OpenAirInterface

“promoting the development, distribution and adoption of the open source hardware and software wireless technology platforms”

Navid Niakein

EURECOM, Mobile Communication Department

This work is licensed under a CC attribution Share-Alike 3.0 Unported license.
OpenAirInterface in a Nutshell

- Open-source (hardware and software) wireless technology platforms for deployment of mock network with high level of realism
  - Soft modem: SDR architecture and full GPP
  - System approach with high level of realism
  - Part of FIRE facility: remote access

- Current focus
  - 3GPP LTE (unicast and multicast), and a subset of LTE-A features
  - IEEE 802.11p and 802.21
  - LTE meshing extension

- Objectives
  - Open forum for innovation in air-interface technologies and wireless networking through experimentation
  - Open platforms and proof-of-concepts through real-time prototypes and scalable emulation platforms
  - Dissemination, education, and training, project
OAI Wireless technology Platforms

- C implementation under realtime Linux optimized for x86
- Development and integration methodology
  - Tight interaction between the system emulation and soft modem
- Rich R&D environment:
  - Aeroflex-Geisler LEON/GRLIB, RTAI, Linux, GNU,
  - Wireshark, control and monitoring tools, message and time analyzer,
  - Low-level log processing, traffic generator, profiling tools, soft scope.
### IoT

- OAI eNB with a UE dongle (Huawei E392U-12)
- OAI eNB with smartphone
- OAI eNB with other EPC
Hardware Targets for Openair4G

- **ExpressMIMO2**
  - Designed and maintained by EURECOM
  - Used by many academia/industrial partners
  - 1.5/5/10/20 MHz, FDD/TDD (MIMO)

- **USRP B210**
  - Commercial Ettus/National Instruments board
  - Software drivers originally maintained by Orange Labs (Beijing), now officially supported by Eurecom
  - Not 100% LTE compliant (limited to 6.5Msps)

- **USRP X300**
  - Coming soon
Express MIMO 2

RF RX (4 way)

RF TX (4 way)

PCI Express (1 or 4 way)

4x LMS6002D RF ASICs
250 MHz – 3.8 GHz

Spartan 6 LX150T

12V from ATX power supply

GPIO for external RF control

©www.openairinterface.org
Software Architecture

ExpressMIMO2 (LEON)

Kernel Space

PCleexpress

Linux drivers

(nas_driver.ko)

Openair2/NAS/DRIVER/MESH/
openair2/NAS/DRIVER/LITE
openair2/NAS/DRIVER/CELLULAR

targets/ARCH/EXMIMO/DRIVER/eurecom

Linux network driver

Octave API

Octave

HW Space

User Space

Application

Control / Monitoring

Using real-time Linux extension (RTAI, Xenomai, RT-preempt)

Modem control and sync. (lte-softmodem)

targets/RTAI/USER

targets/ARCH/EXMIMO/USERSPACE/LIB

targets/ARCH/EXMIMO/USERSPACE/OCTAVE

©www.openairinterface.org - p 7
Emulator: in-lab system validation platform

**target:** scalability

### Input:
- Description of application scenario
- Initialization and configuration of all the blocks

### Execution:
- PHY procedures, L2 protocols, traffic generator
- PHY abstraction, channel model, and mobility model
- Emulation medium: shared memory

### Output:
- Execution logs
- System/protocol operation
- Key performance indicators: latency, jitter, throughput/goodput

#### Web Portal / Interface

- **Scenario Descriptor**
- **Dispatcher**
- **Console**
- **Result Gen**

#### Application
- **Traffic Gen.**

#### Traffic Gen.
- **Wireshark**
- **Log Gen.**
- **MSC & VCD**
- **Result Gen.**

#### Config. Gen.
- **L3 Protocols**
- **OAI Network Interface**
- **L2 Protocols**
- **PHY Procedures**
- **PHY / PHY Abstraction**

#### Path Loss

#### Channel Model

#### Mobility Gen

#### EMOS

#### Channel Trace
CLOUDIFICATION OF OPENAIRINTERFACE
Cloud Radio Access Networks

- **Two main steps:**
  
  - **Soft RAN:** to perform RAN functions on the top of general-purpose processor and not dedicated hardware such as DSP, FPGA, or ASIC.
  
