided to the main CPU and to the DSPs and is consulted in case of DMA transfers between the DSPs or between the two FPGAs. For internal processing, the DSPs apply a local addressing scheme. In addition, an external DDR memory is available for mass storage on the baseband side and a DDR2 memory (size 16 MByte) contains the LEON3 program code and can be used for mass storage on the control side.

For more details see ACROPOLIS deliverable D5.2

3.1.4.2 ExpressMIMO2

The newest platform which can target OAI software was given the code name ExpressMIMO2 since it follows its predecessor ExpressMIMO. The key target for ExpressMIMO2 was to reduce its cost so as to make it accessible for other laboratories wanting to experiment with OAI using a reasonable number of nodes (i.e. normally greater than 2). The board was designed to allow for stand-alone operation at low-power levels (maximum 0 dBm transmit power per channel) simply by connecting an antenna to the board. The chosen RF technology covers a very large part of the available RF spectrum (250 MHz-3.8 GHz) with channels up to 20 MHz bandwidth.

ExpressMIMO2 is designed to be used with off-the-shelf PCs running standard Linux distributions and potentially a real-time extension (RTAI, Xenomai or PREEMPT-RT) in order to make use of open-source development tools and open-source networking tool suites and applications, both stable and experimental. EURECOM has successfully tested the board with regular laptops through a readily-available (and cheap) conversion cable in order to connect through the ExpressCard slot which also uses PCIe technology. Today OAI supports RTAI but the other options may become available for ExpressMIMO2, as a function of partners' and projects' interest.

The PCIe bus interface allows existing drivers from its predecessors (CBMIMO1 and ExpressMIMO) to be used as is. Similarly the OAI OCTAVE interfaces for non-real time experiments can also be used with changes only related to configuration of the new RF components. The cost of the board is approximately 1500 euros, depending on quantities ordered with the company fabricating the board, and could actually be less in time. EURECOM will provide information to partners in the Network of Excellence wishing to acquire such equipment.

ExpressMIMO2 motherboard characteristics

The board is shown in Figure 3-5 and is built around a low-cost Spartan-6 FPGA (150LXT) with native PCIe on the FPGA fabric and coupled with 4 high-performance LTE RF ASICs on-board, manufactured by Lime Micro Systems (LMS6002D). The combination allows for four 20 MHz full-duplex or half-duplex radios to be interfaced with a desktop or lap-

top PC without the need for external RF. External RF is, however, required if high power output, antenna duplexing or standard-compliant channel filtering are required. This board is described in the subsequent section.

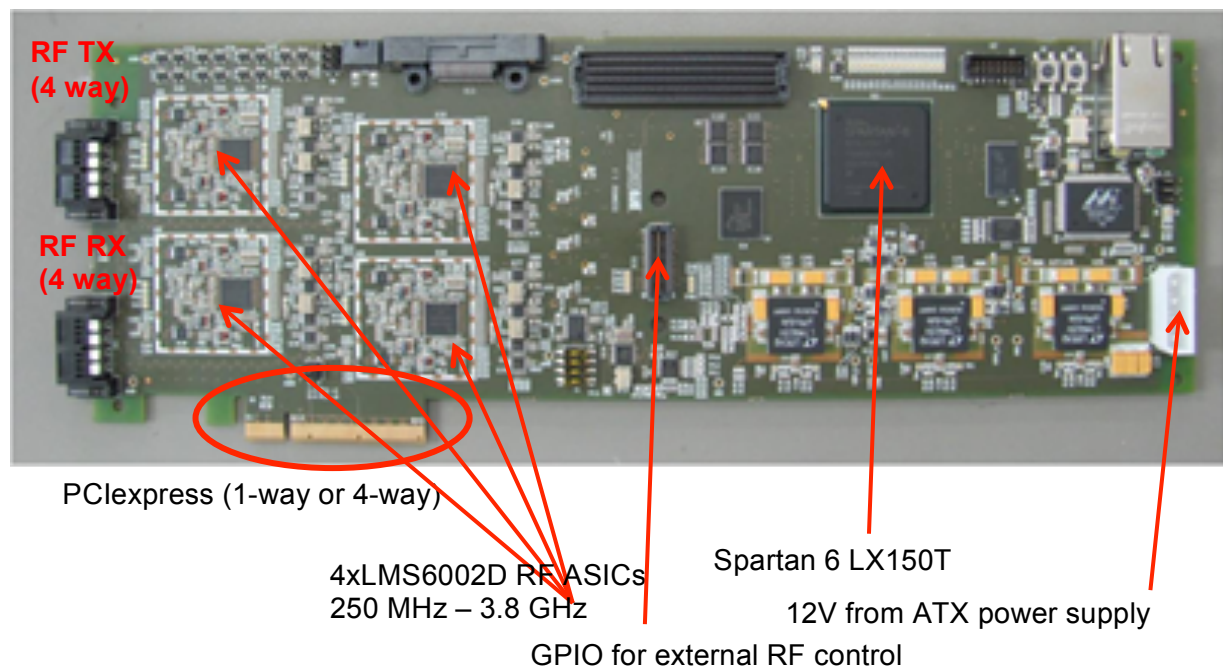


Figure 3-5: ExpressMIMO2 Motherboard

The embedded system on the ExpressMIMO2 FPGA is shown in Figure 3-6. Similarly to ExpressMIMO and CBMIMO1 it is based on a LEON3 microcontroller. In the current design the LEON3 and the on-chip bus are clocked at 61.44 MHz which is sufficient for the throughput of the 1-way PCIe bus (62.5 MHz 32-bit). The embedded system is augmented by a data-acquisition and framing unit which interfaces with the 4 LMS6002D and controls both the sample input-output and the serial programming busses (SPI) for RF and sampling configuration parameters. The LEON3 has a large DDR3 memory for data and program storage. An on-board Ethernet PHY is also provided, although it is currently not used.

The embedded software for the FPGA is booted via the PC or can reside entirely in the boot ROM which is part of the FPGA design. The current software, however, is booted by PClexpress dynamically under control of the PC device driver. A typical application, therefore, is a combination of PC software dialoguing with the card via driver configuration of shared PCI memory space and a program in the local memory on the embedded system. This program can be dynamically loaded from the PC.

The basic design does not include any on-FPGA signal processing and consumes approximately 10-15% of the FPGA resources. There is significant room left for additional processing on the FPGA, for instance Xilinx FFT processors or turbo-decoders to offload some processing from the host PC if required.

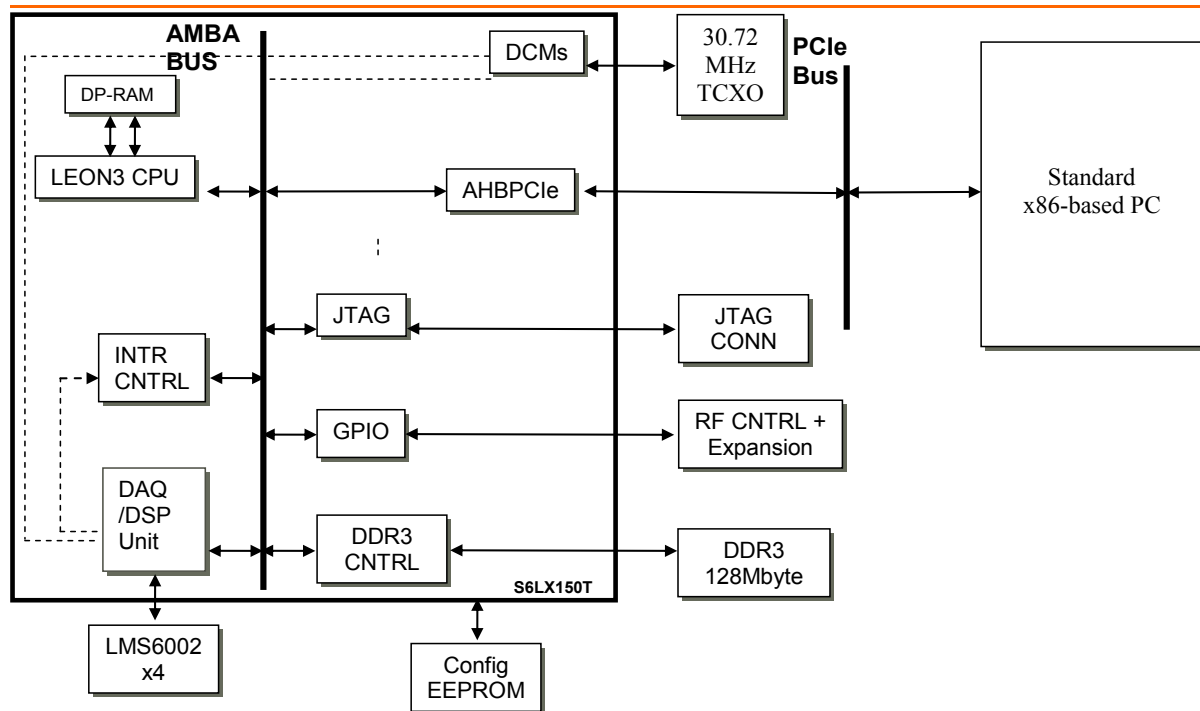


Figure 3-6: ExpressMIMO2 Embedded System

Examples of ExpressMIMO2 RF performance

In order to test the RF capabilities of the LMS6002D RF ASICS initial testing was performed using laboratory testing equipment attached to ExpressMIMO2. An example of the receiver performance on a 5 MHz LTE waveform driven by a Rohde&Schwarz SMBV100 is shown in Figure 3-7. Similarly the RX spectrum (with spectral nulls clearly visible) of an LTE PDCCH channel with 35 dB SNR is shown in Figure 3-8. Both clearly show the excellent dynamic range of the receiver.

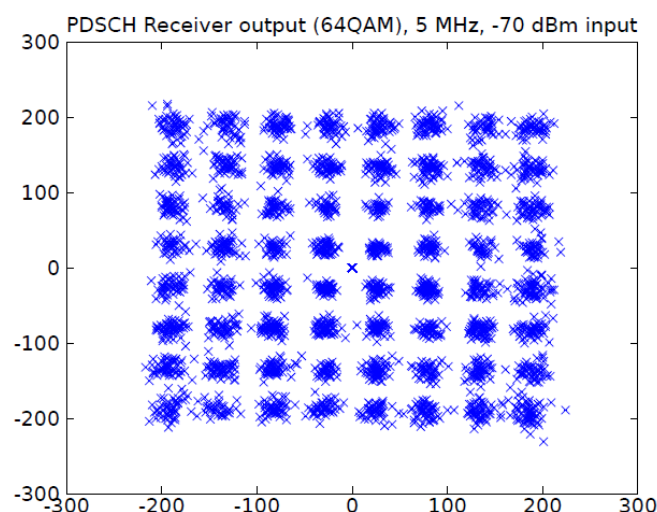


Figure 3-7: RX Constellation, LTE 5MHz 35dB SNR

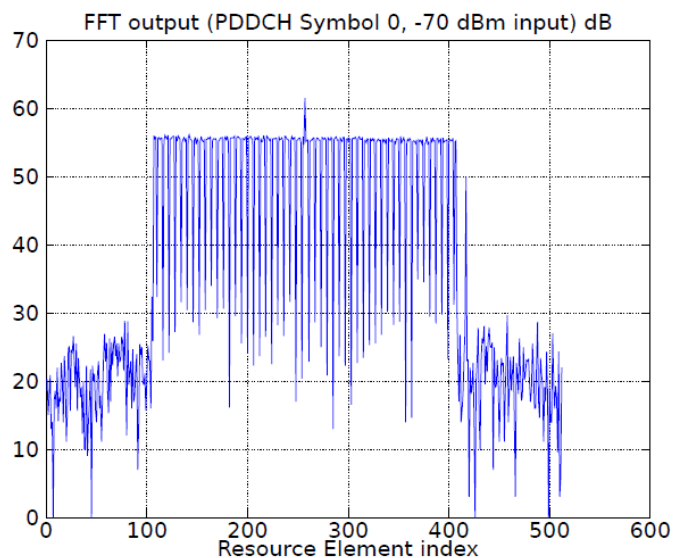


Figure 3-8: RX Spectrum (LTE 5 MHz, 35 dB SNR)

Two examples of the TX performance of ExpressMIMO2 are shown in Figure 3-9 and Figure 3-10. The transmit waveform is clearly spectrally pure at 750 MHz (TV white-space) and can be pushed to approximately 0dBm output with acceptable linearity. Similarly, the received constellation on a Rohde&Schwarz FSQ clearly shows the high dynamic range and linearity of the transmitter.

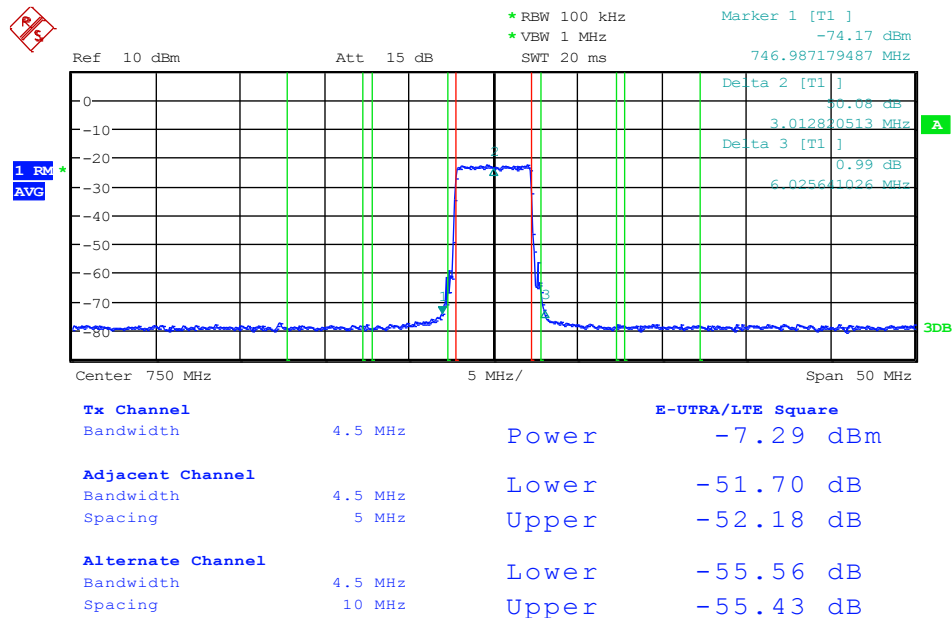


Figure 3-9: ACLR at the TX output of ExpressMIMO2, at 750 MHz

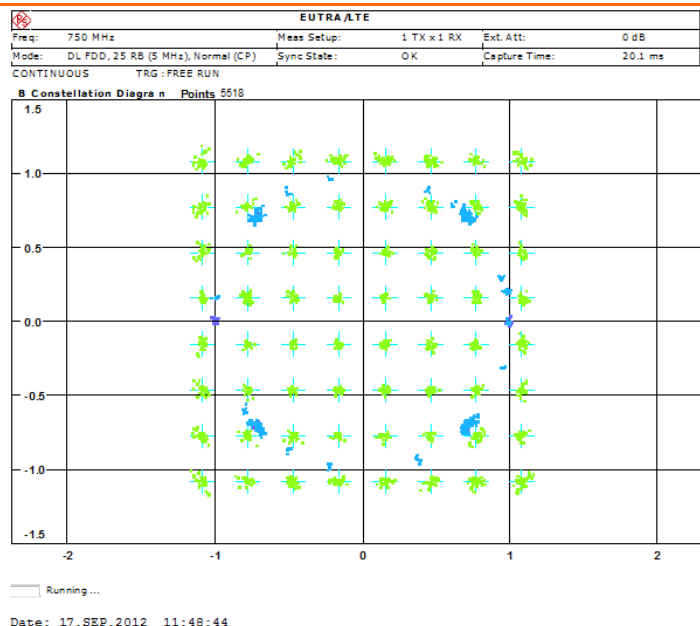


Figure 3-10: Constellation at the TX output of ExpressMIMO2, at 750 MHz

Multi-band RF front-end characteristics

EURECOM has also sub-contracted the design of a higher power RF front-end (21 dBm) per channel in all common bands from 250MHz-8 GHz. Additional upconverter (TX) and downconverter (RX) stages are added to allow for higher frequency operation in TDD mode (above 4 GHz). A picture of the RF front-end module is shown in Figure 3-11. The board is configured for operation as a UE in most cellular FDD bands, but can also be configured for eNB as well in this bands with small component modifications. The current spectral configuration are shown in Table 1. Note that they are useful for TVWS operation.

