EURECOM
Sophia Antipolis

OPEN AIR 5G INTERFACE
5G SOFTWARE ALLIANCE

OpenAirInterface 5G

Overview, Installation, Usage

Florian Kaltenberger

OAI workshop, BUPT, 27.4.2017

Unleashing the potential of open-source in the 5G arena

Overview

- Overview and Ecosystem
- Use cases
- Features (eNB, UE, EPC)
- Hardware targets
- Installation
- Usage
- Debugging tools

What is OpenAirInterface?

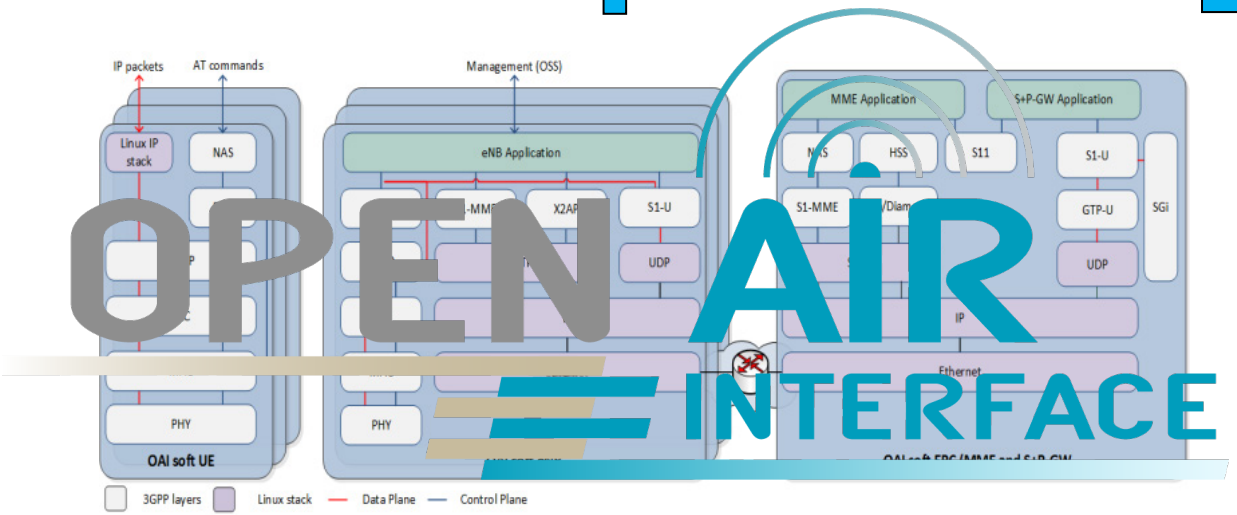
EURECOM

NATIONAL INSTRUMENTS

Ettus Research

nuand

Lte



OPEN AIR INTERFACE

What is OpenAirInterface?

- **Open-source software-based implementation of 3GPP LTE Rel 8/9**
  - Including features from LTE-Advanced (Rel 10/11/12), LTE-Advanced-Pro (Rel 13/14), going on to 5G Rel (15/16/…)
  - Spanning the full protocol stack of 3GPP standard
    - E-UTRAN (eNB, UE)
    - EPC (MME, S+P-GW, HSS)
  - Realtime RF and scalable emulation platforms
  - Works with many SDR platforms (ExpressMIMO2, USRP, LimeSDR, …)
- **Today it is feasible to put a fully-compliant 4G eNodeB and EPC in a commodity x86-based computer (or data center)**
- **Objectives**
  - Building a community of individual developers, academics and major industrials embracing open-source for 5G
  - Become a strong voice and maybe a game-changer in the 3GPP world
    - Real impact from “the little guys” on 3GPP systems

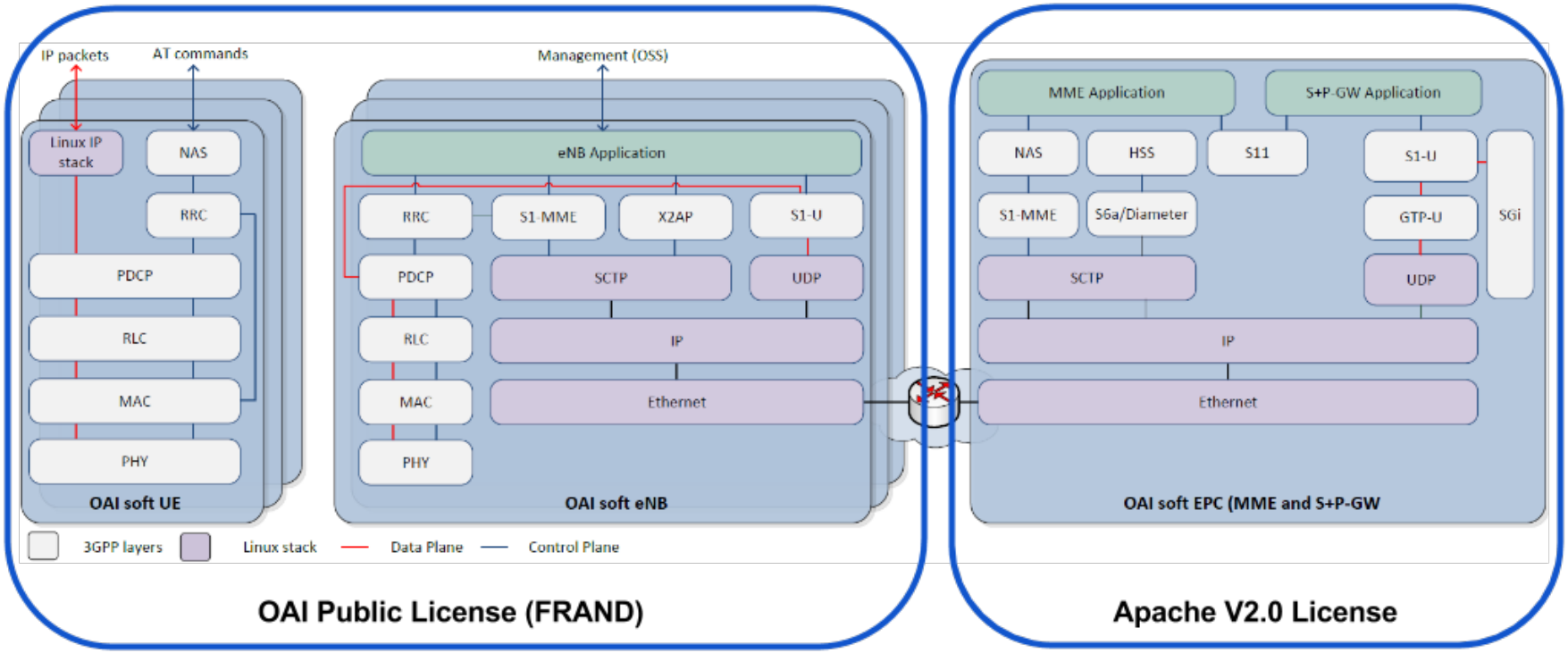
Collaborative Web Tools

- **www.openairinterface.org**
- **OpenAirInterface gitlab server (includes Wiki)**
  - RAN (eNB + UE)
    https://gitlab.eurecom.fr/oai/openairinterface5g
  - EPC
    https://gitlab.eurecom.fr/oai/openair-cn
- **Mailing list**
  - openair5g-user: for the users of OpenairInterface.
  - openair5g-devel: for the developers of OpenairInterface.
  - openaircn-user: for the users of OpenairCN.
  - openaircn-devel: for the developers of OpenairCN.
- **Forum in Chinese**
  - http://bbs.opensource5g.org/forum.php

