Mobile Communications Department

Eurecom

(Collaboration with SoC Laboratory, Telecom ParisTech Sophia)
Overview

- Introduction to www.openairinterface.org
- Current PHY/MAC Development
- OpenAirInterface Simulation/Emulation Methodologies
www.openairinterface.org

- Provides open-source (hardware and software) wireless technology platforms
  - target innovation in air-interface technologies through experimentation

- We rely on the help of
  - Publicly-funded research initiatives (ANR, ICT, CELTIC)
  - Direct contracts with industrial partners
  - Widespread collaboration with a network of partners using open-source development and tools
    - LINUX/RTAI based SW development for PCs
    - LEON3/GRLIB-based HW and eCos-based SW development for FPGA targets
    - LINUX networking environment
  - Experimental Licenses from ARCEP (French Regulator) for medium-power outdoor network deployments
    - 1.9 GHz TDD, 5 MHz channel bandwidth
    - 2.6 GHz FDD (two channels), 20 MHz channel bandwidth
Principal Subject Areas

- **Real-time Radio Signal Processing**
  - Hardware/software architectures in support of real-time signal processing (Software Radio, multi-processor system-on-chip)
  - Algorithmic optimizations at the PHY layer (UMTS-LTE and 802.16m technologies)
  - PHY-layer support for cellular and mesh Network topologies

- **All-IP Wireless Networking**
  - All-IP Cellular mobile network protocols (IPv6 basestation routers, IPv6 mobility management)
  - 802.21
  - IP/MPLS Protocols adapted to MESH topologies
  - Layer 2 Protocols (MAC scheduling, Radio Resource Control, Radio Link Control) for cellular and mesh network topologies

- **Agile RF System Design**
  - Wideband radio design, linear wide-dynamic range receivers
  - "Intelligent" RF (RF/DSP co-design)

- **Design and Simulation Methodologies**
  - Efficient simulation methods (performance, functional and behavioral)
  - Abstraction techniques (hardware modelling, PHY sub-system modelling, traffic modelling, etc.)
  - RF emulation architectures for distributed real-time simulation of wireless networks

- **Propagation and System Measurements and their Analysis (eMOS)**
  - Wideband channel characterization and modelling
  - Real-time measurement collection and offline empirical performance analysis

- **Cognitive Radio**
  - Development of innovative techniques based on sensor networks, that will support the coexistence of licensed and unlicensed wireless users in a same area
  - Design, dimensioning and internetworking of cognitive networks
Collaborative Web Tools

- **OpenAirInterface SVN Repositories**
  - All development is available through [www.openairinterface.org](http://www.openairinterface.org)’s SVN repository 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)
  - Partners can access and contribute to our development

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

- **Forum**
  - external support services (not currently used)
OpenAirInterface Development Areas

- **OPENAIR0: Wireless Embedded System Design**
  - Agile RF design, Reconfigurable High-end Transceiver Architectures, FPGA prototyping, Simulation Methodologies, Software development tools, low-power chip design

- **OPENAIR1: Baseband/PHY**
  - Advanced PHY (LTE/802.16x), Propagation Measurement and Modelling, Sensing and Localization Techniques, PHY Modeling Tools

- **OPENAIR2: Medium-Access Protocols**
  - Cellular topologies, single-frequency resource allocation, cross-layer wideband scheduling, Mesh topologies, distributed resource control

- **OPENAIR3: Wireless Networking**
  - All-IP, Mobility Management, 802.21, Cellular/Mesh Routing Protocols, Mesh Topology Management, Multimodal Radio Resource Management

**Cognitive Technologies**
Current OpenAirInterface Network Topologies

Cellular Topology

Mesh Topology

Mobile Ad-Hoc Network
"The Moving Hot-Spot"

- Fast, deployable, Compatible
- Auto-est. network
- Recognize terminals

The MESA
Firefighters

Backhaul Satcom Link

Telemedical Assistance

Airborne Control

Cluster-Head
Mesh Router
Isolated Node
Edge Router
Other Access Technology

Cluster-Head/NodeB
User Equipment (UE)
Software Roadmap

2003-2011

**WIRELESS3G4FREE**

- **OpenAirInterface (WIDENS/CHORIST)**
  - LTE
  - LTE-DL compliant waveform
  - Mesh extensions from WIDENS/CHORIST to be integrated
  - Integration with WIDENS/CHORIST Protocol Stack (openair2) underway

**AdHoc/Mesh and Cellular Topologies**
- In-house MIMO-OFDMA TDD waveform (WiMAX 2004 like)
- Distributed Signal Processing and Mesh-Topology functions (L2.5 relaying)

**TD-SCDMA SDR**
- IPv6 interconnect
- No longer supported
CardBus MIMO 1

- Current platform for application experimentation and test network deployments
  - 5 MHz channel bandwidth
  - TDD@1900 MHz
  - PCMCIA-CardBus form-factor
  - 2x2 MIMO-OFDMA, LTE-IDL waveform
  - Two-way communications
  - Full Software Radio under RTAI/Linux on x86 architectures

- Cellular Deployments
- MESH deployments
- EMOS Channel sounding
- 32 (still counting) CBMIMO1 – V2 cards fabricated, most for partner labs (Thales, Aalborg, UNICE, TUB, Bilkent)
- 23 dBm output power (<1km range in LOS)
- MAC/PHY are RT threads under RTAI (www.rtai.org)
- They are synchronized (with respect to sample stream) with DMAs to/from DAQ subsystem on PCI
- L3 networking makes use of Linux networking stack
RTAI vs user-space

- RTAI is used in an OpenAirInterface PC environment to
  - Provide hard real-time support of CBMIMO1 resources
    * Scheduling accuracy on order of tens of microseconds
    * Enough for LTE sub-frame (500us reaction time)
  - Provide easy interface to Linux networking sub-system and kernel functions
  - DSP is performed in kernel using RTAI scheduling mechanisms (highly-optimized SIMD code!)