	Band	Duplex
1	TVWS (400-790MHz)	TDD
2	DD (790- 862 MHz)	TDD
3	DD (790- 862 MHz)	FDD
4	1900 Mhz	TDD
5	2,4 Ghz	TDD
6	2,6 GHz	FDD
7	2,6 GHz	TDD
8	3,5 Ghz	TDD
9	5-6 GHz	TDD
10	3,5 GHZ	FDD

Table 1: Frequency Bands and Duplexing (below 4GHz)

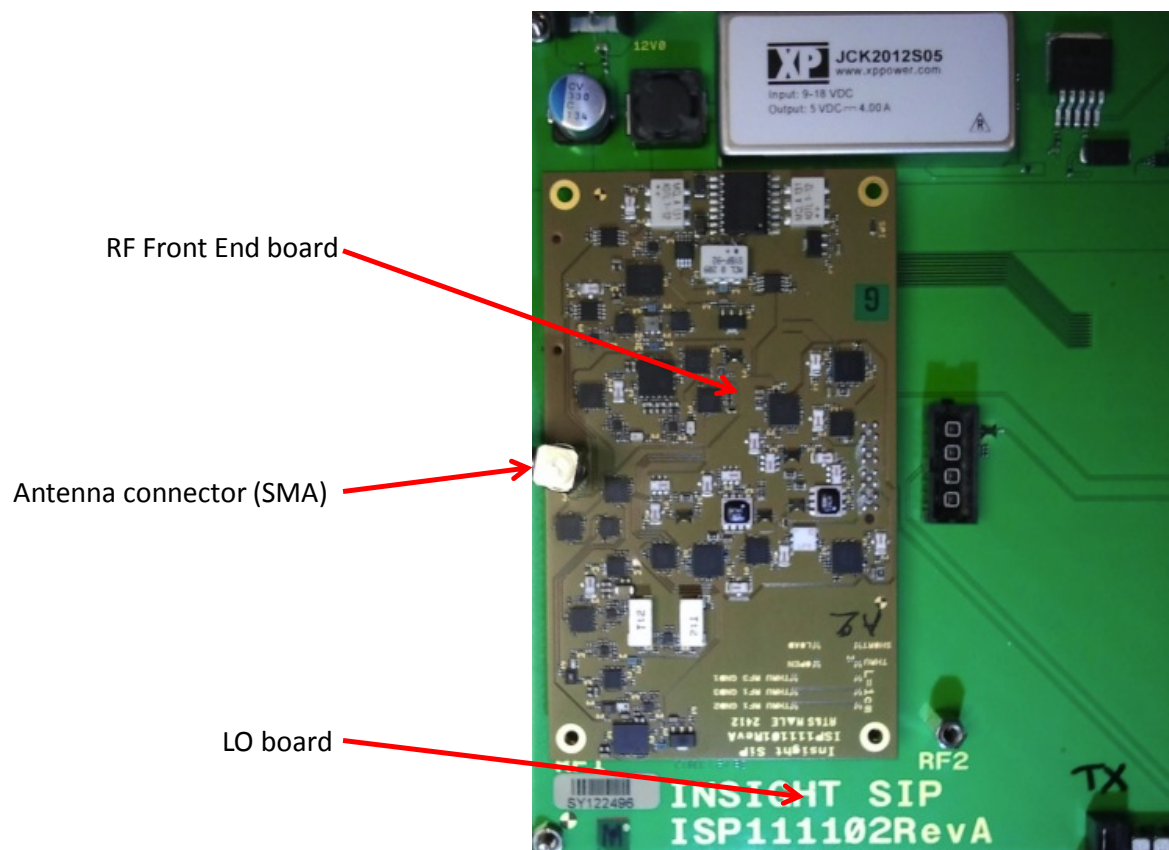


Figure 3-11: External RF/Antenna modules for ExpressMIMO2 (one of 4)

The RF front end boards comprise three parts:

- a RF conversion circuit which aims at transpose the signal in the appropriate frequency band. This circuit includes contains filters, amplifiers and frequency mixers.
- a filter bank part used for band selection and duplexing
- high-power (21 dBm) and low-noise amplifiers

The following figures illustrate the high-power output spectrum at 750 MHz and 2.6 GHz. Note that these plots are below the specifications of the components and thus require some passive component value adjustments to achieve the maximum output power

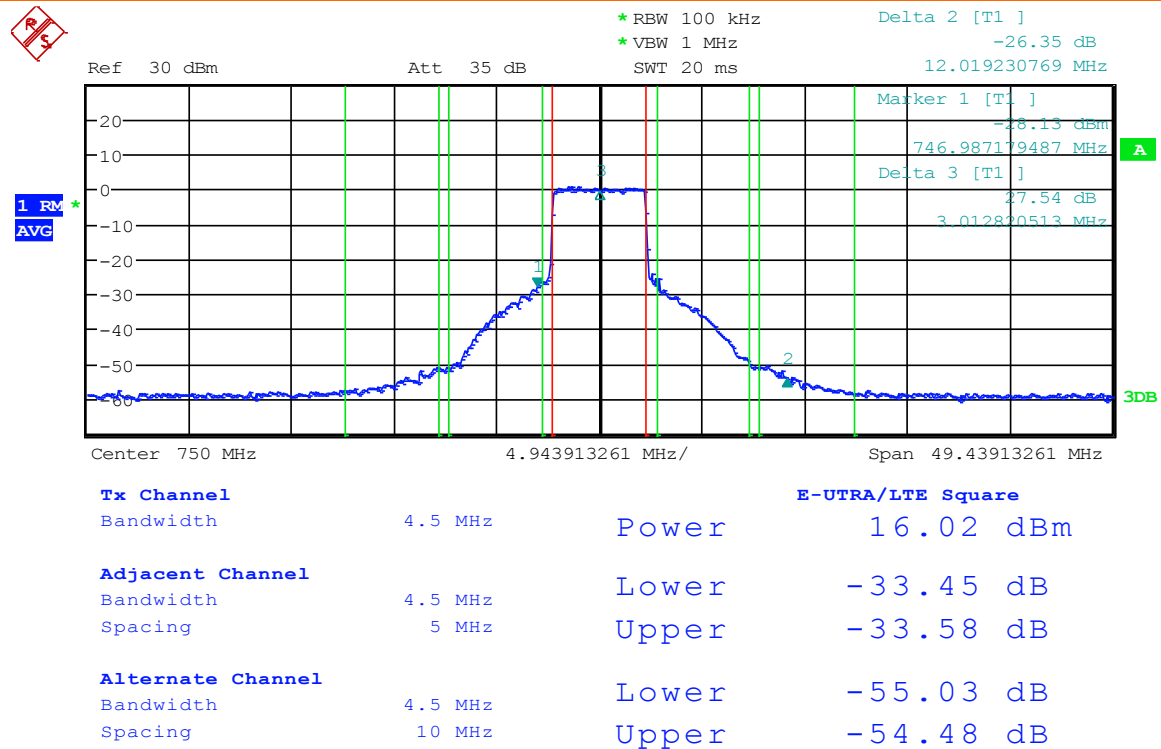


Figure 3-12 : ACLR of RF FE @ 750 MHz, for 16 dBm

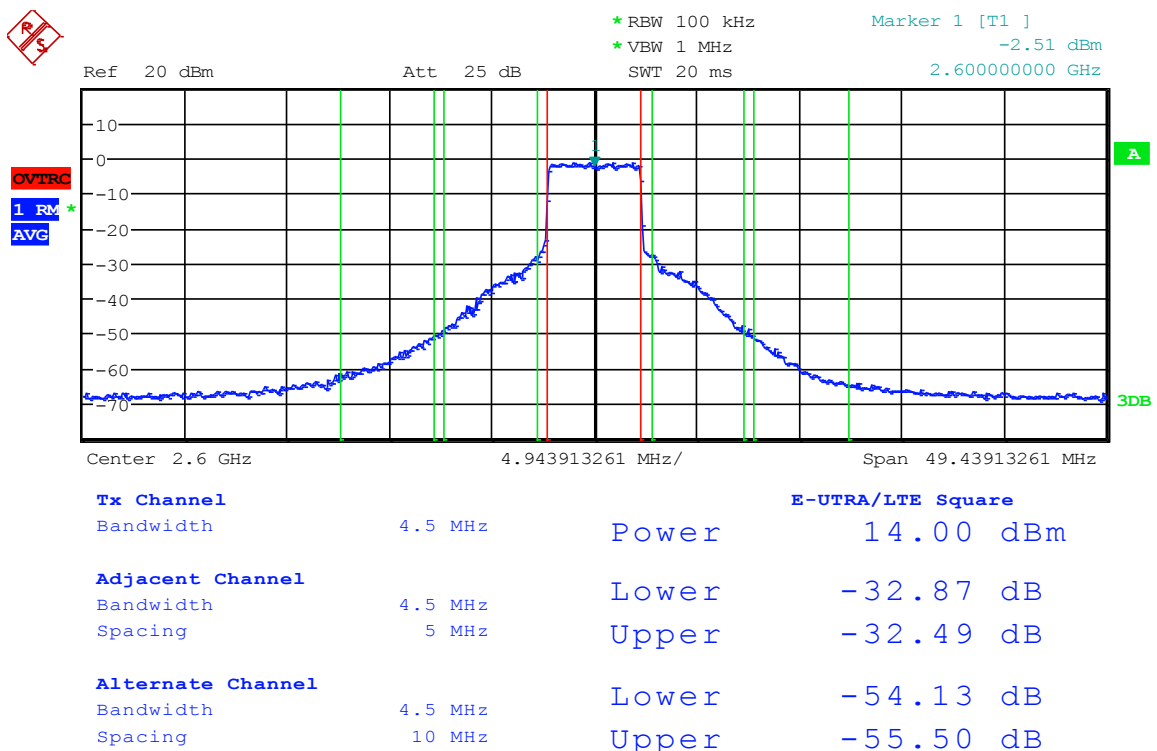


Figure 3-13: ACLR of RF FE @ 2600 MHz, for 14 dBm

3.1.5 OpenAirInterface LTE software modem

The OAI initiative develops open-source MODEM and protocol stack implementation for the ExpressMIMO baseband engine and x86 PC targets. These implementations currently target LTE and 802.11p air interfaces. The LTE implementation, OpenAir4G, provides a standard-compliant LTE Rel-8 implementation of PHY and MAC for a subset of the specifications (36.211, 36.212, 36.213, 36.321, 36.322, 36.323, 36.331). The gnu-C implementation (with x86 SIMD hardware acceleration) can be made to run under any GNU environment, although x86 Linux and RTAI-based targets have only been considered.

An overview of the currently supported physical/transport channels and transmission modes is given in the following tables. Compliance of the implementation is being validated in conjunction with various industrial partners and is summarized here. Basic compliance at the PHY is determined using standard LTE test and measurement equipment from Rhode-Schwarz and partners industry-grade equipment.

Physical Channel	Functionality	LTE Compliance
PSS	TX/RX	Validated for 1 antenna port at eNB (implemented for 1,2)
SSS	TX/RX	Validated for 1 antenna port at eNB (implemented for 1,2)
Cell-specific Reference signals	TX/RX, Modes 1,2,3,4,5,6 1-2 antenna ports at eNB	Validated for 1 antenna port at eNB (implemented for 1,2)
PBCH	TX/RX 1,2 antenna ports at eNB	Validated for 1 antenna port at eNB (implemented for 1,2)
PCFICH/PDCCH	TX/RX 1,2 antenna ports at eNB All 5 MHz DCI Formats	Validated for 1 antenna port at eNB, DCI Format 1,1A (TDD/FDD), (implemented for 1,2)
PHICH	TX/RX 1,2 antenna ports at eNB	Validated for 1 antenna port at eNB (implemented for 1,2)
PDSCH	TX/RX 1,2 antenna ports at eNB Mode 1,2,5,6	Validated for 1 antenna port at eNB (implemented for 1,2)
PUSCH + UCI	TX/RX 1,2 antennas ports at eNB	Validated
PUCCH	TX/RX formats 1,1a,1b	Validated
DRS	TX/RX, 1-2 antenna ports at eNB	Validated
SRS	TX/RX, 1-2 antenna ports at eNB	Not validated yet, implemented
PRACH	TX/RX, 1-2 antenna ports at eNB	Validated (formats 1-3)

Table 2: Physical Channel Support in OpenAirInterface.org (3GPP 36-211)

Coding Methods	Functionality	LTE Compliance
Tail-biting C. code, ,	TX/RX	validated
Turbo code	TX/RX	validated
rate-matching (C. code)	TX/RX	validated
Rate-matching (turbo)	TX/RX	validated
segmentation	TX/RX	validated
CRC 24-bit	TX/RX	validated
CRC 16-bit	TX/RX	validated
CRC 8-bit	TX/RX	validated

BCH	TX/RX	Validated
DCI	TX/RX, 5 MHz TDD/FDD formats	Validated (format 1,1A,1D,1B)
DLSCH	TX/RX	Validated
ULSCH/UCI	TX/RX	Validated (subset of UCI formats)
CQI	TX/RX	Validated
CFI	TX/RX	Validated
HI	TX/RX	Validated

Table 3: Coding and Multiplexing (36.212)

Procedure	Functionality	LTE Compliance
Random-Access	TX/RX, full procedure, Connection Establishment, handover, data transfer	Validated
Random-access response	TX/RX, full procedure	Validated
PDCCH procedures	TX/RX	Validated
DL/UL HARQ procedures	TX/RX, TDD, no PHICH	Not validated
CQI/PMI/RI reporting	TX/RX, HLC and Sub-band PMI on PUSCH	Not validated
PUCCH	Implemented (formats 0,1a,1b)	

Table 4: Physical Layer Procedures (3GPP 36-213)

OpenAir4G Protocol Stack

OpenAir4G provides a full real-time protocol stack for a gnu gcc environment implementing a subset of LTE Rel. 8/9 of access stratum as shown in Figure 3-14 and includes the following blocks:

- Linux Network device driver (kernel)
- MAC/RLC/PDCP/RRC and IP
- PHY procedures
- Can be integrated with openair4G MODEM or abstraction of physical channels, MODEM is abstracted along with propagation
- Can be vectorized for multiple instances (multi eNB, multi UE, combined eNB/UE, multiple component carriers for carrier aggregation)

NAS

The direct inter-connection between LTE and IPv6 is performed using an inter-working function, located in the NAS driver and operating in both the Control Plane and the Data Plane. This function provides the middleware for interfacing IPv6-based mechanisms for signalling and user traffic with 3GPP-specific mechanisms for the access network (e.g. for mobility, call admission, etc.). It is developed as an extension of a standard IPv6/IPv4 network device driver. It implements the EPS bear association with the one RB, which is associated with the one PDCP entity.