The OpenAirInterface Software Alliance

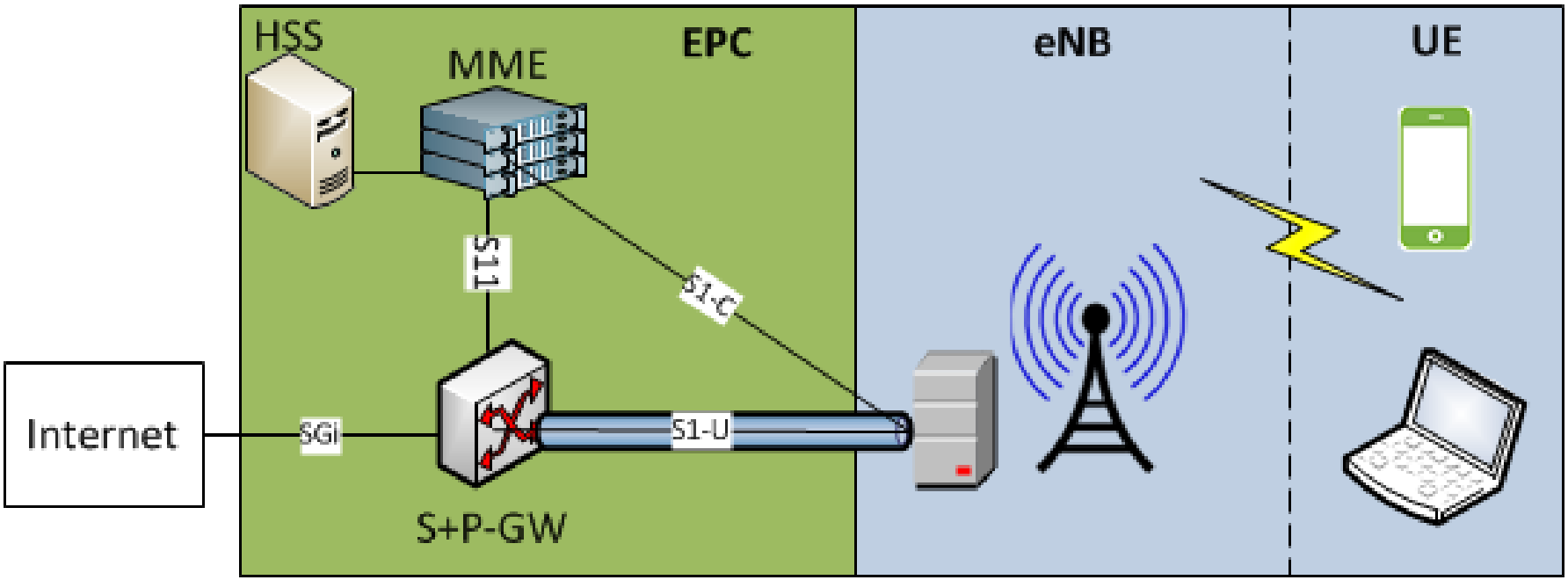
- **Launched in 2014 as a "Fonds de Dotation"**
- **Strategic members in 2015-2017 (Orange, TCL, Ercom, Nokia, Technicolor)**
- **Many associate members (Cisco, B-COM, INRIA, IMT, TNO, III, Rutgers WINLAB, U. Washington, IITH, BUPT, etc.)**
- **Goals:**
  - Promote OpenAirInterface and its open-source licensing model
  - Support the community of developers and users

The OAI Licensing model



- **FRAND License allows committing software with patent rights into OSA and still keep licensing rights -> Inline with 3GPP fair use licensing policy**
- **We aim to work closely with ETSI on implications of open-source for licensing/certification**
- **Future 5G Core Network developed within eNB/UE repository will inherit FRAND license**