- User-space approaches under Linux not for real-time two communications
  - Can work for with OpenAirInterface for offline testing though with user-space C/C++ programs or with OCTAVE
CBMIMO1 Embedded System

- CardBus MIMO I is a front-end for a “pure” software radio

Diagram:

- AMBA BUS
- DP-RAM
- LEON3 CPU
- DMA CNTRL
- INTR CNTRL
- DAQ/DSP Unit
- AD9862 (x2) + Switches
- Config EEPROM
- DCMs
- GRPCI
- 26 MHz TCXO
- JTAG
- JTAG CONN
- RF CNTRL + Expansion
- SDRAM
- PCI Bus
- Standard x86-based PC
Current CBMIMO1 V2 Design

- **CBMIMO1 provides**
  - A Leon3-based embedded processing engine on a Xilinx XC2V3000 FPGA
    - 52 MHz processor speed
    - 64 kByte embedded memory (16 Mbyte SDRAM not used currently)
    - DMA engine
    - PCI/CardBus bridge with burst transfers
  - AD9862 acquisition engine
    - LTE TDD/FDD framing (7.68 Ms/s)
    - LTE 5 MHz baseband filtering (TX)
    - LTE FFT (TX, 512-point) and cyclic-prefix processing (TX/RX)
  - RF control (gains, frequencies, timing of RF)

- **CBMIMO1 allows for 2x2 MIMO operation in either FDD (with external RF) or TDD**
  - Embedded software handles LTE framing and transfers of signals to/from PC memory along with synchronization events for RTAI scheduling
  - PC configures CBMIMO1 with memory regions for signals and frame parameters on init and card does the rest
  - Special frame resynchronization for two-way operation is provided (timing drift adjustments)
Updates to Leon3 Firmware

- **Cards are delivered with the most recent firmware**
  - since we cannot support the old designs (require hard-to-find PCI-native laptops)
  - new LTE software is for new design and new PCs only
  - Calibration and testing has been done and this doesn’t (hopefully) have to be done again

- **Consequences**
  - FPGA bitstream still may be buggy (likely)
  - Leon3 Software will still be updated over coming months

- **Requirements**
  - Xilinx Parallel IV programming cable to update your cards with newer revisions of firmware [openair0/cbmimo1/.... ]
  - Xilinx ISE Software 10.x or 9.x (programming may work with 11.x, to be verified)

- **Otherwise**
  - You have to send the cards back to us (suboptimal solution)
Software Modules

- **openair_rf.ko** [openair1/ARCH/CBMIMO1/DEVICE_DRIVER/]
  - `/dev/openair0` character device (open, close, ioctl, mmap, etc.) providing user-space control over CBMIMO1 card
  - Makefile for compiling/linking
    - OpenAirInterface PHY (either WIDENS/CHORIST or LTE), LTE is current development and starting point for new designs (hardware configured for this) [PHY/]
    - RTAI thread creation for real-time DSP [SCHED/]
  - Can be compiled as a channel sounder too (EMOS) with limited data transmission capacity but enhanced real-time storage capacity for offline analysis
Software Modules (cont’d)

- **openair_l2.ko [openair2/LAYER2]**
  - Layer 2 protocol stack – CHORIST MAC, 3GPP Rel. 5 RLC, PDCP
  - Currently not integrated with new LTE modem (help …)

- **openair_rrc.ko [openair2/RRC ]**
  - Two possible RRC layers (mesh variant is integrated with LAYER2 stack)

- **nasmesh.ko [openair2/NAS/Driver/MESH]**
  - Linux networking device, compatible to 2.6.31
  - Allows for mapping of IP-based traffic (MPLS support will die because of lack of kernel support beyond 2.6.20) onto OpenAirInterface radio-bearers (PDCP interface)
Linux Devices

- /dev/openair0
  - Character device driver for user-space control of CBMIMO1 Card found in openair1 repository
    - Calibration via OCTAVE
    - Card initialization and configuration
    - Mode selection (eNb, UE)
    - Manual control of RF (gains, frequency offset, timing advance, etc.)
    - Signal acquisition (OCTAVE)
    - Test Signal generation (sinusoid, OFDM waveform, etc.)

- Nasmesh
  - Linux networking device, found in openair2 repository
PHY/MAC Characteristics
(OpenAirInterface - LTE)
Purpose

- Develop an open-source baseband implementation of a subset of LTE Release-8 on top of OpenAirInterface.org SW architecture and HW demonstrators

Goals
- Representative of LTE user-plane
  - Full compliance of LTE frame (normal and extended prefix)
  - Full Downlink shared channel compliance
  - Full compliance of DL pilot structure
  - Support for a subset of transmission modes (2x2 operation)
    - Modes 1,2,4,5,6 (Mode 3 to be studied for inclusion)
  - Support for up to 3 sectors in eNb
- Useful for measurement campaigns
- Useful as starting point for research-oriented extensions (to justifiably claim potential impact on LTE-A)
- Provide realistic (and rapid) LTE simulation environment for PHY/MAC
Framing

- LTE is specified for any bandwidth between 1.08 MHz and 19.8 MHz which is a multiple of 180 kHz
- The “common” sizes will be
  - 1.08 MHz transmission bandwidth with 1.25 MHz spacing
  - 2.7 MHz transmission bandwidth with 3 MHz spacing
  - 4.5 MHz transmission bandwidth with 5 MHz spacing (interest for OpenAirInterface.org on CardBus MIMO 1)
  - 9 MHz transmission bandwidth with 10 MHz channel spacing (interest for OpenAirInterface.org on ExpressMIMO)
  - 13.5 MHz transmission bandwidth with 15 MHz spacing
  - 18 MHz transmission bandwidth with 20 MHz channel spacing (interest for OpenAirInterface.org on ExpressMIMO)
**Resource blocks**

- LTE defines the notion of a **resource block** which represents the minimal scheduling resource for both uplink and downlink transmissions.