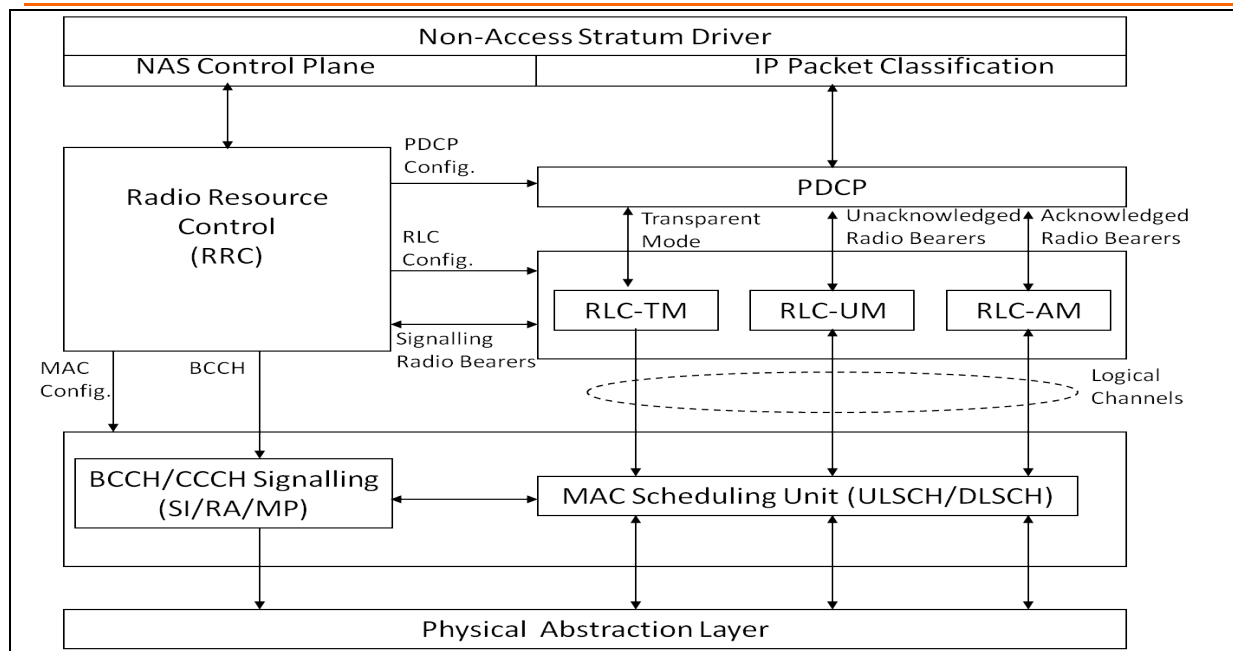


Figure 3-14 OpenAir4G protocol stack

RRC

The RRC layer, shared between the UE and the ENB, performs the control of the radio interface. It is based on 3GPP 36.331 v9.2.0. The control procedures available in the LTE platform are the following:

- System Information broadcast
- RRC connection establishment
- Measurement configuration and reporting
- the signalling data transfer
- Connection reconfiguration (addition and removal of radio bearers, connection release)
- the measurement collection and reporting at UE and eNB
- EUTRA handover is under integration

These procedures are being extended to support MBMS for multicast and broadcast.

MAC

The MAC layer implements a subset of the 3GPP 36-321 release v8.6 in support of BCH, DLSCH, RACH, and ULSCCH channels. The eNB MAC implementation includes:

- RRC interface for CCCH, DCCH, and DTCH
- Schedulers
- DCI generation
- HARQ Support
- RA procedures and RNTI management
- RLC interface (AM, UM)

UE MAC implementation includes

- PDU formats: all control elements and logical channels
- RLC interface AM,UM, TM
- RRC transparent interface for CCCH and BCCH
- Buffer status reporting and scheduling request procedures
- Power headroom reporting

PDCP

The current PDCP is header compliant with 3GPP 36-323 Rel 10.1.0 and implement the following functions:

- User and control data transfer
- Sequence number management
- RB association with PDCP entity
- PDCP entity association with one or two RLC entities

RLC

The RLC layer implements a full specification of the 3GPP 36-322 release v9.3 for all the three mode: transparent mode (TM), unacknowledged mode (UM), and acknowledge mode (AM) with the following characteristics:

- RLC TM (mainly used for BCCH and CCCH)
 - Neither segment nor concatenate RLC SDUs
 - Do not include a RLC header in the RLC PDU
 - Delivery of received RLC PDUs to upper layers
- RLC UM (mainly used for DTCH)
 - Segment or concatenate RLC SDUs according to the TB size selected by MAC
 - Include a RLC header in the RLC PDU
 - Duplication detection
 - PDU reordering and reassembly
- RLC AM, compatible with 9.3
 - Segmentation, concatenation, and reassembly
 - Padding
 - Data transfer to the user
 - RLC PDU retransmission in support of error control and correction
 - Generation of data/control PDUs

3.1.6 OpenAirInterface 802.11p modem

A second air interface implementation running completely on the ExpressMIMO baseband engine is also available with OAI and supports the physical layer of 802.11p and DAB for vehicular communications. This implementation was developed in the context of the PLATA project jointly with the Technical University of Munich and BMW Research and Innovation Center as a proof-of-concept design and is currently being tested in simulation and in a realistic RF environment. Both receivers have been implemented in C by using the Library for Express MIMO baseband (libembbb). This library, which is developed by the System on Chip Laboratory of Telecom ParisTech, allows emulation of the baseband processing part of the platform on a PC as well as an easy verification of the whole design on the platform itself. Hence repeatable and scalable real-time experiments are possible without the need of RF equipment. The next step is the combination of the two receivers on the same hardware-software architecture. This task is quite challenging since the DAB receiver and 802.11p receivers have radically different processing latency requirements.

3.1.6.1 Intel x86-based OpenAirITS implementation

For the purpose of networking experiments using the ExpressMIMO2 platform, an x86-based softmodem was also implemented using x86 SIMD intrinsics and has been integrated into the PCIe-based real-time framework described in Section 0. A light version of the 802.11 MAC layer supporting broadcast transmission (i.e. lack of ACK/NAK, RTS/CTS protocols for

the moment) has been successfully integrated with the mac80211 development found in recent Linux distributions as shown in .

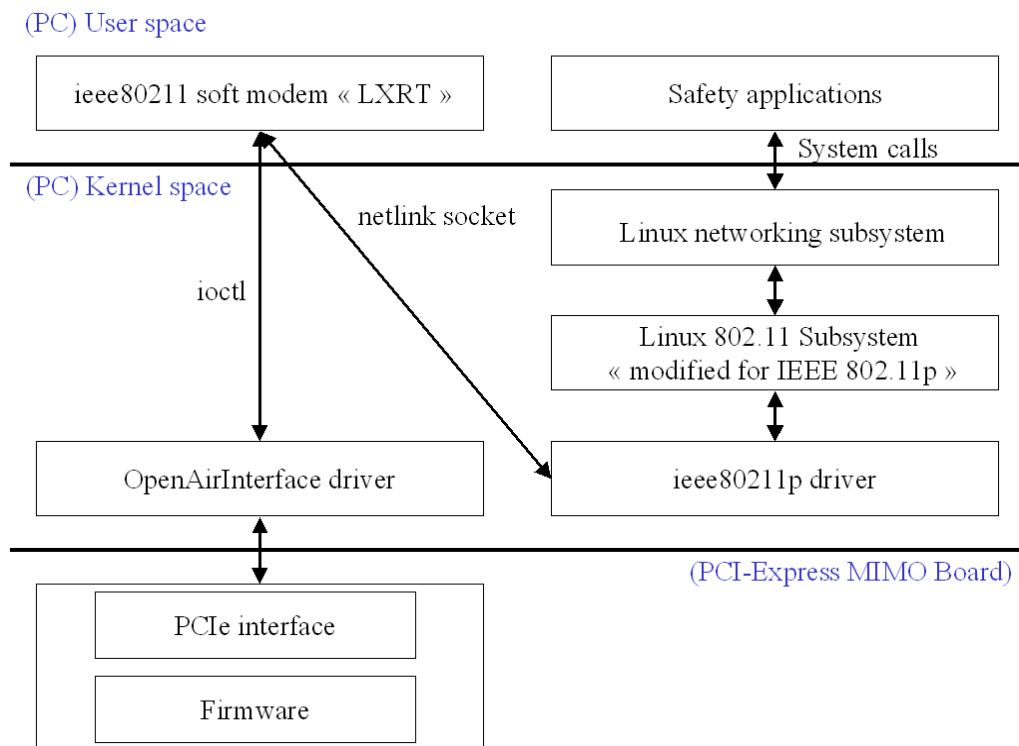


Figure 3-15: Integration of OpenAirITS with mac80211

3.1.7 Collaborative tools

3.1.7.1 SVN repository

OpenAirInterface software and hardware (that is FPGA related) development is maintained with and SVN server hosted by EURECOM. The server can be accessed read-only at <http://svn.eurecom.fr/openair4G/trunk>. To contribute back the URL <http://svn.eurecom.fr/openairwebsvn/openair4G/trunk> should be used. For this users need approval from one of the administrators. We may restrict access to read only until we're confident that the people are responsible and are capable of contributing. Typically, the people outside EURECOM requiring the ability to "commit" changes to OpenAirInterface are project partners in collaborative public initiatives (ANR, ICT, etc.). For such people, this is not a problem and you will receive instructions on how to contribute to the repository.

3.1.7.2 Twiki

A TWiki server is also available for users of OpenAirInterface at <https://twiki.eurecom.fr/twiki/bin/view/OpenAirInterface/WebHome>. It contains practical information regarding software installation, machine configuration, code compilation, etc. People using OpenAirInterface are invited to contribute to the writing of this site, it'll make using OpenAirInterface easier for everyone. Also, we'd like to have people contribute by writing short documents regarding some of the more interesting features and mechanisms in OpenAirInterface, such as PHY/MAC algorithms, routing protocols, etc.

The TWiki server can be accessed by everyone in read-only mode. To get write access to it, send an email to openair_tech@eurecom.fr.

3.1.7.3 Forum

A [Bulletin Board forum](#) has been created to exchange information between developers and people curious about what going on here. This is the preferred means for direct information exchange.

3.1.7.4 Bugzilla

A [Bugzilla](#) has been created to report bugs.

3.1.7.5 Mailing list

This email list is for active developers of the openair4G repository of the OpenAirInterface project. It should be used to announce new features and bugfixes and discuss new developments on all aspects of openair4G, i.e., openair1,2,3 as well as emulation and hardware. So please state clearly in the subject what the discussion is about.

- To subscribe to the list by sending an email to majordomo@eurecom.fr with the command (either in subject or body)
subscribe openair4g-devel
- To unsubscribe by sending an email to majordomo@eurecom.fr with the command (either in subject or body)
unsubscribe openair4g-devel
- More commands are available by sending an email to majordomo@eurecom.fr with the command (either in subject or body)
help

This email list is an alternative to our forum, which unfortunately is not used.

3.2 Interfaces

As described earlier, 10 ExpressMIMO2 hardware targets are available to be used in situ at Eurecom for Newcom# partners. These cards can be used for offline signal transmission and acquisition using a Matlab/Octave interface or can be used with the OpenAirInterface LTE protocol stack in real time. Both methods are described in more detail in the appendix.

Additionally, there are several ways how to access OpenAirInterface remotely. The first option is to download OpenAirInterface to your local machine either by downloading the source-code directly from our SVN server described above or by downloading a ready-installed virtual machine (see below for details). This will allow you to run both link level simulations and system level simulations locally on your machine. We will briefly describe these interfaces in Sections 3.2.1, 3.2.2, and 3.2.3 respectively.

Some features, like the OpenAirInterface system level simulator can also be accessed remotely. This includes access to Eurecom's high performance computing clusters, which is especially useful for large scale simulations. The two existing portals for remote access are described in Sections 3.2.4.1 and 3.2.4.2 respectively.

3.2.1 OpenAirInterface Matlab interface

There are link-level simulation testbenches available in Matlab. These testbenches provide a way for fast prototyping of new functions in Matlab and to compare their performance to the current OAI implementation. The most important C functions of the OAI library are available in Matlab through the mex-interface allowing for fast and coherent simulations. The functions include turbo encoding/decoding as well as the inner receiver functions. The OFDM modula-

tion and demodulation is not implemented to keep simulations fast and the testbenches simple. After rapid prototyping and evaluation of new functions in Matlab, the subsequent step is to implement the new functionality in C and test the implementation with the unitary simulators.

All testbenches and functions are available under the path `/openair1/USERSPACE_TOOLS/MATLAB`. To use the simulator requires the following steps:

1. Working copy: Ensure your OAI installation is compiling and running on your system, for instance by compiling `/openair1/SIMULATION/LTE_PHY/dlsim.c` via "make dlsim" and run it as `./dlsim -x1 -a -s-3 -n100`
2. Compiling Mex-files: In path `/openair1/USERSPACE_TOOLS/MATLAB` compile the libraries with "make libs" and subsequently compile the mex files with "make mex".
3. Use the simulator For the moment there is only a simulator for transmission mode 5 available located in `/openair1/USERSPACE_TOOLS/MATLAB/SIMULATION/LTE_PHY/TM5`.

3.2.2 OpenAirInterface unitary simulators

These testbenches include simulation of transport/physical channels (PBCH, DCI/PDCCH, DL SCH/PDSCH, UL SCH/PUSCH). Further unitary simulation of the coding subsystem components also exists. These simulators firstly constitute the starting point for testing any new code/innovations in the basic PHY functionality and for performance analysis. Secondly, these are the required tests to be performed before committing any new code to the repository, in the sense that all of these should compile and execute correctly.

The simulators are located in `/openair1/SIMULATION/LTE_PHY/`. The basic physical channel simulators are listed in the following table.