Use case I: classical 3GPP network



- OAI EPC
- Commercial/3rd party EPC

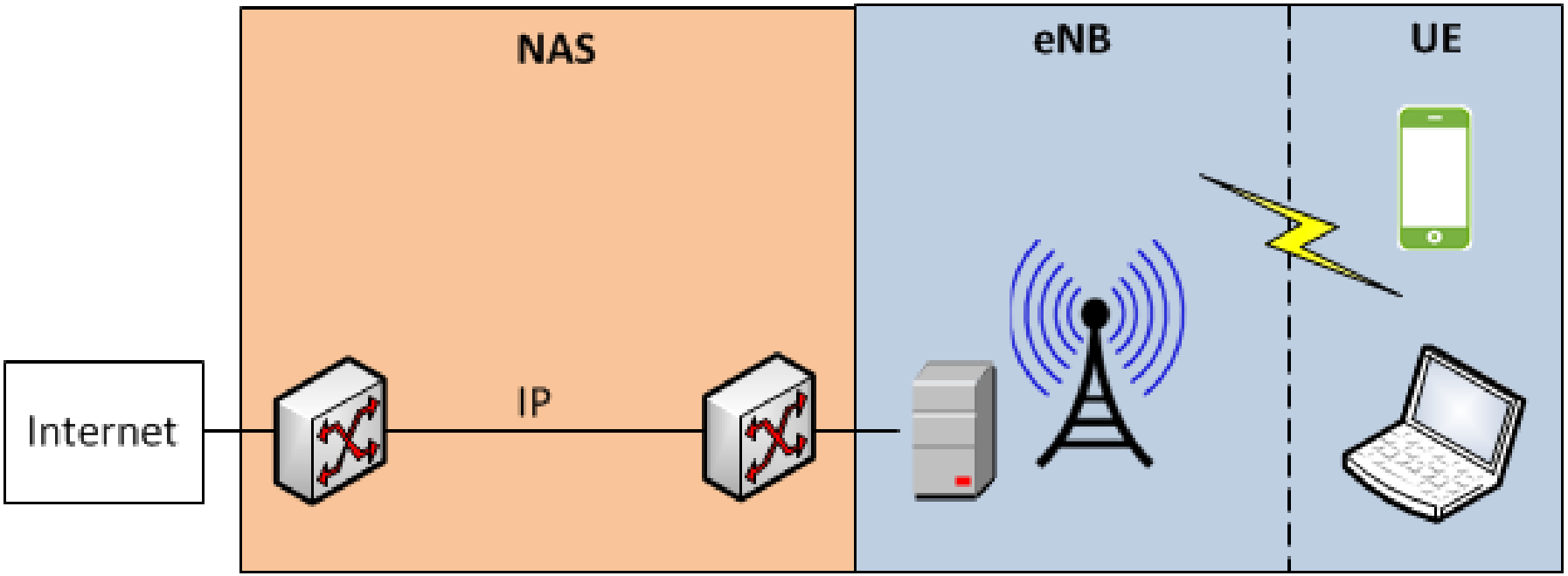
+

- OAI eNB
- Commercial/3rd party eNB

+

- OAI UE
- COTS UE

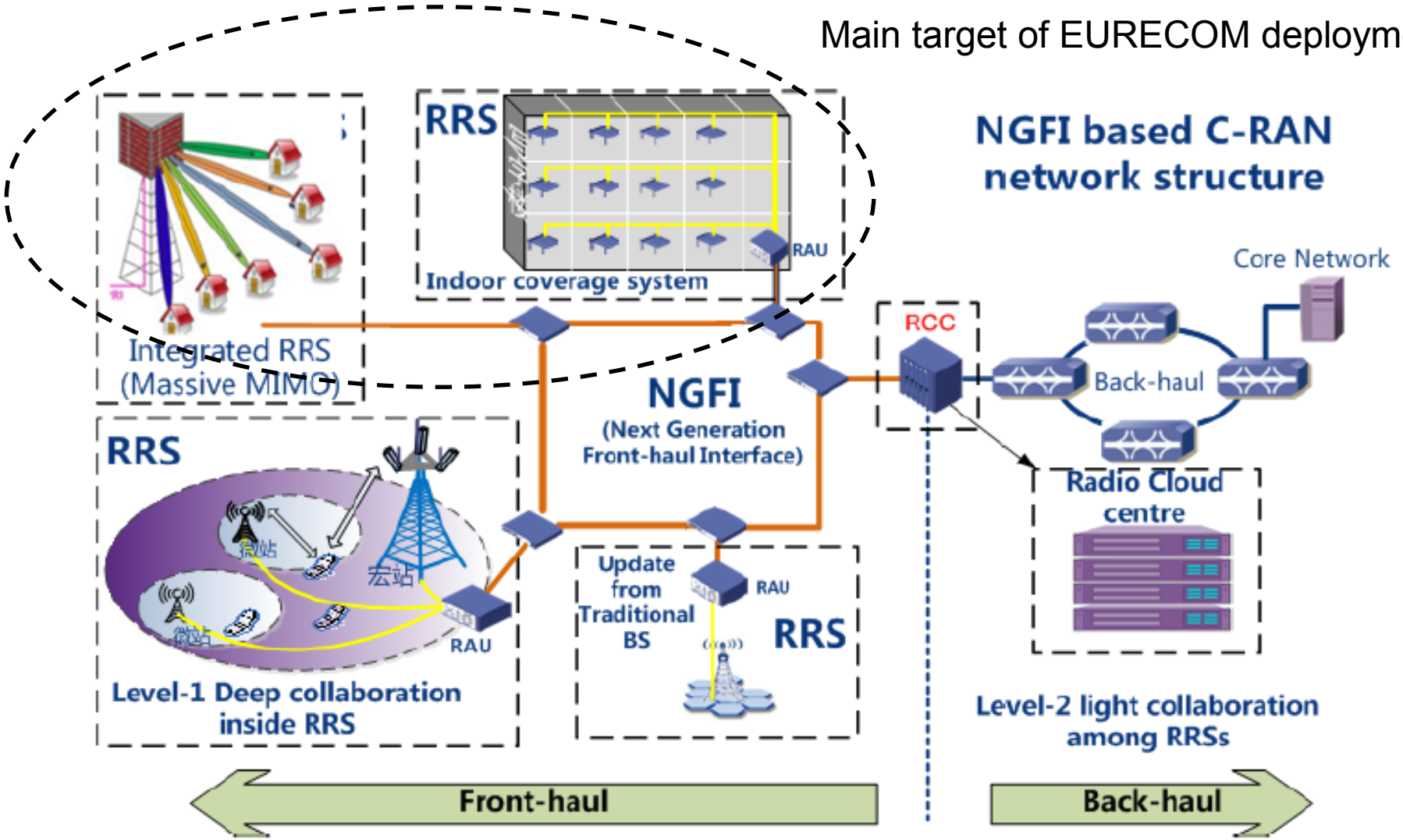
Use case II: simplified network



- **Non-3GPP setup (no-S1 mode):**
  - OAI eNB <--> OAI UE

Use case III: cloud-RAN

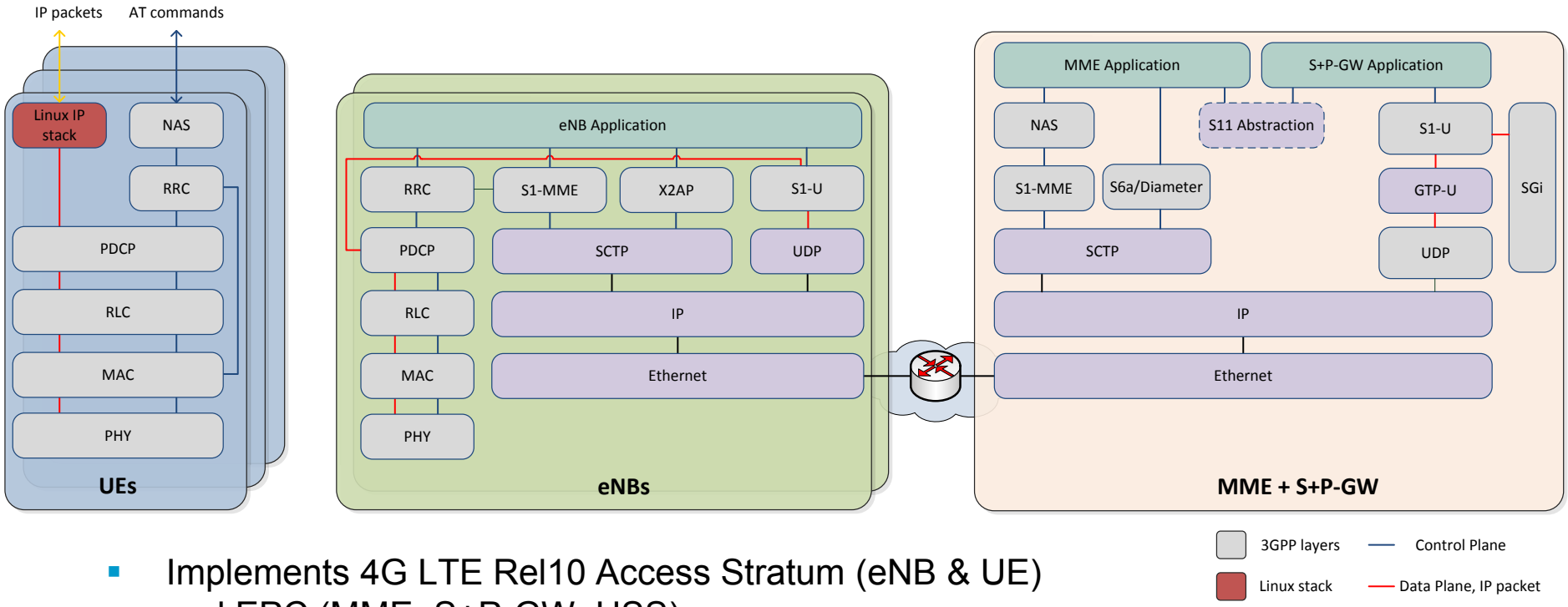
Main target of EURECOM deployment



Use case IV: Simulation/Emulation

- **Simulation/Emulation (oaisim)**
  - OAI eNB <--> OAI UE
  - OAI EPC + OAI eNB <--> OAI UE
  - Commercial/3rd party EPC + OAI eNB <--> OAI UE
- **Unitary simulators**
  - DLSCH simulator dlsim
  - ULSCH simulator ulsim
  - PUCCH simulator pucchsim
  - PRACH simulator prachsim
  - PDCCH simulator pdcchsim
  - PBCH simulator pbchsim
  - eMBMS simulator mbmssim
- **Other uses**
  - EMOS (real-time channel sounding)
  - octave (simple experimentation)

OpenAirInterface Features



- Implements 4G LTE Rel10 Access Stratum (eNB & UE) and EPC (MME, S+P-GW, HSS)
- All the stack (incl. PHY) runs entirely on a PC in real-time operating system
- Works with ExpressMIMO (Eurecom), USRP (Ettus/National Instruments), Blade RF (nuand), LMS-SDR (Lime Micro)

OpenAirInterface eNB features (PHY)

- **The Physical layer implements 3GPP 36.211, 36.212, 36.213 and provides the following features:**
  - LTE release 8.6 compliant, and implements a subset of release 10;
  - FDD and TDD configurations 1 (experimental) and 3;
  - Bandwidth: 5, 10, and 20 MHz;
  - Transmission modes: 1, 2 (stable), 3, 4, 5, 6, 7 (experimental);
  - Max number of antennas: 2
  - CQI/PMI reporting: aperiodic, feedback mode 3-0 and 3-1;
  - PRACH preamble format 0
  - All downlink (DL) channels are supported: PSS, SSS, PBCH, PCFICH, PHICH, PDCCH, PDSCH, PMCH;
  - All uplink (UL) channels are supported: PRACH, PUSCH, PUCCH (format 1/1a/1b), SRS, DRS;
  - HARQ support (UL and DL);
  - Highly optimized base band processing (including turbo decoder).
  - Expected throughputs DL
    - 5 MHz, 25 PRBS/ MCS 28 = 16-17 Mbit/s (measured with COTS UE Cat 3/4)
    - 10 MHz, 25 PRBS/MCS 28 = 34-35 Mbit/s (measured with COTS UE Cat 3/4)
  - Expected throughputs UL
    - 5 MHz, 20 PRBs / MCS 20 = 9 Mbit/s (measured with COTS UE Cat 3/4)
    - 10 MHz, 45 PRBs / MCS 20 = 17 Mbit/s (measured with COTS UE Cat 3/4)

OpenAirInterface eNB features (MAC)

- **The MAC layer implements a subset of the 3GPP 36-321 release v8.6 in support of BCH, DLSCH, RACH, and ULSCH channels.**
- **The eNB MAC implementation includes:**
  - RRC interface for CCCH, DCCH, and DTCH
  - Proportional fair scheduler
  - DCI generation
  - HARQ Support
  - RA procedures and RNTI management
  - RLC interface (AM, UM)
  - UL power control
  - Link adaptation

OpenAirInterface eNB features (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
  - Integrity check and encryption using the AES and Snow3G algorithms

OpenAirInterface eNB features (RLC)

- **The RLC layer implements a full specification of the 3GPP 36-322 release v9.3**
- **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, re-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

OpenAirInterface eNB features (RRC)

- **Based on 3GPP 36.331 v9.2.0.**
  - System Information broadcast (SIB 1, 2, 3, and 13)
  - RRC connection establishment
  - RRC connection reconfiguration (addition and removal of radio bearers, connection release)
  - RRC connection release
  - inter-frequency measurement collection and reporting (experimental)
  - eMBMS for multicast and broadcast (experimental)

OpenAirInterface UE features (PHY)