- A **physical resource block (PRB)** corresponds to 180 kHz of spectrum.
### Common PRB Formats

<table>
<thead>
<tr>
<th>Channel Bandwidth (MHz)</th>
<th>$N_{RB}^{DL}/N_{RB}^{UL}$</th>
<th>Typical IDFT size</th>
<th>Number of Non-Zero Sub-carriers (REs)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.25</td>
<td>6</td>
<td>128</td>
<td>72</td>
</tr>
<tr>
<td>5</td>
<td>25</td>
<td>512</td>
<td>300</td>
</tr>
<tr>
<td>10</td>
<td>50</td>
<td>1024</td>
<td>600</td>
</tr>
<tr>
<td>15</td>
<td>75</td>
<td>1024 or 2048</td>
<td>900</td>
</tr>
<tr>
<td>20</td>
<td>100</td>
<td>2048</td>
<td>1200</td>
</tr>
</tbody>
</table>

- PRBs are mapped onto contiguous OFDMA/SC-FDMA symbols in the time-domain (6 or 7)
- Each PRB is chosen to be equivalent to 12 (15 kHz spacing) sub-carriers of an OFDMA symbol in the frequency-domain
  - A 7.5kHz spacing version exists with 24 carriers per sub
- Because of a common PRB size over different channel bandwidths, the system scales naturally over different bandwidths
  - UEs with different bandwidth constraints can still be served by an eNb with a wider channel bandwidth
**OFDMA/SC-FDMA Mapping**

- OFDMA/SC-FDMA Sub-carriers are termed “Resource Elements” (RE)
- DC carrier and high-frequencies are nulled
  - Spectral shaping and DC rejection for Zero-IF receivers
  - Half the bandwidth loss w.r.t. WCDMA (22%)

<table>
<thead>
<tr>
<th>Channel Bandwidth (MHz)</th>
<th>( N_{RB}^{DL}/N_{RB}^{UL} )</th>
<th>Bandwidth Expansion</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.25</td>
<td>6</td>
<td>8%</td>
</tr>
<tr>
<td>5</td>
<td>25</td>
<td>11%</td>
</tr>
<tr>
<td>10</td>
<td>50</td>
<td>11%</td>
</tr>
<tr>
<td>15</td>
<td>75</td>
<td>11%</td>
</tr>
<tr>
<td>20</td>
<td>100</td>
<td>11%</td>
</tr>
</tbody>
</table>
Example: 300 REs, 25 RBs (5 MHz channel)

Current HW configuration

“Normal” Cyclic Prefix Mode
(7 symbols)

“Extended” Cyclic Prefix Mode
(6 symbols)
One frame = $T_f = 307200 \, T_s = 10\, \text{ms}$

$T_{\text{slot}} = 15360 \, T_s = 500\, \mu s$

One subframe

Frequency Domain View

Time-domain View

Normal Prefix

Extended Prefix

83\, \mu s

13.9\, \mu s

4.69\, \mu s

5.2\, \mu s

71.3\, \mu s

71.9\, \mu s
Downlink Physical Channels

- **Physical Downlink Shared Channel (PDSCH)**
  - Carries user and control plane traffic. It is dynamically scheduled every sub-frame by the eNb

- **Physical Broadcast Channel (PBCH)**
  - Carries a minimal amount of cell-specific control plane traffic (1 kbit/s)
  - Implemented in OpenAirInterface.org (with Turbo-code instead of rate 1/3 tail-biting code)

- **Physical Multicast Channel (PMCH)**
  - Carries cell-specific user-plane broadcast/multicast traffic (e-MBMS)
  - Not implemented

- **Physical Control Format Indicator Channel (PCFICH)**
  - Carries cell-specific control format information. This channel indicates to the UE the control format (number of symbols comprising PDCCH, PHICH) of the current subframe.
  - Not implemented currently

- **Physical Downlink Control Channel (PDCCH)**
  - Carries user and cell-specific control information (scheduling, resource assignments, etc). This collection of channels provides the UE with the DLSCH assignments in the sub-frame
  - Implemented in OpenAirInterface.org (with turbo code instead of rate 1/3 tail-biting code)

- **Physical Hybrid ARQ Indicator Channel (PHICH)**
  - Carries user-specific control information for HARQ (ACK/NACK)
  - Implemented
Physical Signals

- **Reference Signals**
  - Cell-specific reference signals: Used for channel estimation and frequency-offset estimation in UE
  - User-specific reference signals: Used for channel estimation for a specific user receiving a tailored signal

- **Synchronization Signals**
  - Primary synchronization signal: Used for timing acquisition at UE
  - Secondary synchronization signal: Used for basic framing acquisition at UE allowing demodulation of physical channels
Antenna Ports

- **LTE supports up to 4 physical antennas**
  - Single-antenna configuration, \( p=\{0\} \)
  - Dual-antenna configuration, \( p=\{0,1\} \)
  - Quad-antenna configuration, \( p=\{0,1,2,3\} \) – currently not supported in OpenAirInterface.org (for ExpressMIMO and ExpMIMO-lite later …)

- A fifth “virtual” antenna, \( p=\{4\} \) is included to indicate MBSFN (MBMS) reference and physical channels
  - This is not supported in OpenAirInterface.org

- A sixth “virtual” antenna, \( p=\{5\} \), is included to indicate user-specific antenna processing (e.g. beamforming) with an unspecified number of physical antennas
  - This is not supported in OpenAirInterface.org
Virtual Resource Blocks

- PRBs can be permuted into virtual RBs (VRBs) for the purpose of increasing diversity against fading and/or inter-cell interference
  - Currently not supported in OpenAirInterface.org (but reasonably simple if someone really wants to help)
Resource Element Groups (REG)

- Resource Element Groups (REG) are used to describe the portions of PRBs which are used for control signals.

Two Cell-Specific Reference Symbols

Four Cell-Specific Reference Symbols

Cell-Specific Ref. Signal
General Transmission Chain

One or two codewords

Bitwise scrambling

QPSK, 16QAM
64 QAM (possibly different on each stream)

Still to be included for DLSCH!

$$y(i) = \left[ \begin{array}{c} y^{(0)}(i) \\ \vdots \\ y^{(P-1)}(i) \end{array} \right]$$

$$x(i) = \left[ \begin{array}{c} x^{(0)}(i) \\ x^{(1)}(i) \\ x^{(2)}(i) \\ x^{(3)}(i) \end{array} \right]$$
# Layer Mapping (Spatial Multiplexing)