  - **Virtual RAN:** to perform RAN functions on the top of virtualized cloud platforms sharing computing and storage capacity among different RAN functions.
Soft RAN
x86 Baseband DSP

- Challenge: efficient base band unit
- OpenAirInterface uses general-purpose x86 processors (GPP) for base-band processing
  - front-end, channel decoding, phy procedures, L2 protocols
- Key elements
  - Real-time extensions to Linux OS
  - Real-time data acquisition to PC
  - SIMD optimized integer DSP
    - 64-bit MMX → 128-bit SSE2/3/4 → 256-bit AVX2
  - Parallelism
  - x86-64: more efficient than legacy x86
Soft RAN
OAI BBU performance

- Three targets
  - Intel IvyBridge i5-3470 @ 3.2GHz → baseline
  - Intel Xeon E5-1607 v2 @3GHz → C-RAN
  - Intel Atom C2758 @2.4GHz → small cell

- Configuration
  - SISO
  - MCS index:
    - QPSK: 0, 4, 9
    - 16QAM: 10, 13, 16
    - 64QAM: 17, 22, 27
  - PRB: 25 (5MHZ), 50 (10MHZ), 100 (20MHZ)
  - gcc 4.7.3, x86-64,1000 frames, full rate, AWGN channel

- Not all the possible optimization are exploited
OAI BBU performance
Intel IvyBridge i5-3470 @ 3.2GHz

OAI BBU DL/UL vs MCS, Tx mode 1

Timing (us) vs MCS Index

- DL: PRB 25
- UL: PRB 25
- DL: PRB 50
- UL: PRB 50
- DL: PRB 100
- UL: PRB 100
OAI BBU performance
Intel Xeon E5-1607 v2@3GHz

OAI BBU DL/UL vs MCS, Tx mode 1

Timing (μs) vs MCS Index

- DL: PRB 25
- UL: PRB 25
- DL: PRB 50
- UL: PRB 50
- DL: PRB 100
- UL: PRB 100
### OAI BBU Performance

**Intel Xeon E5-1607 v2@3GHz**

<table>
<thead>
<tr>
<th>eNB Rx stats (1 subframe)</th>
<th>eNB Tx stats (1 subframe)</th>
</tr>
</thead>
<tbody>
<tr>
<td>OFDM demod: 109.695927 us</td>
<td>OFDM mod: 108.308182 us</td>
</tr>
<tr>
<td>ULSCH demod: 198.603526 us</td>
<td>DLSCH mod: 176.487999 us</td>
</tr>
<tr>
<td>ULSCH Decoding: 624.602407 us</td>
<td>DLSCH scrambling: 123.744984 us</td>
</tr>
<tr>
<td></td>
<td>DLSCH encoding: 323.395231 us</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- 931 us (<1 core)
- 730 us (<1core)

- With AVX2 (256-bit SIMD), turbo decoding and FFT processing will be exactly twice as fast
  - 1 core per eNB

**Configuration**

- gcc 4.7.3, x86-64 (3 GHz Xeon E5-2690)
- 20 MHz bandwidth (UL mcs16 – 16QAM, DL mcs 24 – 64QAM, transmission mode 1 - SISO)
- 1000 frames, AWGN channel
OAI BBU performance

Intel Atom C2758@2.4GHz

OAI BBU DL/UL vs MCS, Tx mode 1

Timing (us)

MCS Index

DL: PRB 25
UL: PRB 25
DL: PRB 50
UL: PRB 50
DL: PRB 100
UL: PRB 100
Discussion

- With the possible/potential optimizations
  - a full software solution would fit with an average of 1 x86 core per eNB instance
  - Mindful about the energy and realtime operation (potential existence of missing slots)

- using an external HW accelerators
  - Reduce the soft modem computational complexity down to 0.4 cores per eNB and improve energy efficiency
  - Mindful about the bus utilization from the inner-modem to the hardware and back for further processing
**Discussion**

**BBU scales with average MCS: load balancing**

- **For a RRH**
  - $MCS_i < MCS_{max}$

- **For a BBU**
  - $\overline{MCS} \ll N \times MCS_{max}$

- **Num CPU scales with**
  - $\text{num CPU} \approx \overline{MCS}$
Parallelization of OAI BBU

efficient use of available CPU cores and process prioritization

Digital Baseband Inputs

Prefix Extraction → FFT → Channel Compensation

To timing (DAQ) and frequency correction units, and I/Q imbalance

Pilot extraction

Channel Estimation

Broadband Channel Estimates

Lower-priority thread (scheduled every .5 ms)

8 lowest-priority threads (scheduled every 1 ms if needed)

Slot 2n

CPU 3

CPU 2

CPU 1

CPU 0

TX (2n+2), RX (2n-1)

RX (2n)

RX (2n)

RX (2n)

RX (2n+2)

RX (2n+1)