- **The Physical layer implements 3GPP 36.211, 36.212, 36.213 and provides the following features:**
  - LTE release 8.6 compliant, and implements a subset of release 10;
  - FDD (stable) and TDD configurations 1 and 3 (experimental);
  - Bandwidth: 5, 10 and 20 MHz
  - Transmission modes: 1, 2, 3 (stable), 4, 5, 6, 7 (experimental);
  - Max number of antennas: Rx: 2, Tx: 1
  - CQI/PMI reporting: aperiodic, feedback mode 3-0 and 3-1;
  - All downlink (DL) channels are supported: PSS, SSS, PBCH, PCFICH, PHICH, PDCCH, PDSCH, PMCH;
  - All uplink (UL) channels are supported: PRACH, PUSCH, PUCCH (format 1/1a/1b/2), SRS, DRS;
  - HARQ support (UL and DL);
  - Highly optimized base band processing (including turbo decoder).
  - Carrier frequency synchronization (around a given frequency)

OpenAirInterface UE features (MAC)

- **The MAC layer implements a subset of the 3GPP 36-321 release v8.6 in support of BCH, DLSCH, RACH, and ULSCH channels.**
- **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

OpenAirInterface UE features (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
  - Integrity check and encryption using the AES and Sonw3G algorithms

OpenAirInterface UE features (RLC)

- **The RLC layer implements a full specification of the 3GPP 36-322 release v9.3**
- **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, re-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

OpenAirInterface UE features (RRC)

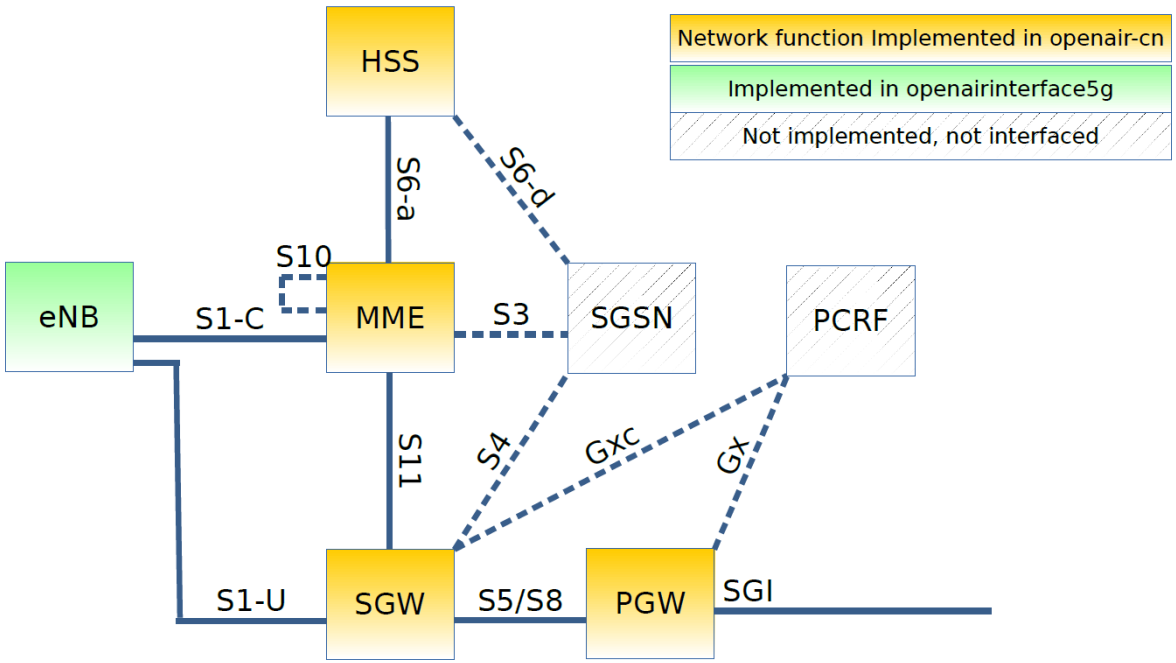
- **Based on 3GPP 36.331 v9.2.0.**
  - System Information reception (SIB 1, 2, 3, and 13, all others just displayed)
  - RRC connection establishment
  - RRC connection reconfiguration (addition and removal of radio bearers, connection release)
  - RRC connection release
  - inter-frequency measurement collection and reporting (experimental)
  - eMBMS for multicast and broadcast (experimental)

OpenAirInterface UE features (NAS)

Implemented
Partially implemented
Not implemented

- **EMM procedures:**
  - GUTI reallocation
  - authentication;
  - security mode control;
  - identification;
  - EMM information.
- **EMM specific procedures:**
  - attach
  - combined attach.
  - detach and combined detach.
  - normal tracking area updating and combined tracking area updating (S1 mode only);
  - periodic tracking area updating (S1 mode only).
- **EMM connection management procedures (S1 mode only):**
  - service request.
  - paging procedure.
  - transport of NAS messages (for SMS);
  - generic transport of NAS messages.
- **ESM Procedures related to EPS bearer contexts:**
  - default EPS bearer context activation;
  - dedicated EPS bearer context activation;
  - EPS bearer context modification;
  - EPS bearer context deactivation.
- **ESM Transaction related procedures:**
  - PDN connectivity procedure;
  - PDN disconnect procedure;
  - bearer resource allocation procedure;
  - bearer resource modification procedure.
  - ESM information request procedure.
  - ESM status procedure;
  - notification procedure.

OpenAirInterface EPC overview



- **MME, HSS, S/P-GW: different executables, can run on same host,**
- **eNB can also run on same host, but recommended on different host**

OpenAirInterface EPC features

Implemented
Partially implemented
Not implemented

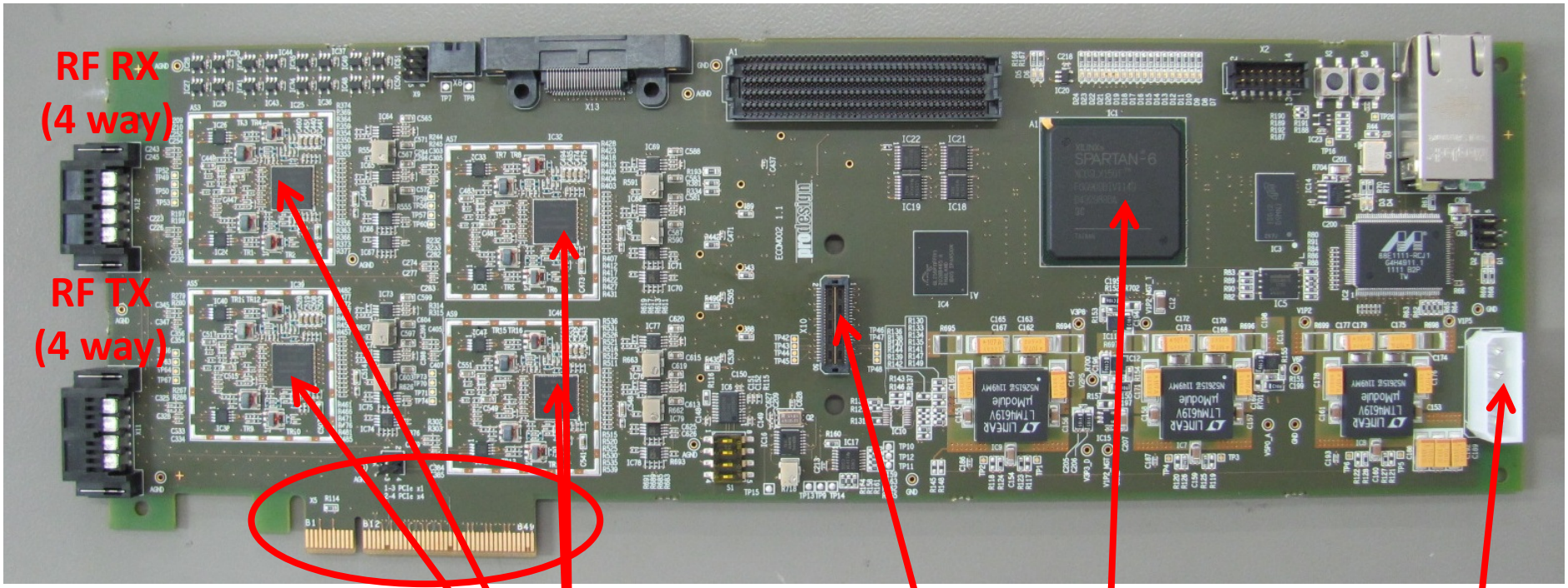
- **E-UTRAN Initial attach**
  - Attach with IMSI
  - Attach with GUTI
- **Tracking Area Update procedures: Always reject**
- **Routing Area Update procedures**
- **Service Request procedures**
  - UE triggered Service Request: always reject
  - Network triggered Service Request
- **S1 Release procedure**
- **GUTI Reallocation procedure**
- **Detach procedure**
  - UE-Initiated Detach procedure for E-UTRAN
  - MME-Initiated Detach procedure for E-UTRAN
  - HSS-Initiated Detach procedure for E-UTRAN
- **HSS User Profile management function procedure**
- **Bearer deactivation**
  - PDN GW initiated bearer deactivation
  - MME initiated Dedicated Bearer Deactivation
- **Dedicated bearer**
  - Activation through config file originating from P-GW
  - Waiting for Gx interface to be implemented
- **Intra E-UTRAN handover**