<table>
<thead>
<tr>
<th>Number of layers</th>
<th>Number of code words</th>
<th>Codeword-to-layer mapping</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>$x^{(i)}(i) = d^{(i)}(i)$</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>$M_{\text{symb}}^{\text{layer}} = M_{\text{symb}}^{(0)}$</td>
</tr>
<tr>
<td></td>
<td></td>
<td>$x^{(1)}(i) = d^{(1)}(i)$</td>
</tr>
<tr>
<td></td>
<td></td>
<td>$M_{\text{symb}}^{\text{layer}} = M_{\text{symb}}^{(1)}$</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>$x^{(0)}(i) = d^{(0)}(2i)$</td>
</tr>
<tr>
<td></td>
<td></td>
<td>$x^{(1)}(i) = d^{(0)}(2i+1)$</td>
</tr>
<tr>
<td>3</td>
<td>2</td>
<td>$x^{(0)}(i) = d^{(0)}(i)$</td>
</tr>
<tr>
<td></td>
<td></td>
<td>$x^{(1)}(i) = d^{(1)}(2i)$</td>
</tr>
<tr>
<td></td>
<td></td>
<td>$x^{(2)}(i) = d^{(1)}(2i+1)$</td>
</tr>
<tr>
<td></td>
<td></td>
<td>$M_{\text{symb}}^{\text{layer}} = M_{\text{symb}}^{(0)} = M_{\text{symb}}^{(1)} / 2$</td>
</tr>
<tr>
<td>4</td>
<td>2</td>
<td>$x^{(0)}(i) = d^{(0)}(2i)$</td>
</tr>
<tr>
<td></td>
<td></td>
<td>$x^{(1)}(i) = d^{(0)}(2i+1)$</td>
</tr>
<tr>
<td></td>
<td></td>
<td>$x^{(2)}(i) = d^{(1)}(2i)$</td>
</tr>
<tr>
<td></td>
<td></td>
<td>$x^{(3)}(i) = d^{(1)}(2i+1)$</td>
</tr>
<tr>
<td></td>
<td></td>
<td>$M_{\text{symb}}^{\text{layer}} = M_{\text{symb}}^{(0)} / 2 = M_{\text{symb}}^{(1)} / 2$</td>
</tr>
</tbody>
</table>

Not implemented
Layer Mapping (TX Diversity)

<table>
<thead>
<tr>
<th>Number of layers</th>
<th>Number of code words</th>
<th>Codeword-to-layer mapping</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>1</td>
<td>$x^{(0)}(i) = d^{(0)}(2i)$, $x^{(1)}(i) = d^{(0)}(2i + 1)$, $M_{\text{symb}}^{\text{layer}} = M_{\text{symb}}^{(0)}/2$</td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td>$x^{(0)}(i) = d^{(0)}(4i)$, $x^{(1)}(i) = d^{(0)}(4i + 1)$, $x^{(2)}(i) = d^{(0)}(4i + 2)$, $x^{(3)}(i) = d^{(0)}(4i + 3)$, $M_{\text{symb}}^{\text{layer}} = \begin{cases} M_{\text{symb}}^{(0)}/4 &amp; \text{if } M_{\text{symb}}^{(0)} \mod 4 = 0 \ (M_{\text{symb}}^{(0)}+2)/4 &amp; \text{if } M_{\text{symb}}^{(0)} \mod 4 \neq 0 \end{cases}$</td>
</tr>
</tbody>
</table>

Second format not implemented
Precoding serves two purposes

- TX diversity through either Space-Frequency Block Coding (Alamouti, Double Alamouti) or Single-stream Delay Diversity
- Rank increase for closed-loop techniques (dual-stream MIMO and MU-MIMO)

Spatial Multiplexing

Large Cyclic Delay-Diversity

Not implemented for now
Spatial Multiplexing / MU-MIMO

Selection of precoding codewords by eNodeB based on feedback

\[ s(n) = \begin{bmatrix} s^{(0)}(n) \\ \vdots \\ s^{(P-1)}(n) \end{bmatrix} = y^{(0)}(n)p^{(0)} + y^{(1)}(n)p^{(1)} \]

Under Implementation (QPSK/QPSK, QPSK/16QAM, 16QAM/16QAM)

<table>
<thead>
<tr>
<th>Codebook index</th>
<th>Number of layers $v$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>( \begin{bmatrix} 1 \ \sqrt{2} \end{bmatrix} )</td>
</tr>
<tr>
<td>1</td>
<td>( \begin{bmatrix} 1 \ \sqrt{2} \end{bmatrix} )</td>
</tr>
<tr>
<td>2</td>
<td>( \begin{bmatrix} 1 \ \sqrt{2}j \end{bmatrix} )</td>
</tr>
<tr>
<td>3</td>
<td>( \begin{bmatrix} 1 \ -j \end{bmatrix} )</td>
</tr>
</tbody>
</table>
Transmit Diversity (2 Antennas)

\[
\begin{bmatrix}
    y^{(0)}(2i) \\
y^{(1)}(2i) \\
y^{(0)}(2i+1) \\
y^{(1)}(2i+1)
\end{bmatrix}
= \frac{1}{\sqrt{2}}
\begin{bmatrix}
    1 & 0 & j & 0 \\
0 & -1 & 0 & j \\
0 & 1 & 0 & j \\
1 & 0 & -j & 0
\end{bmatrix}
\begin{bmatrix}
    \text{Re}(x^{(0)}(i)) \\
    \text{Re}(x^{(1)}(i)) \\
    \text{Im}(x^{(0)}(i)) \\
    \text{Im}(x^{(1)}(i))
\end{bmatrix}
\]

\[
Y(i) = \begin{bmatrix}
    y^{(0)}(2i) & y^{(0)}(2i+1) \\
y^{(1)}(2i) & y^{(1)}(2i+1)
\end{bmatrix}
= \frac{1}{\sqrt{2}}
\begin{bmatrix}
    x^{(0)}(i) & x^{(1)*}(i) \\
-x^{(1)*}(i) & x^{(0)*}(i)
\end{bmatrix}
\]

\[
r(i) = h(i)^t Y(i) + z(i)
= \left[h_1(2i)y^{(0)}(2i) + h_2(2i)y^{(1)}(2i) \right.
\left. + h_1(2i+1)y^{(0)}(2i+1) + h_2(2i+1)y^{(1)}(2i+1) \right] + z(i)
= \frac{1}{\sqrt{2}}
\left[h_1(2i)x^{(0)}(i) - h_2(2i)x^{(1)*}(i) \right.
\left. + h_1(2i+1)x^{(1)*}(i) + h_2(2i+1)x^{(0)*}(i) \right] + z(i)
\]