<code>dlsim.c</code>	Simulates PDSCH. Currently supported are transmission modes 1,2,5 and 6.
<code>pbchsim.c</code>	Simulates the physical broadcast channel.
<code>pdcchsim.c</code>	Simulates control channel, i.e., decoding of common and/or specific DCIs
<code>ulsim.c</code>	Simulates the uplink channel PUSCH.
<code>mbmssim.c</code>	Simulates the Multimedia Broadcast/Multicast Service channel
<code>prachsim.c</code>	Simulates the random access channel
<code>pucchsim.c</code>	Simulates the uplink control channels

All simulators are compiled by executing "make dlsim", "make pbchsim" etc. To launch the simulation simply execute `./dlsim`, `./ulsim` etc. Possible parameters can be added like `./dlsim -x1 -s20 -n100` which sets the transmission mode to 1, the SNR to 20dB and runs the simulation for 100 frames. The list of all possible parameters is accessible via `./dlsim -h`, `./ulsim -h` etc.

3.2.3 OpenAirInterface system level simulator

The system emulator allows for simulation and emulation of an OpenAirLTE network. It provides either simulation with full PHY and synthetic radio channels, or with PHY abstraction. In both PHY modes, the full protocol stack is executed as is the case on OpenAirInterface hardware. Several eNB and UE are virtualized in the same process. Support for Ethernet-based emulation is also provided so that nodes can be distributed on a PC network.

The simulator is located in targets/SIMU/USER and manually compiled as
“make oaisim”

Refer to the on-line help to see a list of command-line arguments
“./oaisim -h”.

A more detailed description of the system level simulator and instructions how to conduct experiments with it are described in the Appendix. The system level simulator can also be accessed remotely, which is described next.

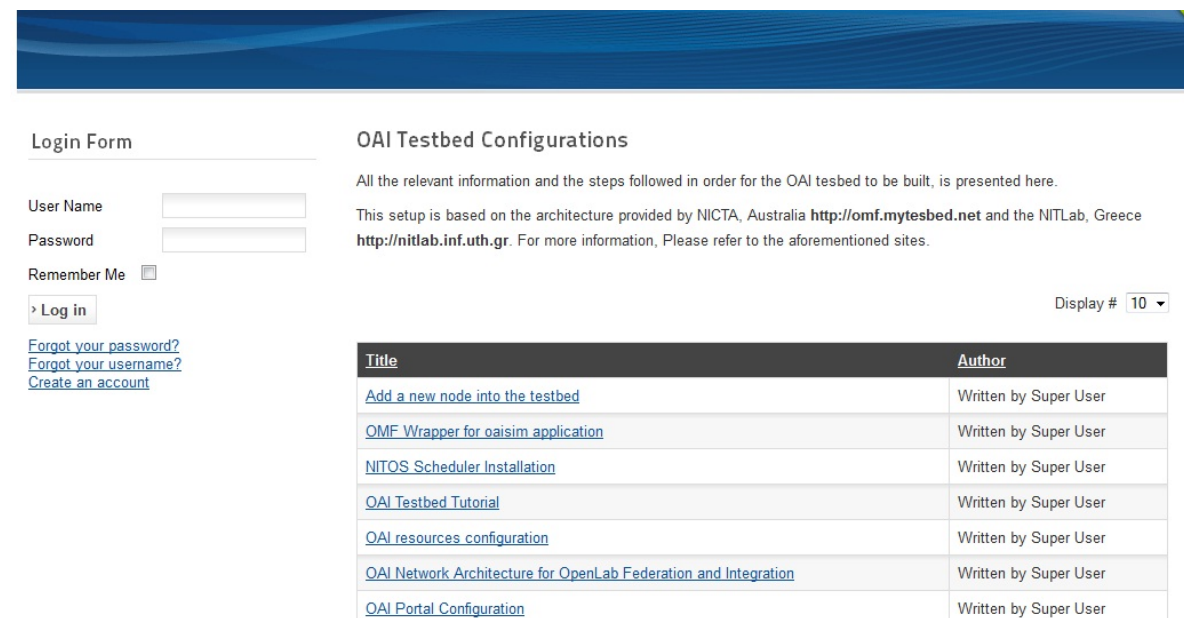
3.2.4 Remote access

There are three methods to use OAI easily:

1. The direct access to the simulation/emulation platform through the onelab interface (OMF methodology)
2. The Frontend access using the eurecom emulation web portal
3. The full integrated OAI virtual machine image which can be activated using VMware software on your local machine.

3.2.4.1 OAI remote access through Onelab

The first solution to use OIA is the OMF-based platform, which provides a remote accessibility through a web-portal hosted on the domain <http://conect.eurecom.fr/scheduler>. The web portal maintains information and tutorial about using and experimenting in the OAI testbed, as well as information relevant for administration purposes and system maintainance. Moreover, it is used as the interface by the users for accessing the OAI testbed through the enhanced NITOS scheduler. Some screenshots of the portal are given below.



The screenshot shows the OAI Testbed login portal. It features a blue header with a wavy pattern. Below the header, there is a 'Login Form' on the left and 'OAI Testbed Configurations' on the right. The login form includes fields for 'User Name' and 'Password', a 'Remember Me' checkbox, and a 'Log in' button. Below the login form are links for 'Forgot your password?', 'Forgot your username?', and 'Create an account'. The 'OAI Testbed Configurations' section contains a paragraph of text and a table with two columns: 'Title' and 'Author'. The table lists several documents, all written by 'Super User'.

Title	Author
Add a new node into the testbed	Written by Super User
OMF Wrapper for oaisim application	Written by Super User
NITOS Scheduler Installation	Written by Super User
OAI Testbed Tutorial	Written by Super User
OAI resources configuration	Written by Super User
OAI Network Architecture for OpenLab Federation and Integration	Written by Super User
OAI Portal Configuration	Written by Super User

Figure 3-16: OAI testbed login portal

User Menu

Your Profile

Make a Reservation

My Reservations

Login Form

Hi Apostolos Apostolaras,

Log out

Scheduler

Current Slices: apaposto.

Current Server Time: 2012-12-19 14:22:00

Click on a date to select the day you want to start the reservation.

<< December 2012 >>

Sun	Mon	Tue	Wen	Thu	Fri	Sat
						01
02	03	04	05	06	07	08
09	10	11	12	13	14	15
16	17	18	19	20	21	22
23	24	25	26	27	28	29
30	31					

Choose the time you want to start the reservation and for how long (max 4 hours).

EDO

Select Slice, Start Time and Duration

Slice: apaposto

Start Date: yyyy/mm/dd

Start Time: 00 hh 00 min

Duration: 0.5 hours

Check Available Nodes

Available Nodes and Channels between 2012-12-19 17:30:00 and 2012-12-19 19:30:00 :
Slice: apaposto

Resource Type A - OAI emulation	Resource Type B - OAI emulation	Resource Type C - OAI emulation
Node 1: ☐	Node 11: ☐	
Node 2: ☐	Node 12: ☐	
Node 3: ☐	Node 13: ☐	
Node 4: ☐	Node 14: ☐	Node 21: ☐
Node 5: ☐	Node 15: ☐	Node 22: ☐
Node 6: ☐	Node 16: ☐	Node 23: ☐
Node 7: ☐	Node 17: ☐	
Node 8: ☐	Node 18: ☐	
Node 9: ☐	Node 19: ☐	
Node 10: ☐	Node 20: ☐	

Reserve

Figure 3-17: OAI testbed scheduler

To conduct an Experimentation through OMF – OMF Experiment Description Language (OEDL) is used. OMF defines and uses a Domain Specific Language to describe an experiment. This language is named OEDL for “OMF Experiment Description Language” and it is written in ruby scripting language. A domain specific language (OEDL) allows the experimenter/(testbed user) to write an Experiment Description (ED) which details the resource requirements, their initial configuration, and a state machine describing the time/event-triggered actions required to execute the experiment.

Briefly, the user submits the ED to an Experiment Controller (EC), which is hosted on the Web Portal (In our case the NITOS and the OAI web portal). The EC coordinates the provisioning and configuration of the various requested resources. An ED script allows an experimenter to precisely describe all resources needed as well as their configuration settings at various abstraction levels. Once the experiment pre-requisites are met, the EC sends/directives to the Resource Controller(s) associated with each resource, such as \start application X with rate of Y kbps.

OEDL also allows an experimenter to define the orchestration of an experiment either along a timeline (e.g. increase the sending rate every 60 seconds) or in reaction to context changes, such as measurements (e.g. \increase sending rate until packet error is greater than X"), or environment changes, such as location (e.g. \increase beaconing interval within 200 m of an intersection") or power. Besides modifying the actual resources used by an experiment, OEDL scripts can also control the experiment environment itself. Specifically, an OMF Experiment Description (ED) is composed of 2 parts in the following order:

Security: Public

Page 40

- **Resource Requirements & Configuration:** this part enumerates the different resources that are required by the experiment, and describes the different configuration that need to be applied on them
- **Task Description:** this part is essentially a state-machine, which enumerates the different tasks to perform with the required resources in order to realize the experiment

Moreover, OEDL commands are grouped in the following categories:

- *Top-level commands:* (used for property definition and system debug messaging)
- *Topology-specific commands:* (used in the Resource Requirements & Configuration section of the ED. They allow the definition of a topology involving specific resources, and some potential related constraints.)
- *Group-specific commands:* (used in the Resource Requirements & Configuration section of the ED. They allow the definition of a given group of resources, the description of the specific resources that should be placed in that group, and optionally the configuration to apply to them)
- *Prototype-specific commands:* (used for defining an OMF 'prototype')
- *Application-specific commands:* (used for defining an OMF 'application')
- *Execution-specific commands:* (used in the Task Description section of the ED. They allow the definition of the different tasks to execute when the experiment reaches a specific state)
- *Resource Paths:* (A resource path allows the access and the value assignment of a specific configuration parameter of a resource)
- *Testbed-specific commands:* (These OEDL commands are only available for specific testbed deployments, i.e. they act on particular types of resources that are only available on some specific testbeds according to their setup and provision technology).

OEDL commands are used to write wrappers and application definition for the OAI executable binary and this is analyzed in the next subsection.

In order to enable conducting experimentation in Eurecom's testbed on OAI, OMF must be able to understand the language of OAI software. For that reason we built a wrapper for the OAI binary that is used to configure a particular experimentation setup by allowing specific options to be enabled such the number of eNBs, the number of UEs, the number of frames in the experiment execution etc. Moreover, by having OMF experiment descriptions we can specify the topology and the parameters of an experimentation setup regarding the description that is given as an input. Experiment descriptions in OAI language can be parsed by ruby functions inside OMF scripts thus giving the opportunity to call a script inside the other. In this way, parameter specification should be defined in order for the experiment description written in OEDL to be able to parse the appropriate definition used in OAI. Here we give a snapshot of the OAI testbed where an OMF wrapper for the **oaisim** binary is presented. The oaisim binary is the OAI executable produced by the compilation of the OAI software. For more details please refer to the website link (<http://conect.eurecom.fr/scheduler/index.php/oai-testbed-setup/80-omf-wrapper-for-oaisim-application>)

3.2.4.2 OAI Frontend

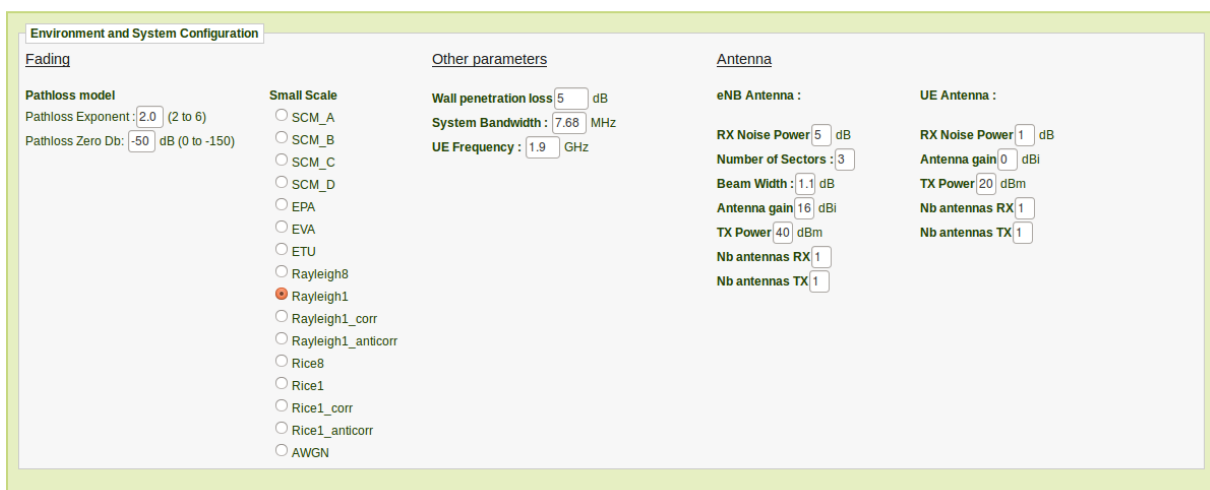
The second solution to access the uses directly the OAI emulator is the OpenAirWeb Server is based on the Content Management System DRUPAL 6. The emulation platform relies on Several PC clusters interconnected through the network file system (NFS) at EURECOM. PC Cluster 1 hosts the OpenAirWeb Server, whereas the other clusters are in charge of execution of the experiment. In order to provide load blanching between the PC clusters, the OpenPBS infrastructure will be deployed.

The OpenAirEmu web server is accessible from the web via the address <https://emu.openairinterface.org/>. In order to access to the OpenAir Scenario Descriptor (OSD),

the user should first successfully log in. If it's the first time that the user comes, he will be invited to create an account and will wait for EURECOM to validate it.

Once logged in, the user can access to the OSD and will fill it in with the emulation parameters he wants. The web server will summary the emulation details into a XML file that it will place on the NFS Share. OAISIM will detect this XML file, parse it and start the emulation according to the parameters. Progressively, OAISIM will write down raw emulation data on the NFS Share. Once the emulation is over, the web server will get the raw data and generate it as results for the user. Because installing DRUPAL on a single machine is not really easy and worthwhile, a portable version is also implemented using simple PHP pages. These PHP pages are clones of the PHP pages made for the DRUPAL version, so the user interface between the remote and the local versions is similar.

The achievement of a simulation using the web portal frontend requires four steps: first the user must defines the system configuration (form of Figure 3-18). Second, it should define the topology of the simulation including the area definition, the network type, the cell type and the mobility model(from of Figure 3-19). Third, it configures the application behaviour, including the traffic pattern and the application type(form of Figure 3-20). Finally, it defines the duration and the log pattern, which specifies the nature of the requires trace, including the log level, the target layer and the chosen variables.