HARDWARE TARGETS

Hardware Requirements

- **SDR platform**
  - ExpressMIMO2
  - USRP B2x0, X300
  - Blade RF
  - LMS-SDR
  - Sidekiq (experimental)
- **A powerful x86 PC**
  - Intel Core i5, i7
  - Intel Xeon
  - Intel Atom
  - 4 cores, > 3GHz, SSE 4, AVX
- **ARM (experimental)**
  - Odroid
- **Antennas, Duplexers, etc**

Express MIMO 2

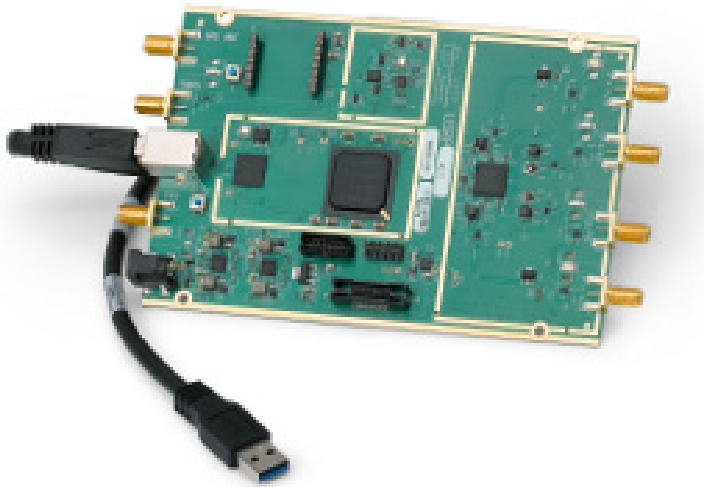


Express MIMO 2

- **Integrated baseband/RF PCI Express board for x86-based software defined radio**
- **Xilinx Spartan 6 FPGA**
- **4 RF chains based on LIME LMS6002D Semiconductor zero-IF RF chipsets**
  - Eurecom board, designed and maintained by EURECOM
  - 1.5/5/10/20 MHz, FDD/TDD
  - 4 channels (4x4 MIMO or 4 SISO Component Carriers)
  - Total aggregate bandwidth: full duplex 64Msps (Corresponds to 4x5MHz, 2x10MHz, or 1x 20MHz full duplex)
  - Carrier frequencies: 300 MHz – 3.8 GHz
  - ~10 dBm output power
  - LTE RF compliance (UE, small-cell eNB)
- **Status:**
  - more than 60 cards currently fabricated
  - used by many research institutes (academic and industrial)

USRP B210 (mini)

- **Designed by ETTUS (now part of NI)**
- **Analog Devices AD9361 RFIC Dual Channel Transceiver (70 MHz - 6GHz)**
- **Full duplex, MIMO (2 Tx & 2 Rx) operation with up to 56 MHz of real-time bandwidth (61.44MS/s quadrature)**
  - Slightly less in our experiments
- **Data acquisition over USB3**

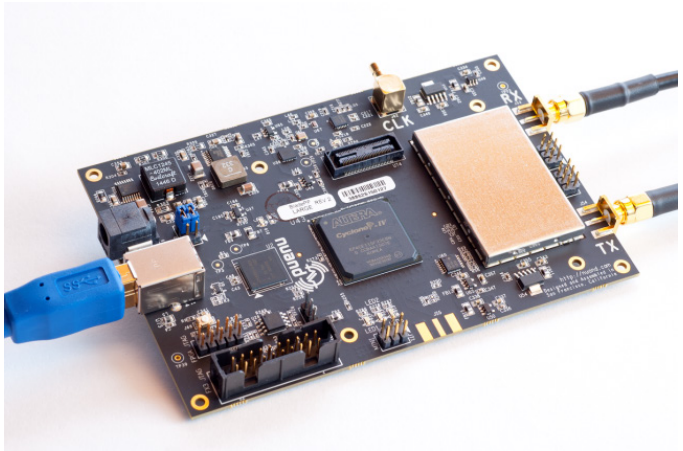


USRP X310

- Designed by ETTUS (now part of NI)
- 2 TX/RX chains, 120Msps
- Several RF daughterboards available covering DC to 6 GHz
- Xilinx Kintex 7 FPGA
- 10Gbit Ethernet or 4x PCIexpress

Blade RF

- Designed by nuand
- Based on LimeMicro LMS6002D (same as ExpressMIMO2)
- Altera Cyclone IV FPGA
- USB3



LMS SDR

- **Designed by Lime Microsystems**
- **Based on LMS7002M**
  - Same as LMS6002D but with 2TX/2RX
- **Altera Cyclone IV FPGA**
- **USB3**

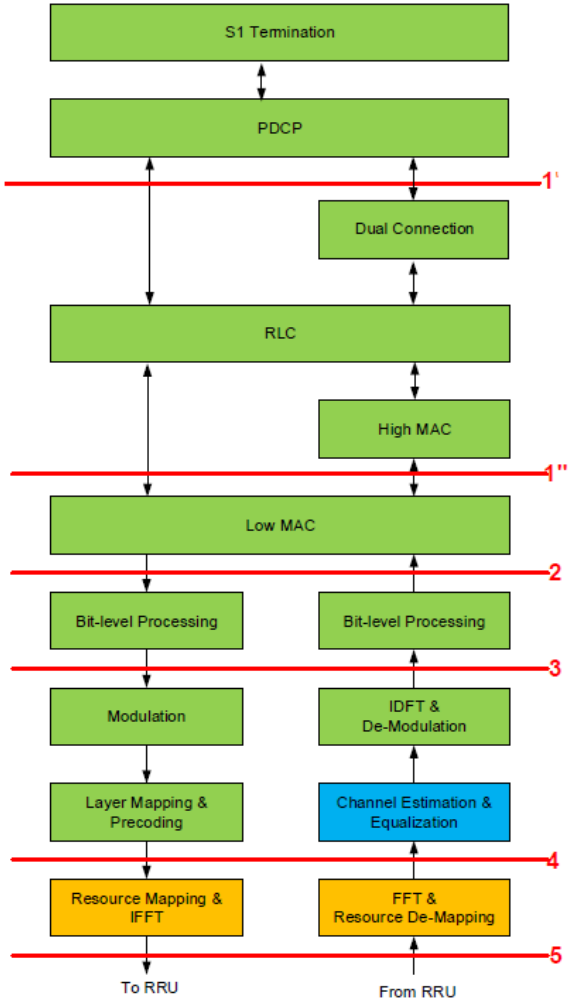
Epiq Sidekiq

- **Based on AD 9361 chipset**
  - 70MHz - 6GHz with up to 50MHz bandwidth per channel
- **SidekiqTM - MiniPCIe**
  - MiniPCIe card form factor (30mm x 51mm x 5mm)
  - 2 independet RF channels (2xRx or Tx+Rx)
  - PCIe Gen1.1 x1 (2.5 Gbps) interface to host + USB 2.0 interface
- **SidekiqTM - M.2**
  - M.2 T3042-D3-B card form factor (30mm x 42mm x 4mm)
  - Up to 2x2 MIMO
  - PCIe Gen2 x1 (5 Gbps) interface to host + USB 2.0 interface
- **Under beta-testing**

Comparison

	USRP B210	USRP X310	ExpressMIMO 2	Blade RF	LMS SDR
Data acquisition	USB3	Gbit EtherNet, PCIexpress	PCIexpress	USB3	USB3
MIMO and bandwidth capabilities	2x1 MIMO 20MHz or 2x2 MIMO 10MHz	2x2 MIMO, 120MHz	4x4 MIMO 5 MHz, 2x2 MIMO 10Mhz, SISO 20MHz	1x1 SISO 20MHz	2x2 MIMO 20MHz
RF chip	AD9361	n/a	LMS6002D (x4)	LMS6002D	LMS7002M
Frequency range	70MHz – 6GHz	DC-6GHz (depends on daughterbrd)	300 MHz – 3.8GHz	300 MHz – 3.8GHz	300 MHz – 3.8GHz
Price	€1,130	~€5,000	~€3,000	$420	$299
Duplexing	FDD or TDD	FDD or TDD	FDD or TDD	FDD	FDD or TDD
Output power	10dBm	10dBm	0dBm@ 2.6GHz 10dBm @ 700MHz	6dBm	10dBm
Noise figure	<8dB	?	10-15dB	?	?