Implemented (QPSK, 16QAM, 64QAM)
$$\hat{x}^{(0)}(i) = \sqrt{2}\left( h_1^* (2i) r(2i) + h_2 (2i + 1) r^* (2i + 1) \right)$$
$$= \left( \| h_1 (2i) \|^2 + \| h_2 (2i + 1) \|^2 \right) x^{(0)}(i) + \left( h_2 (2i + 1) h_1^* (2i + 1) - h_1^* (2i) h_2 (2i) \right) x^{(1)*}(i)$$
$$= 0, \text{if } h_k (2i) = e^{j\phi} h_k (2i+1), k=1,2$$

$$\hat{x}^{(1)}(i) = \sqrt{2}\left( - h_2 (2i) r^* (2i) + h_1^* (2i + 1) r(2i + 1) \right)$$
$$= \left( \| h_1 (2i + 1) \|^2 + \| h_2 (2i) \|^2 \right) x^{(1)}(i) + \left( - h_2 (2i) h_1^* (2i) + h_1^* (2i + 1) h_2 (2i + 1) \right) x^{(0)*}(i)$$
$$= 0, \text{if } h_k (2i) = e^{j\phi} h_k (2i+1), k=1,2$$

$$= \left( \| h_1 (2i + 1) \|^2 + \| h_2 (2i) \|^2 \right) x^{(1)}(i)$$

Implemented (QPSK, 16QAM, 64QAM)
PDCCH

- Carries scheduling assignments and other control information
  - UE decodes PDCCH every sub-frame to know which PDSCH to decode PUSCH/PUCCH/SRS/PRACH to transmit and where to find them in time/frequency
- Transmitted on one or several CCEs (Control Channel Elements) using QPSK modulation
  - 1 CCE = 9 REGs
- Assignments of CCEs are described in “PHY Procedures” lecture

<table>
<thead>
<tr>
<th>PDCCH format</th>
<th>Number of CCEs</th>
<th>Number of resource-element groups</th>
<th>Number of PDCCH bits</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>9</td>
<td>72</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>18</td>
<td>144</td>
</tr>
<tr>
<td>2</td>
<td>4</td>
<td>36</td>
<td>288</td>
</tr>
<tr>
<td>3</td>
<td>8</td>
<td>72</td>
<td>576</td>
</tr>
</tbody>
</table>
Currently OpenAirInterface supports PDCCH

- Still under integration but
  - Tail-biting C. Code and rate matching is implemented
  - Performance testing still to be performed
**PHICH**

- **PHICH are orthogonal sequences defined by**
  - Group and sequence id’s
  - Group identifies the set of REGs and the sequence is the orthogonal sequence used within the group. Up to 4 sequence superimpose on REs

- **BPSK modulation, 12 REs/sequence**

- **Precoding like PBCH (Alamouti)**

<table>
<thead>
<tr>
<th>Sequence index</th>
<th>Orthogonal sequence Normal cyclic prefix</th>
<th>Orthogonal sequence Extended cyclic prefix</th>
</tr>
</thead>
<tbody>
<tr>
<td>$n_{PHICH}$</td>
<td>$N_{SF}^{PHICH} = 4$</td>
<td>$N_{SF}^{PHICH} = 2$</td>
</tr>
<tr>
<td>0</td>
<td>$[+1 \ +1 \ +1 \ +1]$</td>
<td>$[+1 \ +1]$</td>
</tr>
<tr>
<td>1</td>
<td>$[+1 \ -1 \ +1 \ -1]$</td>
<td>$[+1 \ -1]$</td>
</tr>
<tr>
<td>2</td>
<td>$[+1 \ +1 \ -1 \ -1]$</td>
<td>$[+j \ +j]$</td>
</tr>
<tr>
<td>3</td>
<td>$[+1 \ -1 \ -1 \ +1]$</td>
<td>$[+j \ -j]$</td>
</tr>
<tr>
<td>4</td>
<td>$[+j \ +j \ +j \ +j]$</td>
<td>-</td>
</tr>
<tr>
<td>5</td>
<td>$[+j \ -j \ +j \ -j]$</td>
<td>-</td>
</tr>
<tr>
<td>6</td>
<td>$[+j \ +j \ -j \ -j]$</td>
<td>-</td>
</tr>
<tr>
<td>7</td>
<td>$[+j \ -j \ -j \ +j]$</td>
<td>-</td>
</tr>
</tbody>
</table>

For extended cyclic prefix,

$$
\begin{bmatrix}
  d^{(0)}(4i) & d^{(0)}(4i+1) & d^{(0)}(4i+2) & d^{(0)}(4i+3)
\end{bmatrix}^T = \begin{bmatrix}
  d(2i) & d(2i+1) & 0 & 0 \\
  0 & 0 & d(2i) & d(2i+1)
\end{bmatrix}^T \quad \text{if } n_{PHICH}^{\text{group}} \mod 2 = 0
$$

$$
\begin{bmatrix}
  d^{(0)}(4i) & d^{(0)}(4i+1) & d^{(0)}(4i+2) & d^{(0)}(4i+3)
\end{bmatrix}^T = \begin{bmatrix}
  0 & 0 & d(2i) & d(2i+1)
\end{bmatrix}^T \quad \text{if } n_{PHICH}^{\text{group}} \mod 2 = 1
$$
Cell-Specific Reference Signals

$p = \{0\}, p = \{0, 1\}$

$p = 0$

$p = 1$ (if active)

$p = \{0, 1, 2, 3\}$

$p = 0$

$p = 1$

$p = 2$

$p = 3$ (if active)

Not implemented for 4 Antennas
Cell-Specific Reference Signals

- **Pseudo-random QPSK OFDM symbols**
  - Based on generic LTE Gold sequence
  - Different sequence for different cell IDs
  - Different in each symbol of sub-frame
  - Different in each sub-frame, but periodic across frames (10ms)

- **Evenly spaced in subframe to allow for simple and efficient least-squares interpolation-based receivers**
  - Between REs in frequency-domain
  - Across symbols in time-domain
MIMO Channel Estimation in LTE (simple)

- Recall that receiver sees

\[ R_k^N = \text{DFT}(r_k^N) = H^N S_k^N + Z_k^N \]

- Must get channel estimate for channel compensation

\[ \hat{H}^N = H^N + H_c^N \]

\[ \hat{H}_1(16) = P^*(16)R(16) \]

\[ \hat{H}_0(13) = P^*(13)R(13) \]

\[ \hat{H}_1(10) = P^*(10)R(10) \]

\[ \hat{H}_0(7) = P^*(7)R(7) \]

\[ \hat{H}_1(5) = (5/6)\hat{H}_1(4) + (1/6)\hat{H}_1(10) \]

\[ \hat{H}_0(0) = (7/6)\hat{H}_0(1) - (1/6)\hat{H}_0(7) \]