RX (2n+2)

RX (2n+1)

RX (2n+2)

RX (2n+1)

RX (2n+2)

RX (2n+1)

RX (2n+2)

RX (2n+1)

RX (2n+2)

RX (2n+1)

RX (2n+2)

RX (2n+1)

RX (2n+2)

RX (2n+1)

RX (2n+2)

RX (2n+1)

RX (2n+2)

RX (2n+1)

RX (2n+2)

RX (2n+1)

RX (2n+2)

RX (2n+1)

RX (2n+2)

RX (2n+1)

RX (2n+2)

RX (2n+1)

RX (2n+2)

RX (2n+1)

RX (2n+2)

RX (2n+1)

RX (2n+2)

RX (2n+1)

RX (2n+2)

RX (2n+1)

RX (2n+2)

RX (2n+1)

RX (2n+2)

RX (2n+1)

RX (2n+2)

RX (2n+1)

RX (2n+2)

RX (2n+1)

RX (2n+2)

RX (2n+1)

RX (2n+2)

RX (2n+1)

RX (2n+2)

RX (2n+1)

RX (2n+2)

RX (2n+1)

RX (2n+2)

RX (2n+1)

RX (2n+2)

RX (2n+1)

RX (2n+2)

RX (2n+1)

RX (2n+2)

RX (2n+1)

RX (2n+2)

RX (2n+1)

RX (2n+2)

RX (2n+1)

RX (2n+2)

RX (2n+1)

RX (2n+2)

RX (2n+1)

RX (2n+2)

RX (2n+1)

RX (2n+2)

RX (2n+1)

RX (2n+2)

RX (2n+1)

RX (2n+2)

RX (2n+1)

RX (2n+2)

RX (2n+1)

RX (2n+2)

RX (2n+1)

RX (2n+2)

RX (2n+1)

RX (2n+2)

RX (2n+1)

RX (2n+2)

RX (2n+1)

RX (2n+2)

RX (2n+1)

RX (2n+2)

RX (2n+1)

RX (2n+2)

RX (2n+1)

RX (2n+2)

RX (2n+1)

RX (2n+2)

RX (2n+1)

RX (2n+2)

RX (2n+1)

RX (2n+2)

RX (2n+1)

RX (2n+2)

RX (2n+1)

RX (2n+2)

RX (2n+1)

RX (2n+2)

RX (2n+1)

RX (2n+2)

RX (2n+1)

RX (2n+2)

RX (2n+1)

RX (2n+2)

RX (2n+1)

RX (2n+2)

RX (2n+1)

RX (2n+2)

RX (2n+1)

RX (2n+2)

RX (2n+1)

RX (2n+2)

RX (2n+1)

RX (2n+2)

RX (2n+1)

RX (2n+2)

RX (2n+1)

RX (2n+2)

RX (2n+1)

RX (2n+2)

RX (2n+1)

RX (2n+2)

RX (2n+1)

RX (2n+2)

RX (2n+1)

RX (2n+2)

RX (2n+1)

RX (2n+2)

RX (2n+1)

RX (2n+2)

RX (2n+1)

RX (2n+2)

RX (2n+1)

RX (2n+2)

RX (2n+1)

RX (2n+2)

RX (2n+1)

RX (2n+2)

RX (2n+1)

RX (2n+2)

RX (2n+1)

RX (2n+2)

RX (2n+1)

RX (2n+2)

RX (2n+1)

RX (2n+2)

RX (2n+1)

RX (2n+2)

RX (2n+1)

RX (2n+2)

RX (2n+1)

RX (2n+2)

RX (2n+1)

RX (2n+2)

RX (2n+1)

RX (2n+2)

RX (2n+1)

RX (2n+2)

RX (2n+1)

RX (2n+2)

RX (2n+1)

RX (2n+2)

RX (2n+1)

RX (2n+2)

RX (2n+1)

RX (2n+2)

RX (2n+1)

RX (2n+2)

RX (2n+1)

RX (2n+2)

RX (2n+1)

RX (2n+2)

RX (2n+1)

RX (2n+2)

RX (2n+1)

RX (2n+2)

RX (2n+1)

RX (2n+2)

RX (2n+1)

RX (2n+2)

RX (2n+1)

RX (2n+2)

RX (2n+1)

RX (2n+2)

RX (2n+1)

RX (2n+2)

RX (2n+1)

RX (2n+2)

RX (2n+1)

RX (2n+2)

RX (2n+1)

RX (2n+2)

RX (2n+1)

RX (2n+2)

RX (2n+1)