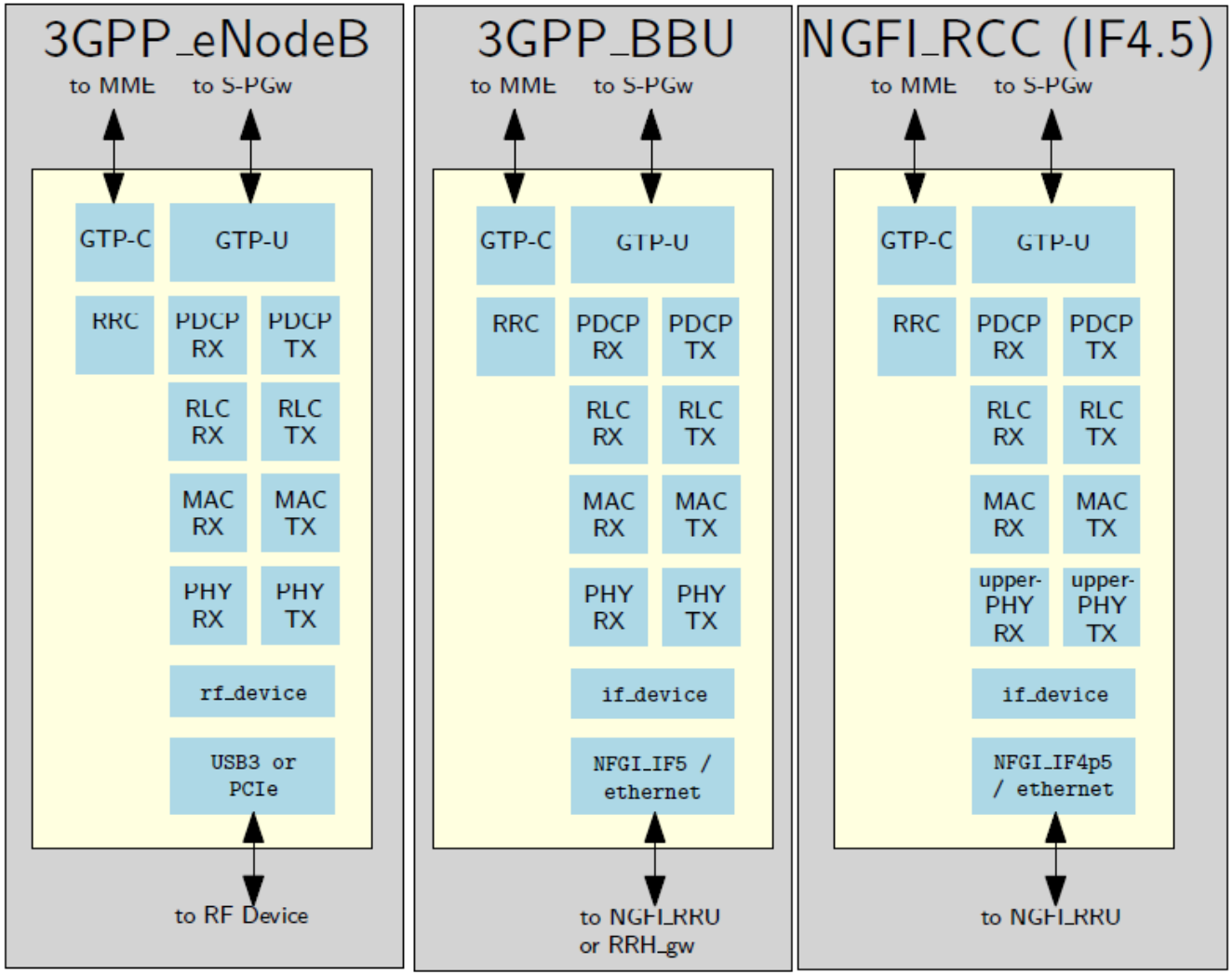
Functional splits in OAI



IEEE 1914 NGFI functional splits

- **Current OAI implementation supports**
  - IF5 time-domain fronthaul (> 1 GbE required)
  - IF4.5 split (FFTs) (280 Mbit/s/antenna port/20 MHz carrier)
  - IF2 via NFAPI (SCF082.09.04) under integration
  - IF1’ will follow 3GPP specifications (ongoing)