NO pilots here
Channel Estimation in LTE (simple) – cont’d

- The previous steps allow for determining the frequency response (MIMO) on symbols where the pilots are located.
- For the remaining symbols, we perform time-interpolation across adjacent symbols with pilots.

\[
\hat{H}_i(3) = \frac{1}{4}\hat{H}_i(0) + \frac{3}{4}\hat{H}_i(4) \\
\hat{H}_i(2) = \frac{1}{2}\hat{H}_i(0) + \frac{1}{2}\hat{H}_i(4) \\
\hat{H}_i(1) = \frac{3}{4}\hat{H}_i(0) + \frac{1}{4}\hat{H}_i(4)
\]
Synchronization Signals

10 ms

Subframe 0 Subframe 1 Subframe 4 Subframe 5 Subframe 9

Primary (Y) and Secondary B)
Synchronization Signals
(first half)

PBCH

Primary(Y) and Secondary(B)
Synchronization Signals
(2nd half)
Synchronization Signals/ PBCH

- Primary Synchronization implemented
- Secondary Synchronization not implemented
- PBCH implemented with different channel code
  - Will be migrated to Tail-biting CC (like PDCCH) in very short term
DLSCH Coding and Modulation

- OpenAirInterface is compliant with 36-212 with respect to DLSCH
  - TB CRC
  - Packet segmentation + segment CRC
  - Channel Coding
  - Rate Matching
  - Code block concatenation

- This guarantees representability of results for user-plane data
- The DLSCH implementation supports up to 8 HARQ processes
DLSCH Modulation

- DLSCH Modulation is compliant with a subset of the LTE transmission modes (1-2 antennas, no cyclic delay-diversity for the moment)
Uplink Physical Channels

- Uplink in LTE uses SC-FDMA transmission
  - Scrambling (binary)
  - Modulation (QPSK, 16QAM, 64QAM)
  - Transform precoding (DFT)
  - Resource Mapping
  - OFDM transmission

- OpenAirInterface HW currently supports OFDMA uplink transmission
  - Difficulties with SC-FDMA
    - Implementation of real-time split-radix DFTs over all possible sizes
    - Currently, the OpenAirInterface MODEM sends a minimal amount of information (table lookups for QAM symbols) on TX, due to bandwidth limitations on the CardBus/PCIe interface. With SC-FDMA we have to send raw signals, which poses a bit of a problem on current laptop configurations.
Uplink Physical Channels (cont’d)

- **Uplink physical channels**
  - Physical uplink shared channel (PUSCH): Uplink data and/or feedback information for eNb scheduler
    - Under implementation
  - Physical uplink control channel (PUCCH): Feedback information for eNb scheduler: *not implemented* (use PUSCH for feedback)
  - Physical random access channel (PRACH): Preambles for association, *not implemented*

- **Signals**
  - Reference signal: For UL channel estimation in resources corresponding to PUSCH and PUCCH
    - Under implementation
  - Sounding Reference signal (SRS)
    - Under implementation (old MRBCH from OpenAirInterface-MESH)
**OpenAirInterface PHY/MAC Roadmap**

- **Current development (OpenAirInterface-MESH -> OpenAirLTE)**
  - Nx2 MIMO-OFDMA for Single-Frequency deployment
    - BICM-SIC oriented for dual-stream coded-modulation QPSK->64 QAM (6 bits/s/Hz)
      - 64QAM not running on HW today (simulation yes)
      - Dual-stream limited to QPSK (16QAM in next release)
    - 802.11/16 convolutional or 3GPP turbo code
  - LTE-like MAC for cellular and mesh topologies
    - Shared channels (DL-SCH,UL-SCH) with scheduled MAC
    - L1/L2 Broadcast/Multicast support
    - Feedback signaling for spatial-subband-CSIT
    - HARQ type I/II (next release)
    - 3GPP RLC
    - Linux IP/MPLS networking device
    - Mesh enabled RRC
  - Distributed network synchronization
  - Distributed MU-MIMO - pilot definitions and coding sub-system
    - Dual-antenna receiver for intercell (or intracell in mesh) interference cancellation and distributed MIMO (one stream from each cell)
**PHY/MAC Roadmap**

- **Integration Efforts**
  - Immediate priority is to port existing developments to OpenAirLTE
    - Distributed MU-MIMO done
    - L2 integration in 02/2010
  - Replicate CHORIST mesh topology using OpenAirLTE ASAP

- **Planned New developments**
  - Improvements to software simulator for validation/debug (later)
  - Extensions to LTE PHY for Distributed MIMO via relays – SENDORA WP5/WP7 (2010)
  - Extensions to LTE PHY Distributed MIMO with HARQ via relays LOLA WP4/WP5 with Thales (2011)
  - MU-MIMO downlink (SDMA) – SAMURAI WPx, Newcom++ WPR2
    - Extensions to LTE MU-MIMO formats in 2010
    - Exploitation of reciprocity signaling for MU-MIMO (2010-2011)
    - Use of dual-stream interference canceller at UE for residual MU interference
  - 802.11p transceiver
    - Soft-MODEM for ExpressMIMO (PLATA, 2010-2011)
OpenAirInterface PHY Services in support of MESH or Relay-based deployments

- Over-the-air Network time/frequency-synchronization
  - Firefly synch between basestations (clusterheads) and relays

- Interference mitigation in receivers (through RX antenna processing)
  - Dual-stream reception from two separate sources
  - Interference cancellation of signal from concurrent transmission
Application Scenario 1 (LTE-A Network)

Questions/Issues

• Cheap basestations synch through OTAC? (lack of GPS, lack of sufficient network support)
• Over the air inter-basestations links are a mesh topology (relaying, interference, etc.)
• UEs (terminals) have handle interference in single-frequency networks - > dual-antenna receivers + MU-detection
Application Scenario 2 (Civil Protection)

Questions/Issues

• Largely the same as previous case but …
• Over the air inter-basestations links are still a mesh topology (relaying, interference, etc.) but here there is no backbone (i.e. high rate low-latency, spectral efficiency!)
Firefly Synchronization

- Everything is easier if we have (local) time synchronization in a wireless network
  - Distributed MIMO
  - Smart interference cancellation
  - Resource allocation/multiple-access (MAC)