Environment and System Configuration	
Fading	Other parameters
Pathloss model	Wall penetration loss 5 dB
Pathloss Exponent: 2.0 (2 to 6)	System Bandwidth 7.68 MHz
Pathloss Zero Db: -50 dB (0 to -150)	UE Frequency 1.9 GHz
Small Scale	
☐ SCM_A	
☐ SCM_B	
☐ SCM_C	
☐ SCM_D	
☐ EPA	
☐ EVA	
☐ ETU	
☐ Rayleigh8	
☒ Rayleigh1	
☐ Rayleigh1_corr	
☐ Rayleigh1_anticorr	
☐ Rice8	
☐ Rice1	
☐ Rice1_corr	
☐ Rice1_anticorr	
☐ AWGN	

Antenna	
eNB Antenna :	UE Antenna :
RX Noise Power 5 dB	RX Noise Power 1 dB
Number of Sectors 3	Antenna gain 0 dBi
Beam Width 1.1 dB	TX Power 20 dBm
Antenna gain 16 dBi	Nb antennas RX 1
TX Power 40 dBm	Nb antennas TX 1
Nb antennas RX 1	
Nb antennas TX 1	

Figure 3-18: The web portal provides an easy from to define the environment and system configuration with ample documentation of different options.

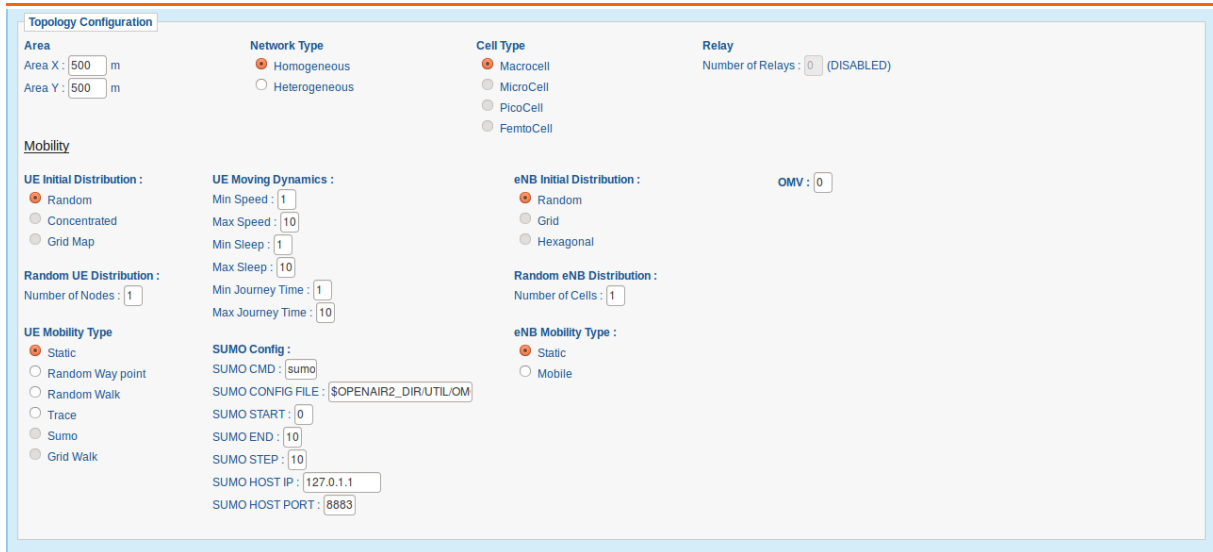


Figure 3-19: The definition of the topology is simplified by a user-friendly form.

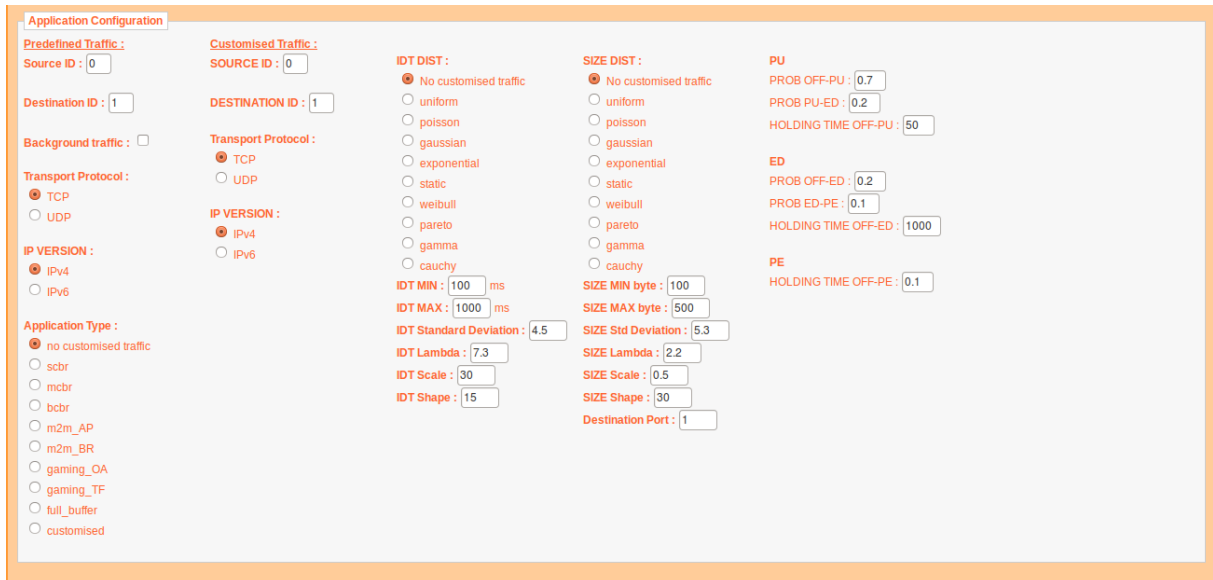


Figure 3-20: The definition of the application behaviour relies on predefined pattern.

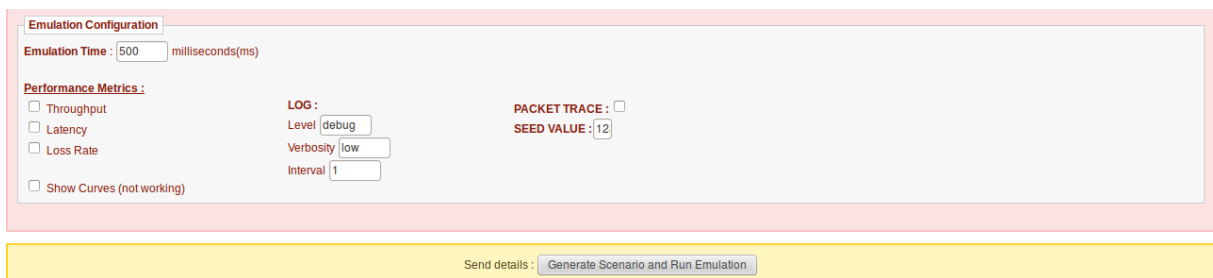


Figure 3-21: The Emulation configuration defines particularly the duration and the log pattern.

3.2.4.3 OAI Virtual Machine

The last method to use OAI is to download a full virtual machine of the OS including the correct installation of all libraries. We make use of VMware to bringup the OAI emulation environment.

To start, first download the virtual machine from <https://emu.openairinterface.org/openairlab/openairlab.zip> and unzip the file, then go to the corresponding directory, and click on "Ubuntu.vmx", and select "moved it" to relocate the image of virtual machine. Then click on "start virtual machine". You should see the Ubuntu 11.10 starting. If needed, use the full screen mode. The login and password of VMware is "**openair**".

Go to the directory openair4G, and perform "svn update" to get the latest version of the oai from the repository. Note that the svn access for this user is READ-ONLY. Then you can start using OAI on your local machine. For more information, please refer to appendix.

4. Preliminary Plans of Activities and Policies

This Section gives details regarding the plans of activities for the next six months (M7 to M12), leaving to the annual Deliverables (to be delivered at M12) the definition of the activities planned for the next years. The Section also provides description of the policy agreed to give access to the lab facilities.

4.1 Introduction

Track #2 intends to create a framework that will survive to the end of Newcom#. Therefore, on the one hand the plan of activities has been devised as a component of the NoE; on the other, all dissemination and promotional actions are conducted under a more general perspective. For the same reason, the access policy has been discussed under a long-term vision.

4.2 Steps Followed

The achievements of Track #2, after the first six months of Newcom# fully dedicated to the lab set-up and the definition of the activity plan, have been reached through the following sequence of steps and actions:

- Set-up of the facilities and creation of the documentation needed for the open access at the individual site premises, since kick-off till M6; it is envisaged that the sites will undergo continuous update during the course of Newcom#;
- Face-to-face meetings among the EuWIn Director and the three WPLeaders to discuss about EuWIn goals and plans, at the Newcom# kick-off meeting (November 2012, Pisa) and the Track #1 meeting (March 2013, Paris);
- Monthly skype meetings among the EuWIn Director and the three WP Leaders to discuss the status of lab set-up and coordinate activity plans;
- First presentation of the EuWIn scopes and facilities in Pisa, during the Newcom# kick-off meeting;
- A Track #2 session dedicated to the discussion of JRAs and presentation of EuWIn facilities in Paris, on March 7, during the Track #1 meeting;
- A Track #2 web meeting held on April 9, discussing among all researchers involved, the access policy and the plan of activities.

4.3 Plans of Activities

At time of preparation of this Deliverable, the following plans have been made. They are presented through the same itemised list reported in subsection 2.3.

1) A Unique Portal

The first release of the website is made public at the end of M6, time of delivery of this Deliverable. The website will be updated every six months, based on the achievements at the end of each period.

2) Web Meetings

Web meetings will be organized based on the request of Newcom# researchers. So, no periodic meetings are planned.

3) Workshops

The first Newcom# Emerging Topic Workshop (ETW) will be dedicated to the theme that represents the motto of EuWIN: "Fundamental Research Through Experimentation". It will be held in Bologna, at the premises of the EuWIN site, on July 8-9. The TPC of the workshop includes the three Track #2 WP Leaders, the EuWIN Director and Dr. Florian Kaltenberger of EURECOM. It will include three sequential sessions: one will be dedicated to industry speeches, one to presentations given by representatives of other EC funded projects, and the last one to talks given by Newcom# researchers involved in some experimental activities. At time of writing, a list of potential industry speakers and projects has been defined, and the invitations are sent. The final programme will be ready after delivery of this Deliverable, and will be posted on the EuWIN website under "EVENTS". The workshop intends to group together separate experiences carried out in Europe, showing how experimental testbeds can be useful to fundamental research and the development of new techniques/protocols.

4) Training Schools

The first Newcom# Training School will be dedicated to the topic of interference management. It will be held in May 28-31, 2013 in Sophia-Antipolis at the premises of the EuWIN site of EURECOM. One of the four days of the school will be fully dedicated to experimental activities, involving the facilities of EURECOM and CTTC. The programme is available in the EURACON website www.euracon.org (EURACON is the association in charge of administration, logistics and publicity of the School).

The third Newcom# Training School will be dedicated to the topic of experimental research. It will be organized, jointly with the COST Action IC1004, in November 2013 in Barcelona at the premises of the EuWIN site. The school will comprise oral, poster and lab sessions. The programme will be defined before end of summer 2013. The TPC includes the three Track #2 WP Leaders, the EuWIN Director, Dr. Florian Kaltenberger of EURECOM, Prof. Claude Oestges of UCL. The programme will be available in the EURACON website www.euracon.org.

5) Industry Liaisons

During the third week of June 2013 (most probably on June 17) Newcom# will visit the Orange Labs at Issy-Les-Moulineaux, in France. The three EuWIN sites will present their facilities in that context.

6) Demonstration activities

The inaugural event of EuWIN will take place in Bologna, at the premises of the EuWIN site, on July 8, 2013. During this event, all EuWIN facilities will be presented to the public, both through talks and demos/posters. The event will be professionally video recorded in order to prepare two video trailers (one and five minutes, respectively) to be used for promotional purposes later. Industry representatives and researchers from other EC projects will be invited. EC officers will also be invited. The event will be followed (till the morning of the 9th) by the Emerging Topic Workshop and (from the afternoon of the 9th till the 10th) by the Track #2 meeting, which will in-

clude parallel networking sessions and plenary talks to present to all Newcom# researchers the scopes of Track #2 JRAs. The entire event will be a major occasion to show EuWIn facilities to many researchers and industry representatives.

7) Experimental Tours

This line of activity deals with PhD students visiting the three sites in sequence to perform measurements that are closely related and that are made toward a unified scope. This kind of activity is planned for the second year of Newcom#.

4.4 Access Policy

The policy for access to the EuWIn facilities has been thoroughly discussed by the WP Leaders under the coordination of the EuWIn Director, and the following decisions have been made. The discussion is reported based on the definition of the EuWIn spheres presented in this Deliverable.

The Newcom# partners.

They are given free of charge and prioritised access to the facilities (this was a commitment taken at time of DoW approval).

The Newcom# associate institutions.

As detailed in the deliverable D43.1 ("The NEWCOM# Handbook), the NoE has two types of Associate Partners: Type I, which are those more interested in conducting dissemination-related activities in collaboration with project partners; and Type II, the main aim of which is to conduct joint research activities in collaboration with project partners. Their number is not defined at time of writing of this Deliverable, since Newcom# can decide to accept new associate partners during its life. Access to EuWin will be granted to Associate Partners, although in the case that a large number of partners is accepted the EuWIn sites can not commit to provide access to all of them. The general policy is that EuWIn intends to be as much open as possible, given the resource constraints (in terms of space, and availability of researchers for support). Therefore, it is decided that all Newcom# Associate Partners (both Type I and Type II) will have free of charge and open access to all information and data made available through the repository of the EuWIn website. On the other hand, physical access to the lab facilities will be as a rule restricted to Type II partners that have signed a specific agreement of cooperation with the NoE based on requests raised by the associate institutions (with specification of the goals and the mutual benefits), and on the availability of resources. These requests will have lower priority with respect to those coming from Newcom# full partners, and will be processed according to an order which will be defined by the WP Leader for each EuWIn site, depending on the mutual benefits expected. Evidence of the access policy above will be given with publication on the (freely accessible) EuWIn website.

Revision of the rules of access to EuWIn will be performed every 12 months after the release of the present document, according to the experience gained during the previous operating period.

A welcome letter will be sent to new associate institutions upon previous

agreement with one of the EuWIn WP Leaders, informing about the general policy of collaboration. These institutions will be listed on the EuWIn website for better visibility.

External Institutions.

In this case the policy depends on the specific platform and EuWIn site. The WP Leaders will decide whether to provide access to the databases and/or the facilities, and whether free of charge or based on the payment of a fee, on a case-by-case basis, in agreement with the guidelines and the approval of the Executive Board.

5. Conclusions

This deliverable summarizes the EuWin@Eurecom facilities that are made available in the context of Newcom#. The highlight of those facilities are 10 brand-new ExpressMIMO2 radio acquisition cards, which can be used for simple offline experimentation using a Matlab/Octave interface or for real-time processing using the Openairinterface LTE software modem. The facilities also include a high power base station with an experimental license for TVWS, the 1.9GHz and the 2.6GHz bands as well as experimental vehicles for drive tests. Last but not least, some of the facilities, such as the OpenAirInterface simulation and emulation tools, are also available remotely.

This deliverable marks the official start of the lab, so we welcome joint research actions (JRAs). We briefly list some JRAs that have already been identified together with the partners that have expressed interest.