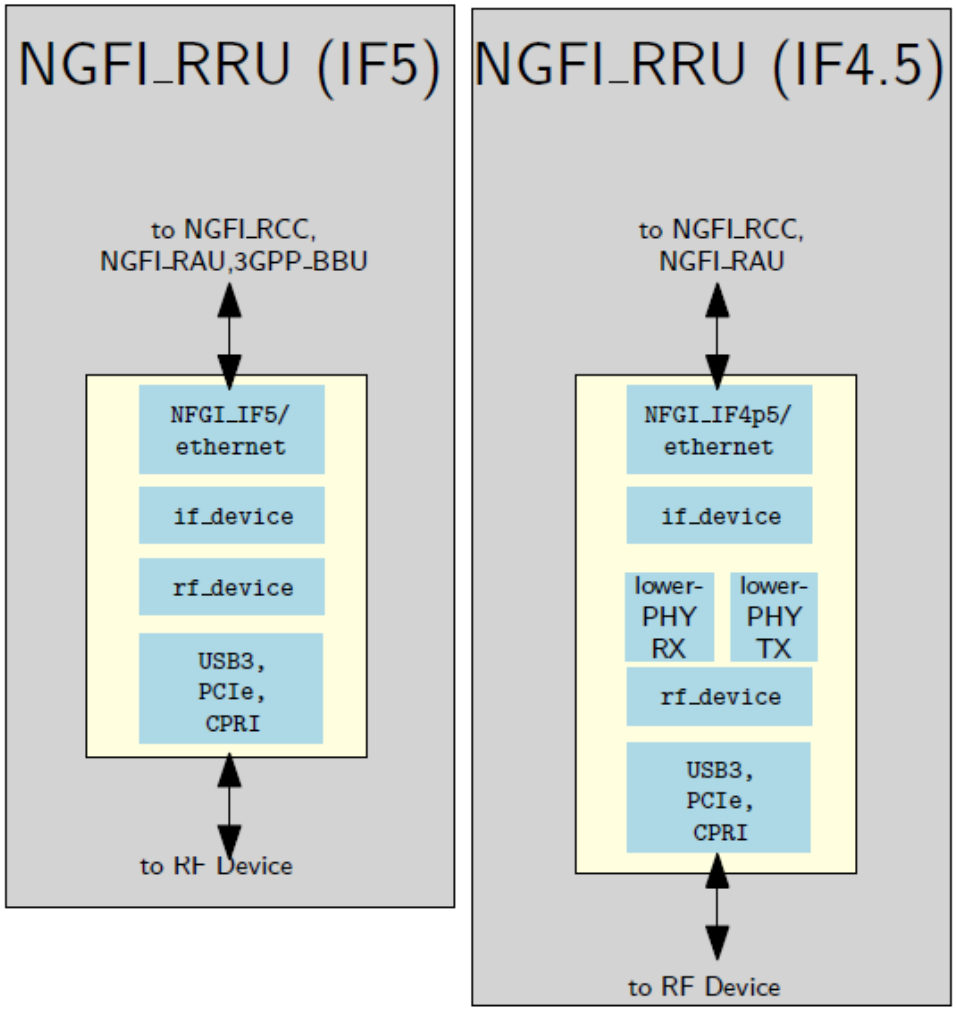
Functional units in OAI



BBU = Baseband Unit

RCC = Radio Cloud Center

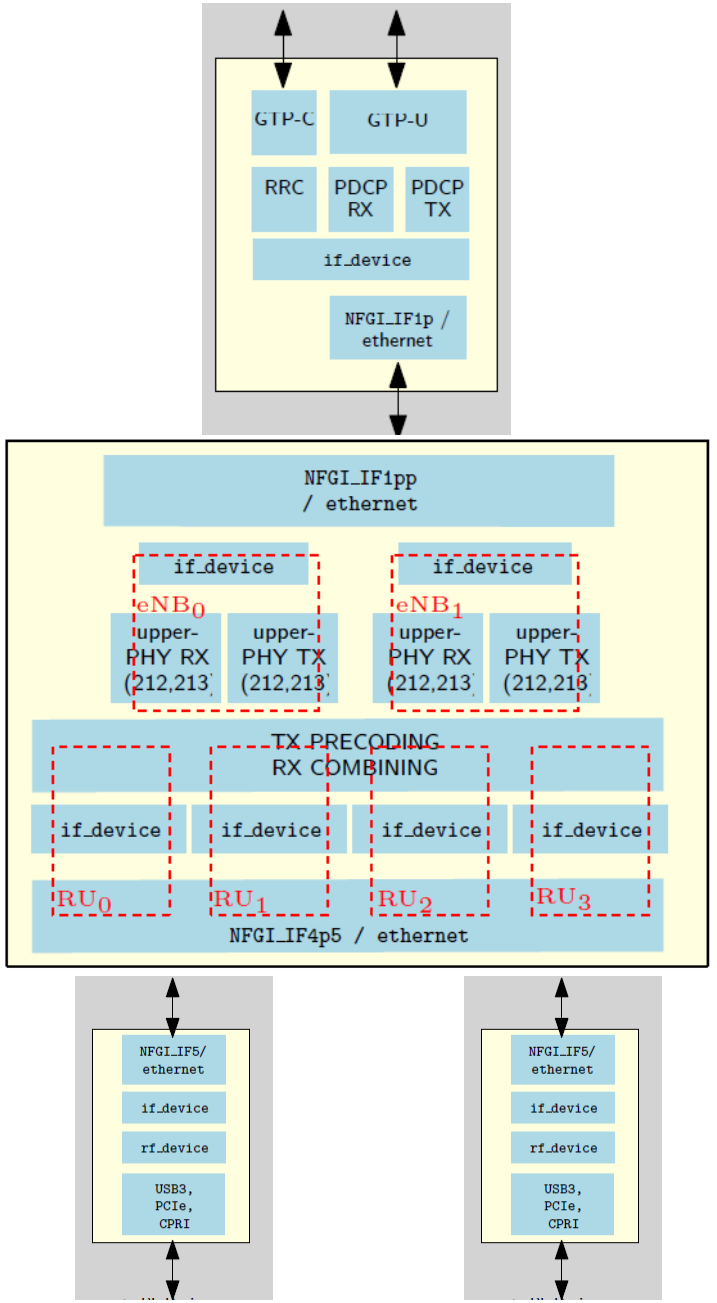
Functional units in OAI



RRU =
Remote Radio Unit

RCC/RAU/RRU

- **Radio Cloud Center (RCC)**
  - multiple RRC/PDCP entities
- **Radio-Access Unit (RAU)**
  - multiple MAC/RLC entities with medium-latency midhaul
  - L1 entities with low-latency fronthaul.
  - Under construction
- **Remote Radio-Unit (RRU)**
  - Equipment at radio site.
  - IF5 or IF4.5



Other experimental targets

- **CPRI - PCIexpress**
  - IT Avero
  - Based on Xilinx eval board
- **CPRI gateway**
  - Bell Labs
  - Based on Xilinx or Intel platform
- **Syrtem UED platform**
- **NXP QorIQ9131 platform (experimental)**

INSTALLATION

Software Requirements

- **Host PC configuration**
  - Disable power saving mode (sleep states)
  - Enable maximum performance (BIOS, cpu gouvenor)
- **Operating system**
  - Ubuntu 16.04.2, kernel 4.8
    - works for both openairinterface5g and openair-cn
    - For real-time operation, a low-latency kernel is recommended
    - For P/S-GW, gtp kernel module needs to be patched
    - See details on Wiki
- **Get code from our gitlab server**
  - RAN (eNB+UE): https://gitlab.eurecom.fr/oai/openairinterface5g
    - Branch master most stable
    - Branch develop latest features (recommended)
    - Several feature branches for cutting-edge developments
  - EPC: https://gitlab.eurecom.fr/oai/openair-cn
    - Branch master most stable (recommended)
    - Branch develop latest features

OpenAirInterface5G directories

- **cmake_targets**
  - New directory for building all the targets
  - Contains “mother” build_oai script
- **targets**
  - Hardware specific code (drivers, tools, etc)
  - lte-softmodem, oaisim
- **openair1**
  - Basic DSP routines for implementing subset of LTE specifications under x86 (36.211, 36.212, 36.213 3GPP specifications)
  - Channel simulation, sounding and PHY abstraction software,
- **openair2**
  - MAC/RLC/PDCP/RRC
- **openair3**
  - Contains interfaces S1-C, S1-U (GTP, SCTP, S1AP) and NAS UE
- **common/utils**
  - Utilities such as the T tracer or the ITTI

Compiling OpenAirInterface5G

- **Top-level build script located in**
  - cd openairinterface5g/cmake_targets
- **Compile lte-softmodem**
  - ./build_oai
  - -I installs additional required software
  - -w <hw_target> select HW target
  - --eNB compiles the lte-softmodem (for UE and eNB)
  - -x compiles with support for xforms softscope
  - -V compiles with support for VCD debugging
  - --UE compiles UE specific NAS parts
  - --T-tracer compiles with T support
  - --lte-simulators compiles the unitary simulators
  - -h help
- **This creates executables in**
  - openairinterface5g/targets/bin

Compiling OAI EPC + HSS

- **Top-level build script located in**
  - cd openairCN/SCRIPTS
- **For dedicated RB support**
  - Install kernel sources
  - Patch kernel using patch in /build/tools
- **Compile EPC**
  - ./build_spgw
    - -i installs additional software packages*
  - ./build_mme
    - -i installs additional software packages*
    - requires FQDN to be set in /etc/hosts
  - ./build_hss
    - -i installs additional software packages*
    - -l installs database

*when asked to set password for root, use “linux” (or change it later in config file).

HSS and SIM card configuration

- **Configuration file: /usr/local/etc/oai/hss.conf**
- **Use PHPmyadmin: http://yourhsshost/phpmyadmin**
  - User: hssadmin, password: admin
- **Add your MME**
- **Add your user**
- **Configure your SIM card**
  - Use a blank SIM card, card reader and programming tool,
  - or a pre-programmed SIM card
- **See Wiki for details**

EPC and eNB configuration

- **EPC configuration**
  - /usr/local/etc/oai/mme.conf, /usr/local/etc/oai/spgw.conf
    - Check MCC, MNC, TAC
    - Check IP addresses for interfaces
    - Take care of S-GW list selection
- **eNB configuration**
  - targets/PROJECTS/GENERIC-LTE-EPC/CONF/
  - Select the config file that is most appropriate for your configuration (Band and Hardware)
  - Check
    - MCC, MNC, TAC
    - downlink_frequency
    - mme_ip_address
    - IP addresses of S1-MME and S1-U interfaces

Running OAI

- **Running EPC**
  - ./run_hss
  - ./run_mme
  - ./run_spgw
- **Run eNB**
  - sudo ./lte-softmodem –O <file.conf> -d –V
- **Run UE**
  - sudo ./lte-softmodem –U –C <freq> -r [25|50|100] –ue-scan-carrier –ue-txgain xx –ue-rxgain yy
- **Have fun!**

Troubleshooting

- **eNB not connection to MME / RRH**
  - Check IP addresses in config files
- **I get a lot of UUUs and LLLs**
  - Check the performance setting of CPU (C-states, CPU frequency)
  - Check USB3 connection (some cables are bad)
- **Phone does not connect**
  - Analyze S1AP messages in wireshark
  - Check keys in SIM card and HSS
  - …
- **Throughput is very low**
  - Check radio conditions: duplexer, antennas, interference