- In practice, basestations are often synchronized using GPS and use high-precision expensive clocking (real-time network support too)
  - What if we lack GPS coverage (indoors, underground, etc.), we use cheap networks (ADSL, ethernet) for some basestations? -> over-the-air synchronization (and signaling too!)
Fireflies in Distributed MIMO networks

Primary Synch Source (basestation) $p_0(t)$

Synch relays (basestation or relay) each adjust timing based on $(p_0 * h_{0,i})(t - \tau_{0,i})$
and collaborate as a distributed array to synchronize a secondary synch source (basestation)

Secondary Synch source uses
$\sum (p_1 * h_{1,i})(t - \tau_{0,i} - \tau_{1,i})$
OpenAirInterface PHY/MAC Protocol Stack (3GPP-LTE like)
Layer 2 Topology

Cluster-Head
Mesh Router
Isolated Node
Edge Router
Other Access Technology
Label 1 – Voice communication
Label 2 – Video communication
Roles of Nodes @ L2

- **Role of CH is to**
  - Manage radio resources in cluster (scheduling of transmission opportunities/label schedules)
  - Provide mechanisms for measurement reporting to L3 (for routing, QoS management, etc.)

- **Role of Normal nodes**
  - Search (on behalf of CH) for isolated nodes to trigger topology updates which guarantee connectivity of the mesh network (not part of WIDENS protocol)
  - Potentially, it can perform opportunistic multiplexing, when CH schedules only TXops.

- **MAC provides services to mesh routing mechanisms (MPLS-like)**
  - Logical flows between CH for label management using appropriate metrics (achievable throughput, delay)
  - Broadcast flows from all nodes in mesh for route discovery and maintenance
L2.5 View

- Edge Router (Ingress/egress)
- Label switch routers
- 802.11 Node
- Free labels (potential)

Ingress function

MACm → PHYm → IP "MPLS" → MACm

MACn → PHYn → IP "MPLS" → MACn
Roles of nodes at L2.5

- Role of Edge Router is to
  - Aggregate traffic (ingress) from IP flows to MPLS-like labels
  - Demultiplex traffic (degress) from MPLS-like labels to IP

- Role of Label switching router
  - Switch incoming/outcoming labels according routing table

- All nodes in MESH use a three-way hand-shaking protocol (label REQ, label REP, label ACK) for label management
  - Path discovery
  - Path maintenance (as local as possible)

- This protocol can use the same logical resources as the L2 topology management services
L2 QoS Measurement reporting

- CH manages measurement reports @ L2 for nodes within the cluster.
- Measurements reports can be exchanged between nodes in the Mesh using a logical flow for topological control signaling.
- Edge routers can provide these measurements to IP.
- Measurements are attached to labels within the mesh.
L2 QoS Measurement Reporting

- Since the CH scheduler has access to low-level PHY measurements, the MAC layer is responsible for measurement reporting on behalf of the PHY and itself. The CH obtains raw measurements of all links in the cluster for which it is responsible.

- Measurements are processed in nodes to the degree required for higher level services. For example
  - Nodes will extract link quality (rate/delay) indicators from low-level services (MAC) to form L2.5 measurement messages which are transported using special signaling flows offered by the MAC. This is used for L2.5 topology maintenance (label (re-)assignement)
  - Edge routers will extract L2.5 measurement information on labels to provide IP with quality indicators
    - **Establishment**: HS Label – REP message comes with measurement information on label (number of hops/delay, rate, stability)
    - **Steady state**: periodic updates on label quality (to advise IP in advance of total degradation), one-shot updates IP when local recovery (L2.5) cannot assure service and reconfiguration at IP level is required.
OpenAirInterface
Simulation/Emulation Environment
Real-time distributed validation environment comprising
- IP Multicast over Ethernet: PHY emulation layer
- PHY behavioural abstraction models
- OpenAirInterface Layer two real-time protocol stack (openair2) potentially virtualized into $N_{\text{inst}}$ instances in the same physical machine.

Virtualization of protocol stacks (L2 + L3) in one machine
OpenAirInterface Emulation and Simulation Methodologies (cont’d)

- **Protocol Implementation Validation**
  - Enables developers of L2/L3 and applications to test their implementation in a real-time setting without the need for RF equipment
  - Repeatable and scalable real-time experiments (hundreds of nodes)

- **System Performance Evaluation**
  - For L2/L3 protocol assessment
  - Use of accurate and fast PHY abstraction models

- **Possibility of using real channel measurement traces as simulation stimulus (input from EMOS)**
Modes of Operation

- **Two modes**
  - Hard Real-time with virtualization (RTAI/Linux)
  - Soft Real-time (maintain frame timing on average) in Linux user-space

- **Soft real-time is simpler for debugging L2 protocol stack**
OpenAirInterface RT Emulation on PC Cluster

IPv6

VLAN EXP

R4G1
RF Topology Server

R4G2
R4G3
R4G4

R4G5
R4G6
R4G7
R4G8

IPv4

Application Data + Emulation Parameters

Emulation traffic (Transport channels)

Switch Gigabit Ethernet

IPv6, NAS Driver

RF Emulation Measurements, PHY Error

L2
RTAI

TKN-TUB, Berlin, August 20th 2009
PHY Emulation

- Each node in the emulated network (either on a distinct PC or in a different MAC instance within one PC) has/shares a kernel module responsible for PHY emulation which
  - Fully implements the MAC/PHY interface (Cellular/MESH)
  - transports MAC PDUs via IP Multicast between correspondent nodes or direct memory transfer
  - Generates measurements as would the real PHY
  - Uses a PHY abstraction simulation module to inject simulated error patterns in the different MAC-layer streams. This is configured dynamically by a radio topology server or locally
PHY Emulation (cont’d)
PHY Abstraction

- PHY Abstraction is done at the receiver of each node. Wideband SINR are computed every transmission frame based on the RF topology and pre-defined propagation models.
  - The function can be system dependent (i.e. based on precomputed probability of error simulations for specific modulation and coding formats) or generic (PHY-agnostic).