- Low-power architectures for channel decoding (CNIT-Polito, CNRS-Eurecom, Bilkent, CTTC, Inov)
- Cellular broadcasting (CNRS-Supelec, CNRS-Eurecom, Ucam)
- PHY abstraction (Eurecom, IASA, Bilkent)
- Internet of things (CNIT-Bologna)

These and more JRAs will be reported in more detail in the next deliverable.

6. Appendix

6.1 ExpressMIMO2 Software Framework

6.1.1 Architecture

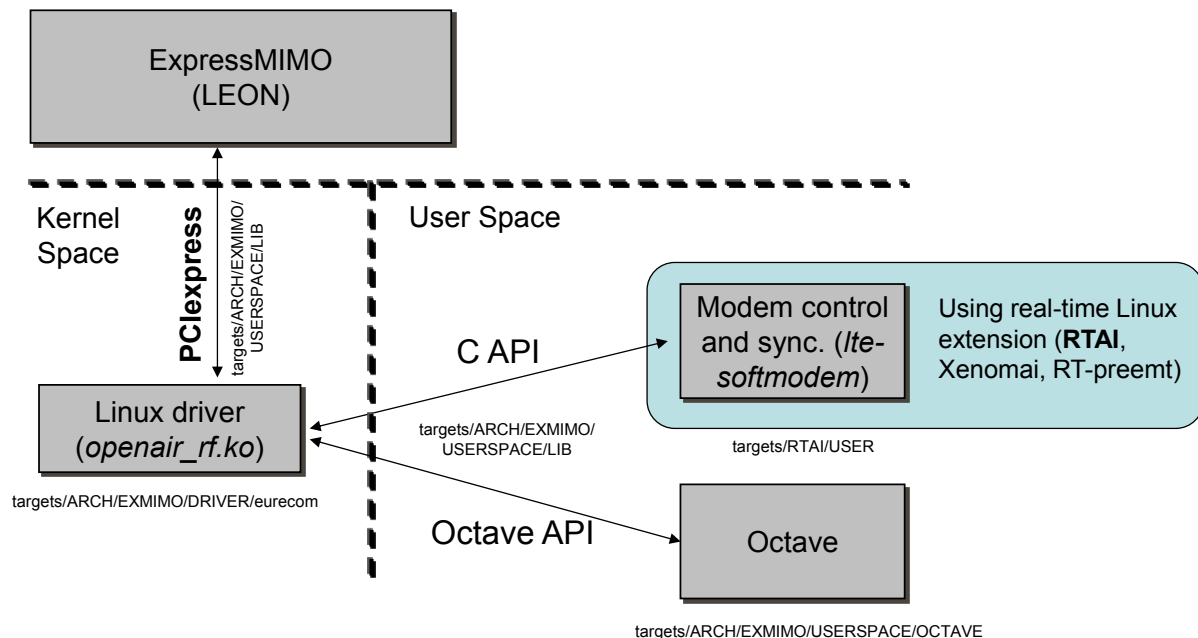


Figure 6-1: ExpressMIMO software framework

An overview of the software framework for ExpressMIMO is depicted in Figure 6-1. The main components are the Express MIMO card, its Linux driver module (*openair_rf.ko*), the API, and the applications. The driver module and the firmware are further described in Section 6.1.2. The basic API is written in C and described Section 6.1.3. The API can also be used in non-real-time mode from Octave, which is described in more detail in Section 6.1.4. For real-time operation, a real-time extension for Linux, such as RTAI is needed. We will describe the current version of the OpenAirLTE software modem as an example application in Section 6.2.

6.1.2 Driver and firmware update

The *openair_rf* driver establishes communication between the card and the PC.

6.1.2.1 Prerequisites

Before compiling you need to set `OPENAIRx_DIR` to the root directory of *openairx*:

set `OPENAIR_HOME` variable and export the following:

```
export OPENAIR1_DIR=$OPENAIR_HOME/openair1
export OPENAIR2_DIR=$OPENAIR_HOME/openair2
export OPENAIR3_DIR=$OPENAIR_HOME/openair3
export OPENAIR_TARGETS=$OPENAIR_HOME/targets/
```

6.1.2.2 Openair kernel modules

To operate the [ExpressMIMO](#) cards, you need a hardware driver (linux kernel module). This driver is located in `$OPENAIR_TARGETS/ARCH/EXMIMO/DRIVER/eurecom`. RTAI is no longer required to build this driver. RTAI is only required for the real-time operation of the card (see below).

To build the driver just call "make" from \$OPENAIR_TARGETS/ARCH/EXMIMO/DRIVER/eurecom. It will produce a file "openair_rf.ko" in the same directory. Insert the module using "sudo insmod openair_rf.ko".

6.1.2.3 Firmware upload

Before using the card, the firmware needs to be loaded in the card. The program to do this is located in \$OPENAIR_TARGETS/ARCH/EXMIMO/USERSPACE/OAI_FW_INIT. To build the tool call "make" from that directory.

In the same directory there is the compiled firmware for both [ExpressMIMO1](#) (sdr_expressmimo1) and [ExpressMIMO2](#) (sdr_expressmimo2). If you want to compile the firmware from scratch, look into the openair0 SVN repository, folder exmimo2/software/sdr.

To load the firmware into the card, type "updatefw -s 0x43ffff0 -b -f sdr_expressmimox", where x has to be replaced with the Express MIMO board version.

Type "dmesg" to check that the card is operational. You should see the message "[LEON card0]: ready".

6.1.2.4 Scripts

Everything can also be compiled comfortably from the directory \$OPENAIR_TARGETS/RTAI/USER using "make drivers". There are also the scripts init_exmimo.sh or init_exmimo2.sh to load the driver and initialize the firmware.

6.1.3 C API

The API is used to build applications on top of the ExpressMIMO cards. It is located in targets/ARCH/EXMIMO/USERSPACE/LIB/openair0_lib.h providing the following functionality:

- int openair0_open(void);
 - ☞ Initializes PCI interface openair0_exmimo_pci
 - ☞ this also contains pointers to signal buffers and counters
 - ☞ see targets/ARCH/EXMIMO/DEFS/pcie_interface.h
- int openair0_close(void);
- int openair0_dump_config(int card);
 - ☞ Dumps openair0_exmimo_pci to the card
- int openair0_get_frame(int card);
- int openair0_start_rt_acquisition(int card);
- int openair0_stop(int card);

6.1.4 Octave API wrappers

Octave wrapper functions for the above API are located in targets/ARCH/EXMIMO/USERSPACE/OCTAVE. They give access to some of the original AI functions

- oarf_config_exmimo (see online help for parameters)
- sig = oarf_get_frame(card)
- oarf_send_frame(card,sig,nbits)
- oarf_stop(card)

To compile the .cc to .oct files (note: octave-headers needs to be installed), do

- make clean
- make oarf
- make gpib (if you want gpib)

Examine rx_spec.m as an example

6.2 Real-time operation of OpenAirInterface LTE software modem on ExpressMIMO2

6.2.1 ExpressMIMO data acquisition

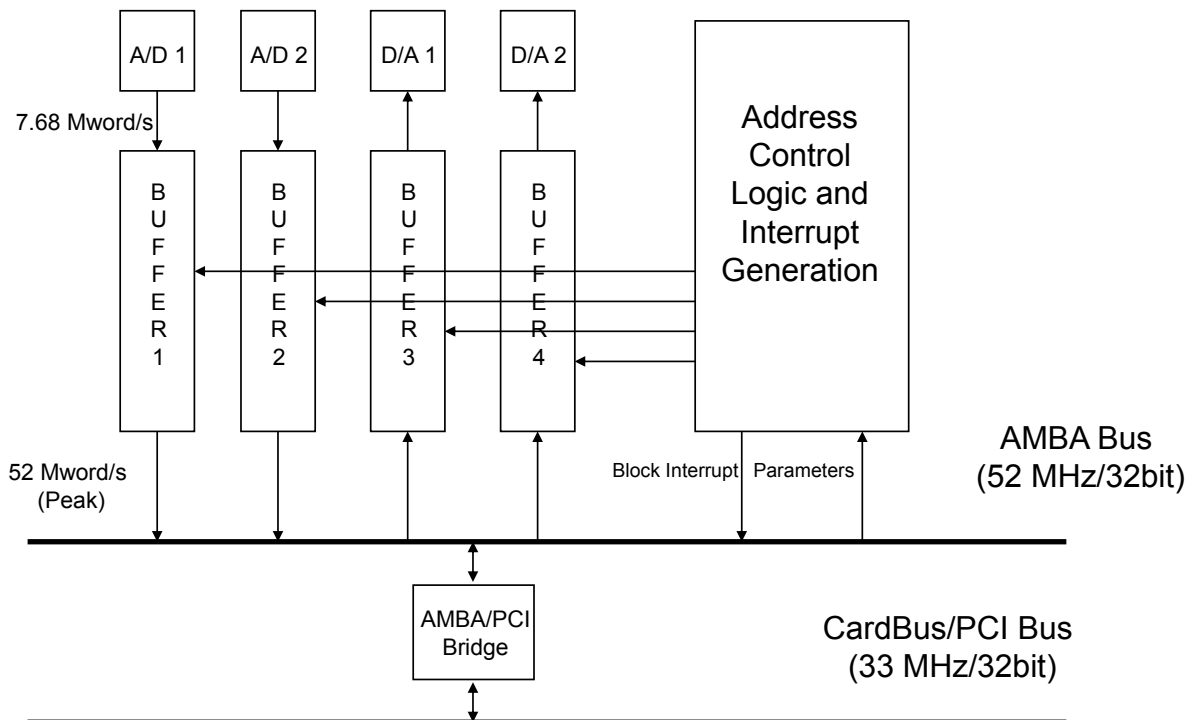


Figure 6-2: ExpressMIMO data acquisition system

- AMBA can burst at a peak rate of 52 MHz, very comfortable.
- PCI DMA controller (GRPCI) on AMBA can't do quite this but it's close enough
- Acquisition unit stores blocks (minimum 2) of a programmable size ($\leq 1\text{Kbyte}$) and generates an interrupt to CPU at the end of each block. The CPU programs a 2 DMAs (one for each chain) AMBA->PCI (RX) or PCI->AMBA (TX)
- TX and RX cannot occur at the same time (time-division duplex)

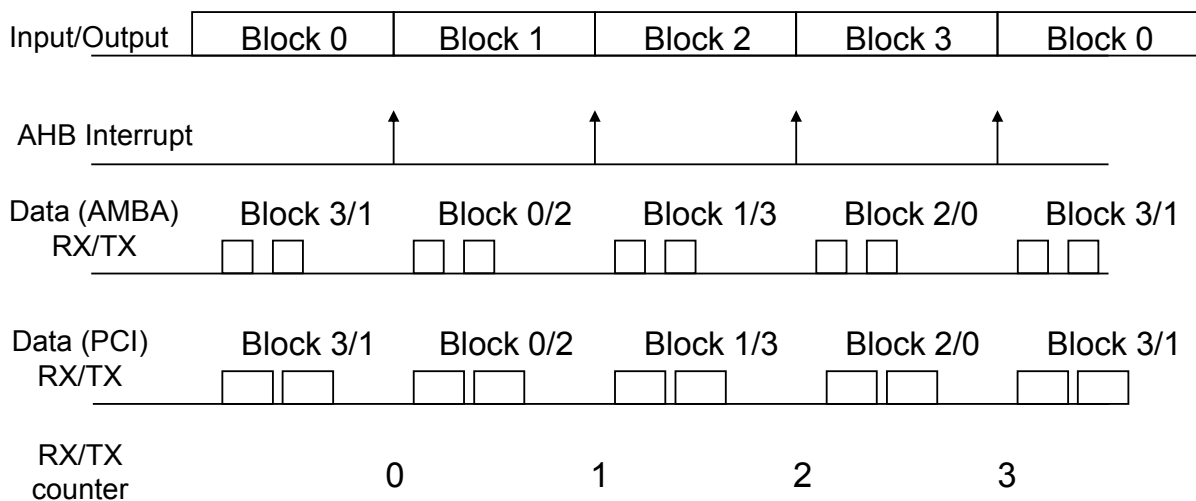


Figure 6-3: Data acquisition timing diagram

6.2.2 Synchronization of the software modem

As explained above, the data acquisition system directly writes and reads signals to the PC memory using DMA transfers over the PCIe express bus. The API described in provides access to these buffers and the corresponding counters over the `openair0_exmimo_pci` structure. The counters are used by the `lte-softmodem` application to synchronize to the Express-MIMO card.

A prerequisite for real-time operation is a real-time extension to the Linux kernel. Currently RTAI is used, but others are possible too (Xenomai, RT-preempt). This assures that threads are served with minimum latency and highest priority.

The `lte-softmodem` application is located in `targets/RTAI/USER/lte-softmodem.c`. It provides the following basic operations:

- **Top-level UE/eNB real-time thread**
 - Synchronization to TX/RX counter using polling and sleep
 - Invokes inner-modem DSP processing (`phy_procedures_lte_ue` or `phy_procedures_lte_eNB`)
- **Other real-time threads for ULSCH/DLSCH decoding**
 - Woken up on reception of a ULSCH/DLSCH
 - Make use of multi-core CPUs to parallelize inner-MODEM and channel decoding (turbo-decoder)
 - Multiple decoding threads for higher throughput

6.2.3 Compilation and usage

This page describes how to run [OpenAirInterface](#) LTE softmodem in real-time on the [ExpressMIMO](#) cards in user-space using RTAI LXRT.

6.2.3.1 Prerequisites

- Make sure that the latest version of RTAI is installed and working (see [OpenAirKernelMainSetup](#))
- Make sure that the [ExpressMIMO](#) card is operational (see [ExpressMIMO](#))
- Insert the `rtai_lxrt.ko` module, which is the user-space version of `rtai_sched.ko`. You must therefore first remove `rtai_sched` and all its dependencies (using `rmmod`) and then insert `/usr/realtime/modules/rtai_lxrt.ko` and all the others you removed before. For the last step you have to use `insmod` instead of `modprobe` because `modprobe` will re-insert `rtai_sched.ko` which you don't want (Note from Ruben: basically, `modprobe` wants to satisfy the dependencies and by loading `rtai_sched`, there is a symbol conflict). You can do that by using the attached script `start.sh`.
- You could also set the following paths (you might want to include these in your `.bashrc` file)

```
export PATH=$PATH:/usr/realtime/bin
```

```
export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:/usr/realtime/lib
```

- If the latter does not work you can also create a file `/etc/ld.so.conf.d/rtai.conf` with the content `"/usr/realtime/lib"` and run `ldconfig`
- Also make sure that `asn1` is set up correctly for use with RTAI. See the README in `$OPENAIR2_DIR/RRC/LITE/MESSAGES` for details.