RX (2n+2)

RX (2n+1)

RX (2n+2)

RX (2n+1)

RX (2n+2)

RX (2n+1)

RX (2n+2)

RX (2n+1)

RX (2n+2)

RX (2n+1)

RX (2n+2)

RX (2n+1)

RX (2n+2)

RX (2n+1)

RX (2n+2)

RX (2n+1)

RX (2n+2)

RX (2n+1)

RX (2n+2)

RX (2n+1)

RX (2n+2)

RX (2n+1)

RX (2n+2)

RX (2n+1)

RX (2n+2)

RX (2n+1)

RX (2n+2)

RX (2n+1)

RX (2n+2)

RX (2n+1)

RX (2n+2)

RX (2n+1)

RX (2n+2)

RX (2n+1)

RX (2n+2)

RX (2n+1)

RX (2n+2)

RX (2n+1)

RX (2n+2)

RX (2n+1)

RX (2n+2)

RX (2n+1)

RX (2n+2)

RX (2n+1)

RX (2n+2)

RX (2n+1)

RX (2n+2)

RX (2n+1)

RX (2n+2)

RX (2n+1)

RX (2n+2)

RX (2n+1)
From ExpressMIMO2 to RRH+CPRI Integration

- Today, ExpressMIMO2 boards support peak throughput of 2Gbps (Theoretical PCI Express throughput: 2.5Gbps)
  - 1eNB 20MHz 1Tx/1Rx (SISO) ≡ 1Core
  - 1eNB 10MHz 2Tx/2Rx (MIMO) ≡ 1Core
  - 4eNB 5MHz 1Tx/1Rx (SISO) ≡ 1Core

- Connection of commercial RRH with OpenAirInterface (future)
Parallel use of ExpressMIMO2
various configurations are possible

16-way PCIe Backplane

16x 2.5Gbit/s = 40 Gbit/s peak

High-End Intel Xeon Computing Engine
3GHz Dual-Proc AVX2
20 Parallel Cores

OpenAir4G
RT-Linux
MODEMs
OAI ECOSYSTEM
**OAI use-cases and avenues**

Supported by our past and on-going projects

- **Standard and non-standard usage of 3GPP LTE systems**
  - Non-standard: IEEE 802.21, PMIP and DMM, Exit native IP at the BS

- **5G evolution path of OAI soft-modem supported by EU/industrial projects**
  - Cloudification of radio networks (RAN+EPC)
  - Massive MIMO, and COMP
  - Cognitive networking
  - Software-defined networking and network function virtualization support
  - Support of machine type communications
  - Mesh extension in support of multihop operation
  - Cooperative transmission and MAC
  - Caching strategy at the eNB or S/P-GW
  - Cooperative eMBMS, proximity networking
  - Scalable system experimentation and evaluation
  - RRM policies, handover logic and performance, MIMO performance, traffic scheduling policy
Collaborative Web Tools

- **OpenAirInterface SVN Repositories**
  - All development is available through [www.openairinterface.org](http://www.openairinterface.org)’s SVN repository (openair4G) containing:
    - OPENAIR0 (open-source real-time HW/SW)
    - OPENAIR1 (open-source real-time and offline SW)
    - OPENAIR2 (open-source real-time and offline SW)
    - OPENAIR3 (open-source Linux SW suite for cellular and MESH networks)
    - TARGETS: different top-level target designs (emulator, RTAI, etc.)
  - Partners can access and contribute to our development
    - [https://svn.eurecom.fr/openair4G](https://svn.eurecom.fr/openair4G) (RO access)

- **OpenAirInterface TWIKI**
  - A TWIKI site for quick access by partners to our development via a collaborative HOW-TO

- **Forum and bugzilla**
  - External support services (not currently used effectively)

- **OAI VM image (start from openair4G/targets/README.txt)**
  - [https://emu.openairinterface.org/openairlab/openairlab.zip](https://emu.openairinterface.org/openairlab/openairlab.zip)
Contacts Information

- **URL:**
  - [www.openairinterface.org](http://www.openairinterface.org)
  - [https://twiki.eurecom.fr/twiki/bin/view/OpenAirInterface](https://twiki.eurecom.fr/twiki/bin/view/OpenAirInterface)

- **Partnership and collaboration:**
  - openair_admin@eurecom.fr

- **Technical Support:**
  - openair_tech@eurecom.fr

- **Developer support:**
  - openair4G-devel@eurecom.fr
  - To subscribe, send an email to majordomo@eurecom.fr with the content "subscribe openair4G-devel"
EURECOM MEMBERS