- The output of the radio simulation is random packet loss indicators for each transport channel block traversing the PHY/MAC interface. Alternatively, if erroneous packets are to be passed to the higher layers (in the spirit of UDP lite type protocols), bit errors must be generated in the MAC PDUs.
PHY Abstraction (cont’d)

- In each radio frame (TTI), the PHY Abstraction unit analyzes the set of received SDUs from the emulation medium and determines those which are sources of information and those which represent interference (based on local protocol information)
  - The targets to be received are the programmed by the MAC as with the true PHY.
  - The interferers are naturally present with the true PHY and thus their impact must be simulated in the abstraction unit. Since a particular node in the network is not aware of all sources of interference a priori, this is done by adding a PHY resource description to each transport block in the emulation medium which is not present in the real PHY.
PHY Abstraction Example

- Consider the following example for PHY abstraction at node \( j \) in the network.
- Let \( \overline{\text{RSSI}}_{i,j}[n] \) be the average received strength in frame (TTI) \( n \) between node \( i \) and node \( j \). This can be generated locally in each node based on a model for mobility or can be signalled by a topology server dynamically.

```
\{\overline{\text{RSSI}}_{i,j}[n], i \neq j\}
```

```
\{\overline{\text{PDP}}_{i,j}[n], i \neq j\},
\{\text{RiceanK}_{i,j}[n], j \neq i\}
```

MACPHY_DATA_IND for each received transport channel (transport blocks, measurements, crc status indicators)

Set of SDUs from Emulation Medium
The goal of the PHY abstraction entity is to simulate the block error rate process (BLER) of each transport block of a particular received resource.

This is fundamentally related to the statistics of the received signal and interference vectors at node $j$. Let $\sqrt{\text{RSSI}_{i,j}[n]H_{i,j}[n,k]}$ be the spatial channel in frequency band $k$ for the signal from node $i$ to $j$ and $K_{1,i,j,m}[n,k] = \sigma^2 + \sum_{i' \neq i, i' \neq j} \sum_{m=0}^{M(i')} \sqrt{\text{RSSI}_{i',j}[n]h_{i',j}[n,m,k]}h_{i',j}[n,m,k]^*$, where $h_{i',j}[n,m,k]$ is the spatial channel for transmit antenna $m$ in band $k$ corresponding to interferer $i'$. $M(i')$ is the number of transmit antennas corresponding to interferer $i'$.

For each received transport channel (transport blocks, measurements, CRC status indicators), MACPHY_DATA_IND is set.
PHY Abstraction Example (cont’d)

- Generation of the random variables $H_{i,j}[n,k]$ and $h_{i',j}[n,m,k]$ depend on the space/time/frequency description of the propagation environment (PDP, Ricean factor, antenna correlation, mobility)

- A description of the BLER as a function of
  \[
  \sqrt{\text{RSSI}_{i,j}[n]}H_{i,j}[n,k] \quad \text{and} \quad K_{I,i,j,m}[n,k]
  \]
  is required.
PHY Abstraction Example (cont’d)

- Description of the conditional BLER is the key issue. This can be done in a variety of ways

  1. for non-ARQ based coding and no transmit filtering (beamforming, dirty-paper, etc.)
     - for a particular receiver structure SINR expressions can be derived from the above and used for BLER lookup based on tabulated performance of a particular code.
     - In addition, PHY agnostic information-theoretic bounds can also be derived

  2. For ARQ-based schemes, errors at a particular time \( n \) also depend on past values of the signal and interference components. Here additional protocol information from the MAC signaling header are required, but similar semi-analytical models can be used.

  3. For transmit filtering (beamforming, dirty-paper, etc.) additional PHY layer information must be transported in the emulation process along with MAC information, namely the linear/non-linear spatial filtering description at the sending nodes.
PHY Abstraction Example (cont’d)

- Type 1 Generic Model for BLER
  - Under the assumption of Gaussian transmit signals and a particular receiver structure (i.e. MMSE which is optimal for Gaussian statistics) a model for achievable BLER (on each transport block sent by PHY to MAC) could be bounded

\[
P_e \left( \left\{ \sqrt{\text{RSSI}_{i,j}[n]}H_{i,j}[n,k] \right\}, \left\{ K_{I,i,j,m}[n,k] \right\} \right) \leq K_{\text{impl}} e^{-NTB(R(n) - R_{\text{MAC}}(n))}
\]

where \( K_{\text{impl}} \) is an implementation degradation factor, \( R_{\text{MAC}}(n) \) is the allocated rate by the MAC layer scheduler in TTI \( n \) and

\[
R(n) = \frac{1}{|A(n)|} \sum_{f \in A(n)} \ln (1 + \text{SINR}_{j}[n,f])
\]
CHORIST Field Trial
Recent Experimentation

A field trial (underground and medium-range outdoor) was carried out near Barcelona (Spain) in February 2009

Goals:
- Validation of rapidly-deployable mesh technologies
  - PHY/MAC (rapidly-deployable WiMAX/LTE-like radio access) – 1.9 GHz, 5 MHz channelization TDD, 2x2 MIMO-OFDMA, 23dBm
  - L3 (routing protocols + QoS management)
  - Public-safety applications (video sensing, group comms)
CHORIST Trials (Barcelona Feb 2009)

Single Cluster Deployment

Cluster Head

Router 1

Router 2

230 meters

630 meters

TKN-TUB, Berlin, August 20th 2009
Dual-Cluster Deployment

Cluster Head 1

Cluster Head 2

Router 1

Router 2

Router 3

5-node Dual Cluster Deployment

TKN-TUB, Berlin, August 20th 2009
Equipment ...

Command Center (MR1)

ClusterHead (Mobile Basestation CH1)

Remote Command Center with WSN + PS apps(MR2)
Some observations

- Distributed synchronization is feasible in medium-range outdoor scenarios
  - Receiver gain control is a bit of an issue in mesh networks but can be overcome by large dynamic range A/D (in our case 12-bits was sufficient to allow synch in the presence of strong and weak signals)
  - Larger networks?

- Distributed MIMO and dual-stream IC similarly but
  - Somewhat sensitive to propagation scenario (LOS/NLOS, angles) especially indoor
  - Power differences significantly alleviate this (with sufficient dynamic range in the receiver) for the max-logmap receiver (very big plus for max-logmap!)
Future similar networking experiments

- **Multiple-relaying / Two-way relaying**
  - Compression-based schemes (interaction with protocol stack!)
  - Hybrid ARQ strategies in collaborative networks (integration with protocol stack!)

- **Advanced Feedback-based transmission**
  - Build upon our MU-MIMO measurement campaigns (GC08, IEEE Wireless) to integrate precoding/feedback into our MODEMs (underway and part of NEWCOM++ activities)
  - Exploitation of reciprocity in TDD through two-way protocols

- **New equipment (20 MHz channels, 8 antennas!)**