6.2.3.2 Compilation

- `cd $OPENAIR_TARGETS/RTAI/USER`, `make clean`, `make lte-softmodem DRIVER2013=1`
- Further compile options are:
 - `XFORMS=1`: compile with `xforms` enabled
 - `OPENAIR2=0`: disables all layer 2 and above
 - `NAS=1`: compiles with NAS support enabled (requires NAS driver)
 - `EMOS=1`: compiles for EMOS channel sounding (only validated together with `OPENAIR2=0`)

- HARD_RT=1: enables hard real-time support
- you should have an executable called lte-softmodem

6.2.3.3 Usage

- lte-softmodem command line parameters:
 - -d: enables xforms
 - -U: start as UE (default: eNB)
 - -C f: set carrier frequency to f0 for all chains (extension to independent frequencies trivial)
 - -S: start fs4 test
 - -T: unsupported (set TCXO for CBMIMO1)
 - -F: calibration filename (see details below)
 - --calib-ue-rx: run in calibration mode with high gain
 - --calib-ue-rx-med: run in calibration mode with medium gain
 - --calib-ue-rx-byp: run in calibration mode with bypass amplifier
 - --debug-ue-prach: only send prach but do not connect
 - --no-L2-connect: only runs rx, does not send prach
- The preferred way to run lte-softmodem is by using make run_UE0 or make run_eNB0 (more targets exist depending on calibration files, see below. This basically calls rtai_load, which in turn uses the command line parameters from the file .runinfo along with some other LXRT related commands.
- You can also run lte-softmodem directly using sudo. This is useful for debugging with gdb.

6.2.3.4 Calibration files

- to properly operate the [ExpressMIMO](#) card, several calibration files are needed. Each file has a root name (for example ex2_2 for [ExpressMIMO2](#) #2) and one of the following extensions
 - rfdc.lime: DC calibration
 - rflo.lime: LO calibration
 - rxg.lime: RX gains
 - txg.lime: TX gains
- see the comments in the calibration files for details
- see also [ExpressMIMOCalibration](#) for details on the calibration procedure

6.3 OpenAirInterface system level emulator

Openair emulation platform allows a real-time repeatable and controlled set of experiments for a small-to-large scale emulated network in the openairinterface laboratory environment applicable to both evolving cellular technologies (LTE/LTE-Advanced) and rapidly-deployable mesh/ad-hoc networks. The Objective of this emulation platform is to fill the gap between the simulation and real experimentation by providing the baselines for protocol validation, performance evaluation and pre-deployment system test.

In the following, we first present the experiment design workflow where we summarize different emulation steps. Then, the emulation kernel and its software architecture are described. Afterwards, we analysis the performance of the emulation platform based on software profiling output, discuss the identified weaknesses and limitations, and propose new solutions towards the target scenario of testbed1. Finally we describe the adapted hardware setup allowing us to achieve large scale experimentation.

6.3.1 Experiment Design Workflow

A sequential experiment workflow, where the output of each step will be the input of the next, is employed to allow an experiment to be reproduced. Five consecutive steps are defined:

scenario description, configuration, execution, monitoring, and analysis, where each step is split into several sub-steps as explained in the following subsections (see Figure 6-4).

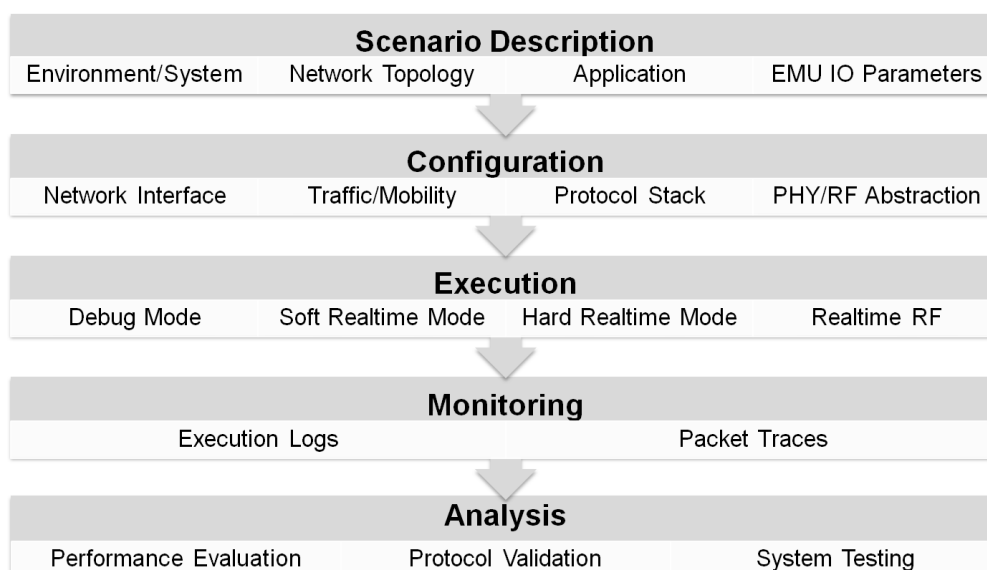


Figure 6-4: Experiment Design Workflow

6.3.1.1 Scenario Description

This step builds a complete xml layout of an experiment. This step is divided into four sub-steps: (i) system/environment, where system (e.g. bandwidth, frequency, antenna) and environment (e.g. pathloss and channel models) parameters are defined; (ii) network topology, where network area, network topology (i.e. cellular, mesh), nodes' type, initial distribution, moving dynamics (speed, pause-time) and mobility model (e.g. static, random way point, grid) are set; (iii) application, where real application and/or emulated traffic pattern in terms of packet size and inter-departure time are defined; (iv) EMU IO Parameters, where supervised parameters (e.g. emulation time, seed, execution logs, performance metrics) and analysis method (e.g. protocol PDUs and operation) are set.

6.3.1.2 Configuration

This step defines a sequence of components' initialization based on the scenario description. It includes four sub-steps: (i) network interface, where the OAI IP interface is configured, (ii) traffic and mobility, where traffic pattern and mobility model parameters are set, (iii) protocol stack, where protocols are configured given the network topology and PHY abstraction, where a channel model predicting the modem performance is configure.

6.3.1.3 Execution

This step defines the execution environment for the emulator in order to synchronize the emulated nodes and run the experimentation. It includes four execution modes: (i) debug mode, where the emulation is executed in user space without any Linux IP connectivity, (ii) soft real-time mode, where the debug mode is interconnected with the Linux IP protocol stack and calibrated to respect the layer 2 frame timing on average, (iii) hard real-time mode, where the emulation is executed in RTAI kernel with Linux IP protocol stack respecting layer 2 frame timing strictly, and (iv) real-time RF mode, where hard real-time mode is extended with the RF equipment interconnected with an attenuator.

6.3.1.4 Monitoring

This step defines how the experiment is monitored (passive and/or active). It includes: (i) execution logs, where experiment traces and logs are collected, labeled and archived, (ii) packet traces, where protocol signaling is captured and stored during the experiment.

6.3.1.5 Analysis

This step processes raw data and produces results and statistics. It includes three non-exclusive analysis objectives: (i) performances evaluation, where the key performance indicators are measured and evaluated, (ii) protocol validation, where the protocol control- and user-plane signaling are validated versus protocol specification, and (iii) system testing, where the system as a whole is analyzed and tested.

6.3.2 Emulation Kernel

The emulation kernel is based on a discrete frame generator (i.e. emulator scheduler), where the chronological sequence of frame/subframe are generated by the system. It is built around five architectural design choices to increase the scalability and facilitate the emulation of a realistic application scenario: (i) real protocol stack, where protocols are implemented as in a real system (not modelled); (ii) protocol virtualization, where the emulated network nodes although separated share the same operating system instance and Linux IP protocol stack within the same physical machine; (iii) parallel processing, where virtualized protocol instances and CPU expensive functions are executed in separate threads; (iv) transparent emulated data flow, where emulated data are exchanged either via a direct memory transfer when they are part of the same physical machine or via multicast IP over Ethernet when they are in different machines, (v) end-to-end validation, where real application client/server are attached on the top of some emulated nodes through legacy IPv4/v6 connectivity and the remaining traffic pattern and mobility model will be generated automatically as would be the behaviour of the real application at the client and server sides.

6.3.2.1 Real Protocol Stack

OAI provides a complete protocol stack for cellular and mesh network topologies applicable to both emulation and real-time RF platforms. At the access layer, it implements the MAC (medium access control), RLC (radio link control), PDCP (packet data convergence protocol), RRC (radio resource control) layers as well as full or abstracted PHY layer. It also provides an IPv4/IPv6 network device interface under Linux. The abstracted PHY is used to enable fast emulation of complex network by predicting the modem performance in terms of BLER (block error rate). It injects error patterns for each transport channel at the receiver of each emulated node based on the wideband SINR, network topology, and propagation model or real channel traces.

The software implements the standard network socket interface for IP layer and a generic request/indicate interface among other layers making the design highly modular. The generic interface also provides a loopback interface at each (sub-)layer for testing and validation purpose.

6.3.2.2 Protocol Virtualization

OAI provides virtualization of the network nodes within the same physical machine to increase the scalability of the emulation platform. Protocol virtualization consists of sharing the same operating system instance and Linux IP protocol stack for independent emulated node instances. It allows networks nodes to coexist in the same execution environment. In some cases, the use of OS virtualization tools such as UML may be needed for the development and performance evaluation of sophisticated layer 3 protocols. Note that, protocol virtualization offers the same functional properties (i.e. services) and the same nonfunctional properties (i.e. performances) than that of a real protocol.

6.3.2.3 Transparent Emulated Data Flow

To increase scalability and allow experimentation of a complex network topology and architecture, two emulated data flows may coexist between emulated nodes. Either nodes communicate via direct memory transfer or via IP multicast (over Ethernet) depending on whether they are part of the same physical machine or not. From the point of view of the protocol stack, the data flow is transparent and that the network connectivity is independent from the node distribution on a physical machine.

6.3.2.4 Parallel Processing

To achieve scalability and exploit the current multi-core hardware architecture, the emulation kernel implements two type of parallelism: CPU and joint CPU-GPU. The CPU parallelism exploits the multi-process/thread capability of CPU by separating the emulated nodes from the CPU-expensive functions such as channel model. This could be extended by separating each emulated node from each other either through threads or processes. The CPU-GPU parallelism is mainly used to offload the CPU-expensive function onto GPU (to increase the scalability) and keep all the emulated nodes in the CPU.

6.3.2.5 Flexible End-to-End Validation

Real applications can be attached on the top of emulated nodes via the network interface, which provides Linux IP interconnection with QoS classification on layer 2 resources. Such an application may interact with an external application through an emulated or true core network to provide end-to-end IP connectivity. The remaining traffic pattern and mobility model will be automatically generated as would be the behavior of the real application at the client and server sides.

6.3.3 Software Architecture

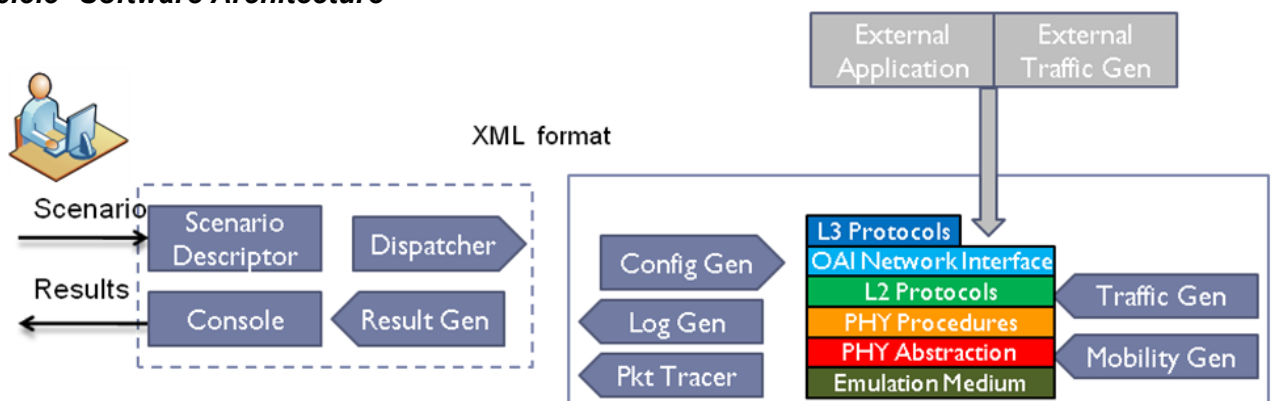


Figure 6-5 shows the high level software architecture of the emulated platform itself and the main building blocks. The user scenarios are described using baseline scenario descriptor and then encoded into xml format and dispatched across OpenAirInterface hardware platform according to some predefined rules. Then, the config generator block translates the high level scenarios into low level configuration files understandable for traffic/mobility generator, UE/eNB protocol stack, phy abstraction and emulation transport medium. The real applications are attached on top of some emulated UE/eNB and the remaining traffic pattern and mobility model will be generated automatically. The behavior of the wireless medium is modeled using a PHY abstraction unit which emulates the error events in the channel decoder and provides emulated measurements from the PHY in real-time. The remainder of the protocol stack for each node instance uses a complete implementation as would a full-RF system. The Log generator is in charge of recording the emulator activities according to the user defined log level, while the packet trace generator captures the frame (potentially extract on-the-fly the relevant field). The result generator produces the user defined test results using the outcome of log generator and packet trace generator.

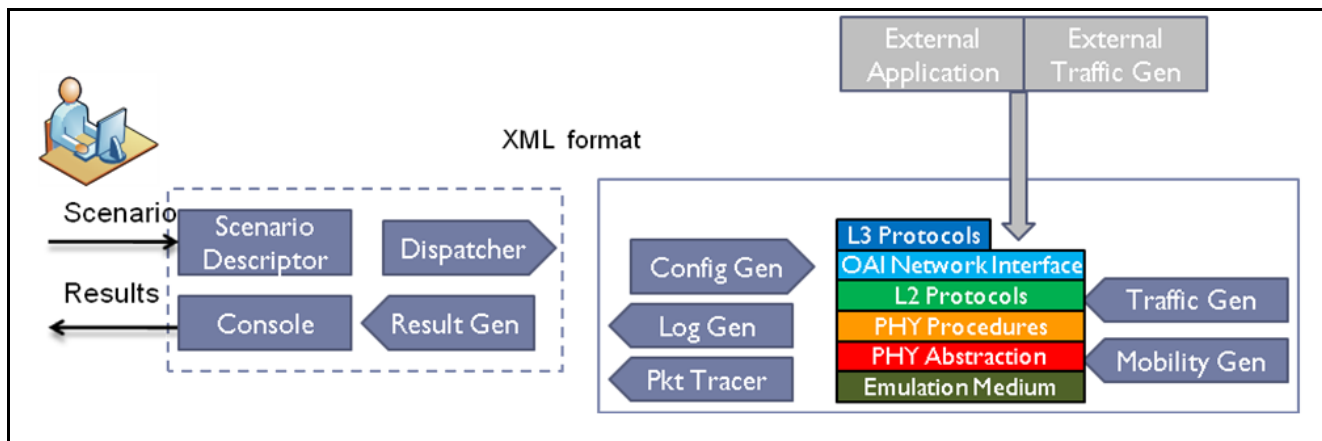


Figure 6-5: building blocks of OpenAirInterface Emulation Platform

The process of emulation has three phases, which are performed sequentially as follows:

1. **Input:**
 - a. Description of application scenario
 - b. Initialization and configuration of all the blocks
2. **Execution:**
 - a. Protocols: PHY procedures, L2 protocols, traffic generator, and packet tracer
 - b. Connectivity: PHY abstraction, channel model, and mobility model
 - c. Emulation data transport: IP multicast, shared memory
3. **Output:**
 - a. Protocol validations and execution logs
 - b. Performance evaluation metrics

The scenario is defined in xml format (see targets/SIMU/EXAMPLES/OSD/WEBXML), which is used to configure environment, network topology, application traffic, and emulation IO parameters. The behaviour of the wireless medium is modelled using a PHY abstraction unit which emulates the error events in the channel decoder and provides emulated measurements from the PHY in real-time.



Figure 6-6: Emulation process

6.3.4 Experiment 1: Analyze the control plane

1. Open a terminal using the left hand side bar
2. Run “cd Openair4G/targets/SIMU/USER”, to change the directory
3. Run “make clean; make cleanasn1; make”: clean the emulation and compile the entire emulator. You will get the **oaisim** binary file.
4. Check the options: `./oaisim -h`
5. Run “sudo wireshark”,
 - in the interface list, select local interface “lo”
 - in the filter type “!icmp”, i.e not icmp
6. Run
 - With full PHY: “./oaisim -A AWGN -s15 -P0 -n40 > log.txt”, to create an execution logs, see the output of “./oaisim -h” for help
 - With Phy abstraction: “./oaisim -a -A AWGN -s15 -P0 -n40 > log.txt”, to create an execution logs, see the output of “./oaisim -h” for help
7. Open the execution logs using an editor
8. You should be able to follow the following steps, the keyword are in **bold**.
9. Follow-up the procedure with the help of the figures and tables provided in the appendix.

6.3.4.1 Step 1: initial Sync

1. Make sure that “PBCH” is decoded successfully, and that the sync signal is sent to the upper layer.
2. Notice the parameters obtained through the PBCH, e.g. cell ID

6.3.4.2 Step 2: dissect the SIB1/SIB2/SIB3 content using wireshark

1. In the wireshark, select the MAC LTE BCH PDUs, then go the “packet detail view”, select the MAC LTE BCH PDUs, then right click on it and select “expand subtree”
2. Identify the **SIB1**, and **SIB2**, and **SIB3**
 - a. eNB Encode SI (**SIB1**, **SIB2**, **SIB3**), and sends it through the BCCH channel
3. Notice the content, and notice the period between each transmission

The content of SIB1/2/3 are used to configure the PDCP/RLC/MAC/PHY parameters. The configuration is send by the eNB in DL. In the code you can see the following log:

- [MAC][I][CONFIG][UE 0] Configuring MAC/PHY from eNB 0

6.3.4.3 Step 3: RRC CONNECTED

1. **RRC_SI_RECEIVED** (SI-RNTI)
 - UE received **SIB1** through BCCH
 - UE received **SIB2**, **SIB3** through BCCH
2. **RAPROC** MAC Random access procedure (RA-RNTI, MAC Attached)
 - Random access preamble **PRACH** : **UE -> eNB**, **msg1 → PRACH**
 - i. the value of RA-RNTI, and that of the preamble index selected by UE
 - ii. the eNB detects the preamble, and adds this UE
 - Random access response **RAR** : **eNB->UE**, **msg2 → PRACH**
 - i. eNB generates the c-RNTI

- ii. UE receives the C-RNTI, as well as timing advance, notice all the parameters
- 3. **RRC_CONNECTED** (RRC connection establishment, successful)
 - **RRCConnectionRequest : UE→eNB, msg3 → CCCH, RLC-TM**
 - i. UE generates/sends the msg3 and sets the contention resolution timer
 - ii. eNB receives the msg3 for that RNTI, and as a consequence it adds the user with that RNTI to the list of UEs, which it accepts the connections
 - **RRCConnectionSetup : eNB→UE, msg4 → CCCH, RLC-TM**
 - i. eNB first configures MAC/PHY
 - ii. eNB generates the msg4 and Piggybacks the content of msg3 (6 bytes), which is used for contention resolution at UE side
 - iii. UE receives the msg4, and clears the contention resolution timer (terminate RAPROC). This indicates a successful random access procedure.
 - iv. Notice the content of the msg4 decoded at the UE side. The msg4 is used to configure PDCP/RLC/MAC/PHY for the UE.
 - **RRCConnectionSetupComplete (UE→eNB) : msg5→DCCH, PDCP → RLC-AM**
 - i. UE generates the msg5 and sends it through **PDCP**.
 - ii. PDCP sends the msg5 to RLC AM, (**RLC_AM_DATA_REQ size 16 Bytes**)
 - iii. UE sends the BSR (buffer status report) level for LCID 1 (DCCH) is non-zero, (**BSR level 1**), this is the msg5. Note that prior to the transmission of BSR, UE sends a SR (scheduling request) and receives a SG (scheduling grant).
 - iv. eNB receives the msg5, and decodes it.

At the end of this procedure, UE is in RRC_CONNECTED state.

6.3.4.4 Step 4: Initial security activation

1. **SecurityModeCommand : eNB→UE on DCCH**
 - a. eNB generates and sends the securityModeCommand on DCCH
 - b. UE receives and decodes the securityModeCommand, configures the security mode for PDCP
 - c. Note the value of CipheringAlg, and IntegrityProtAlgor
2. **securityModeComplete: UE → eNB on DCCH**
 - a. UE generates and sends the securityModeComplete
 - b. eNB receives the securityModeComplete, and configures the PDCP

6.3.4.5 Step5: UE Capability Enquiry

1. **ueCapabilityEnquiry: eNB→UE on DCCH**
2. **ueCapabilityInformation : UE → eNB on DCCH**
 - a. eNB receives and decodes the ueCapabilityInformation
 - b. note the content of the message

6.3.4.6 Step 6: RRCConnectionReconfiguration

1. RRCConnectionReconfiguration : eNB->UE on DCCH

- eNB generates and sends the rrcconnectionreconfiguration message. Note the content, size.
- UE receives the message and configure MAC/PHY layer
- UE also configures the DRB 1 which corresponds to LCID 3 for PDCP/RLC/MAC/PHY

2. RRCConnectionReconfigurationComplete : UE->eNB on DCCH

- UE generates and sends the message
- eNB receives the message and reconfigure the DRB1 which corresponds to LCID 3

This procedure establishes a default data radio bearer (DRB) for best-effort data transmission.

6.3.4.7 Post-processing of logs

You can also further filter the messages by component, e.g. RRC and node type UE or eNB like this

- `cat lte.log | grep RRC > rrc.log` : RRC messages for both eNB and UE
- `cat lte.log | grep RRC | grep eNB > rrc_enb.log` : RRC messages for eNB

6.3.5 Experiment 2: Analyze the Data plane

The purpose of the second experiment is to follow the data plan by sending IP packet to the OAI network device driver. Figure below shows a time domain view. At the bottom are radio frames. A full frame is 10 ms but we normally think in terms of the 1-ms subframe, which is the entity that contains the transport block. Within the transport block is the MAC header and any extra space (padding). Within that there is the RLC header, then within the RLC header there can be a number of PDCPs. There is a somewhat arbitrary relationship between the IP packets coming in, which form the SDUs, and how the RLC PDUs are formed. Therefore you can make the maximum effective use of radio resources in a fixed period of time.

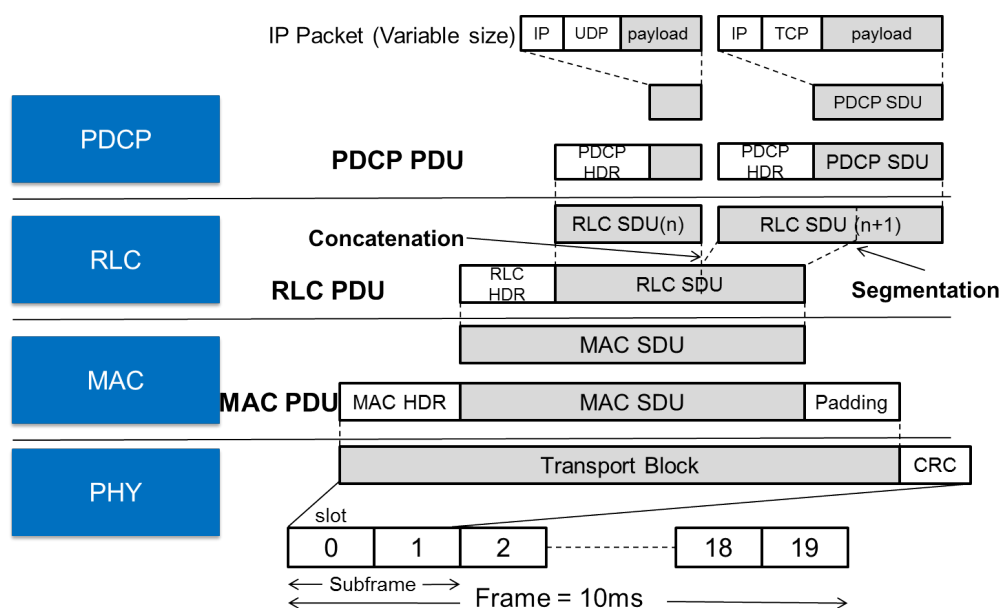
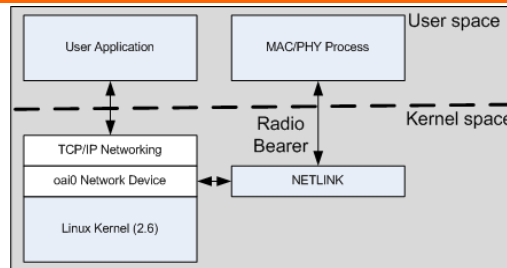


Figure below show the setup. As you see the driver in kernel space communicates with the LTE MAC/PHY process in the user space through the netlink sockets. We now need to setup the IP interface.



To accept IP connections (Non-access stratum)

1. Run “make clean; make all NAS_NETLINK=1”: clean the emulation and compile the entire emulator
2. Run “make nasmesh_fix” to compile the oai driver and insert the module to the kernel space, as well as the radio bear configuration tool, called “rb_tool” (see targets/SIMU/EXAMPLES/VIRT_EMUL_1eNB/ for more information)
 - Check that the driver is inserted into kernel using “dmesg”
3. Bring up oai0 interface for eNB and oai1 interface for UE (**note : passwd = openair**)
 - `sudo ifconfig oai0 10.0.1.1 netmask 255.255.255.0 broadcast 10.0.1.255`
 - `sudo ifconfig oai1 10.0.2.2 netmask 255.255.255.0 broadcast 10.0.2.255`
4. Associate the DRB id to the oai0 (**note : passwd = openair**)
 - `sudo $OPENAIR2_DIR/NAS/DRIVER/MESH/RB_TOOL/rb_tool -a -c0 -i0 -z0 -s 10.0.1.1 -t 10.0.1.2 -r 3`
 - `sudo $OPENAIR2_DIR/NAS/DRIVER/MESH/RB_TOOL/rb_tool -a -c0 -i1 -z0 -s 10.0.2.2 -t 10.0.2.1 -r 3`

Now, check the interfaces using ifconfig, and ping the interface address using “ping 10.0.1.1”, and “ping 10.0.2.2”

Note: To send an icmp packet from 10.0.1.1 (eNB) to 10.0.2.2 (UE), use “ping 10.0.1.2” (this is because when transmitting data among the virtualized protocol instances, the third and fourth byte of IPv4 address represents who is the source and destination, e.g. ping 10.0.1.2 means 10.0.1.1 is sending to 10.0.2.2). For this to work, you need to run the emulator (./oaisim -a >/dev/null).

In the following two subsections, we will trace the data packet in PDCP, RLC, and MAC sub-layers.

6.3.5.1 PDCP

PDCP (Packet Data Convergence Protocol) is a standard designated by 3GPP to transfer user/control plane data by maintaining sequence numbers so to provide in-order delivery and duplication-avoidance, to perform IP header compression (RFC 3095, Robust Header Compression) for better bandwidth utilization (*user plane* only), cipher/decipher functionality for security purposes, and integrity protection facilities (*control plane* only). PDCP functions are symmetrical for the uplink and the downlink.

Run the experiment without defining number of frames as follows:

- `./oaisim -a | grep PDCP`
- Open 2 new terminals and run the ping for both sides, i.e. “ping 10.0.1.2 -p 2B” and “ping 10.0.2.1 -p 2B”, where “p” specifies the pattern to be sent through ICMP message.

- You may change the payload size using “-s 256” or inter-departure time using “-i 0.5”

You should see that the PDPC of eNB (instance 0) is receiving the packet from IP (**IP->PDCP**), and later on the PDCP of UE (instance 1) transmitting the packet to IP (**PDCP->IP**).

Here are the questions :

- What is the size of IP packet? What is the size of PDCP PDU?
- What is the number of frames/sub-frames between PDCP PDU transmission and reception for the same PDU?
- Optionally: When RLC AM (Acknowledged Mode) is activated why there still is the need of PDCP to keep track of sequence numbers and handling retransmission?

6.3.5.2 RLC

RLC performs segmentation and reassembly and operates in three modes: transparent mode (TM), acknowledged mode (AM) and unacknowledged mode (UM). These are used by different radio bearers for different purposes. The RLC provides in-sequence delivery and duplicate detection. RLC functions are symmetrical for the uplink and the downlink.

Run the experiment without defining number of frames as follows:

- `./oaisim -a | grep RLC`
- Open a new terminal, and send one big packet like this
 - `ping -c 1 -s 1024 10.0.1.2 -p 2B`
- Open another terminal, and send one small packet like this
 - `ping -c 1 -s 5 10.0.1.2 -p 2B`

You should see that the packet is fragmented at the RLC of eNB (instance 0) with a variable size (resource block requested by MAC) and reassembled at the RLC of UE (instance 1). When all fragments of a given packet is correctly received, RLC of UE (instance 1) will pass it to PDCP.

Here are the questions :

- What is the size of RLC PDU?
- How many segments have been generated for the big packet and for the small packet?
- Which RLC mode is used for the data connection?

6.3.5.3 MAC

MAC functions are different in the uplink and downlink (not symmetric). Common functions include mapping between logical channels and transport channels, multiplexing/demultiplexing; specific eNB functions are transport format selection, and scheduling; specific UE logical channel prioritization, scheduling information reporting (**LCID**, **BSR**, **SR**).

Run the experiment as follows:

- Let the eNB ping the UE, i.e. “ping 10.0.1.2 -s 1024 -p 2B”
- Run “`./oaisim -a | grep SR_indication`”
- Run “`./oaisim -a | grep BSR`”
- Run “`./oaisim -a | grep LCID`”

You should see that the UE is sending a scheduling request (SR_indication) to eNB and followed by the buffer status report (BSR).

Comments and suggestions for the improvement of this document are most welcome and should be sent to:

project_office@newcom-project.eu



<http://www.newcom-project.eu>