DEBUGGING TOOLS

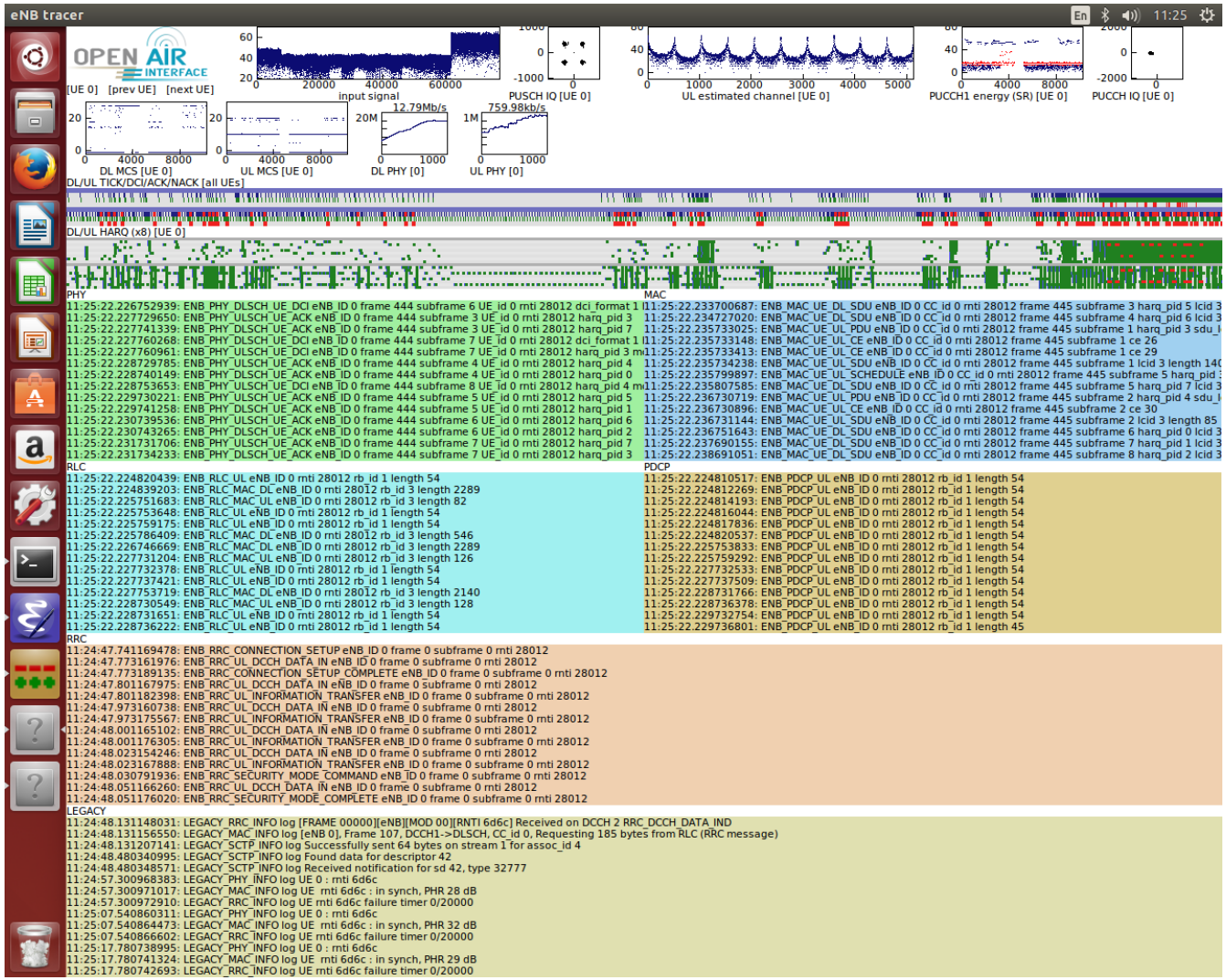
Debug tools

- **New: The T tracer**
- **MME scenario player**
  - Emulates S1AP to test control plane of EPC w/o RAN
  - For non-regression
  - Capture, replay and test scenarios
- **Legacy tools**
  - Wireshark / PCAP Interface
  - Logging
  - OAI Soft Scope and Status
  - Itti analyzer
  - OpenAirInterface performance profiler: for processing time measurement
  - OpenAirInterface timing analyzer: for real-time performance analysis
  - OAI message sequence chart (MSC)
  - Command line interface (CLI)

The T tracer

- **The T tracer is a framework to debug and monitor the softmodem.**
- **Combines logging, timing analysis, signal visualization, MAC PDU analysis (with wireshark)**
- **It is made of two main parts:**
  - an events collector integrated to the real-time processing,
  - a separate set of programs to receive, record, display, replay and analyze the events sent by the collector.
- **Can work locally or over network**

eNB GUI



- HARQ ACK
- HARQ NAK
- New DCI
- Retr. DCI

BACKUP

Debug tools

- **Spectrum Analyzer (UL and DL)**
  - Shows RF performance and signal integrity
- **Logs**
  - Verbosity can be adjusted in config file
  - Shows L2/L3 events
- **PHY scope**
  - signals in time and frequency domain
  - Constellation plots of PUSCH, PUCCH
- **Stats window**
  - eNB measurements (noise, signal power, etc)
  - UE feedback (CQI, etc.)
  - UL and DL HARQ statistics
- **VCD file**
  - Analyze real-time behavior
  - gtkwave -a ~/openairinterface5g/targets/RT/USER/eNB_usrp.gtk
- **Wireshark**
  - To analyze messages over S1 interface
  - Can also analyze MAC, RLC, PDCP, RRC if enables (see twiki for details)
- **Iperf/speedtest**
  - Shows throughout for UDP and IP

OAI Packet tracer API
Interface wireshark

- **Supported information**
  - MAC_LTE_RNTI_TAG; MAC_LTE_UEID_TAG; MAC_LTE_SUBFRAME_TAG; MAC_LTE_PAYLOAD_TAG
- **How to enable**
  - Lte-softmodem
    - Wireshark: lte-softmodem –W (capture in localhost)
    - Pcap: lte-softmodem -P /tmp/oai.pcap
  - Oaisim
    - ./oaisim -P wireshark (capture in localhost)
    - ./oaisim -P pcap (output goes to /tmp/oai_opt.pcap)
- **How to configure wireshark**
  - try heuristics for the UDP protocol, MAC-LTE, RLC-LTE, and PDCP-LTE
- **More information can be found at**
  - https://gitlab.eurecom.fr/oai/openairinterface5g/wikis/IttiAnalyzer

Openairinterface5g LOG APIs

- **LOG_X(COMPONENT, format_string, args …)**
- **Logs are formatted as follows :**

[COMPONENT][LOG LEVEL][FUNC][FILE][NODE ID][FRAME NUM] [CONTENT]

  - COMPONENT : RRC, PDCP, RLC, MAC, PHY, …
  - LOG LEVEL: Emerge, Alert, Critic, Error, Warning, Notice, Info, Debug, Trace
  - FUNC : name of the function inside which the log is called. This is optional
  - FILE: add the file name
  - NODE ID: eNB or UE with their ID
  - FRAME NUM: frame and subframe number
  - CONTENT: content of the log message
  - LOG_FULL include FILE line

- Log verbosity mask
  - LOG_LOW: include the component
  - LOG_MED includes include the level of the log
  - LOG_HIGH includes include function name
  - LOG_FULL include the file name

- LOG Level
  - LOG_EMERG :: LOG_G
  - LOG_ALERT :: LOG_A
  - LOG_CRIT :: LOG_C
  - LOG_ERR :: LOG_E
  - LOG_WARNING :: LOG_W
  - LOG_NOTICE :: LOG_N
  - LOG_INFO :: LOG_I
  - LOF_DEBUG :: LOG_D

Openairinterface5g LOG APIs How to configure

- **Option "-l " with the level as a number**
  - 0 lowest
  - 9 highers
- **Configuration file (for the moment, only valid for lte-softmodem)**

```
log_config :
   {
     global_log_level                ="trace";
     global_log_verbosity            ="medium";
 ..
}
```

- **Manually in oaisim_config.c (func olg_config) or in lte-softmodem local variables**
- **oaisim with option "-c" and xml configuration file**

```
<EMULATION_CONFIG>
 <LOG>  <!-- set the global log level -->
     <LEVEL>debug</LEVEL>
     <INTERVAL>1</INTERVAL>
   </LOG>
   <SEED_VALUE>1234</SEED_VALUE>  <!-- value 0 means randomly generated by OAI -->

  </EMULATION_CONFIG>
```

- **Source files**
  - openairinterface5g/openair2/UTIL/LOG/

Openair-CN Log API

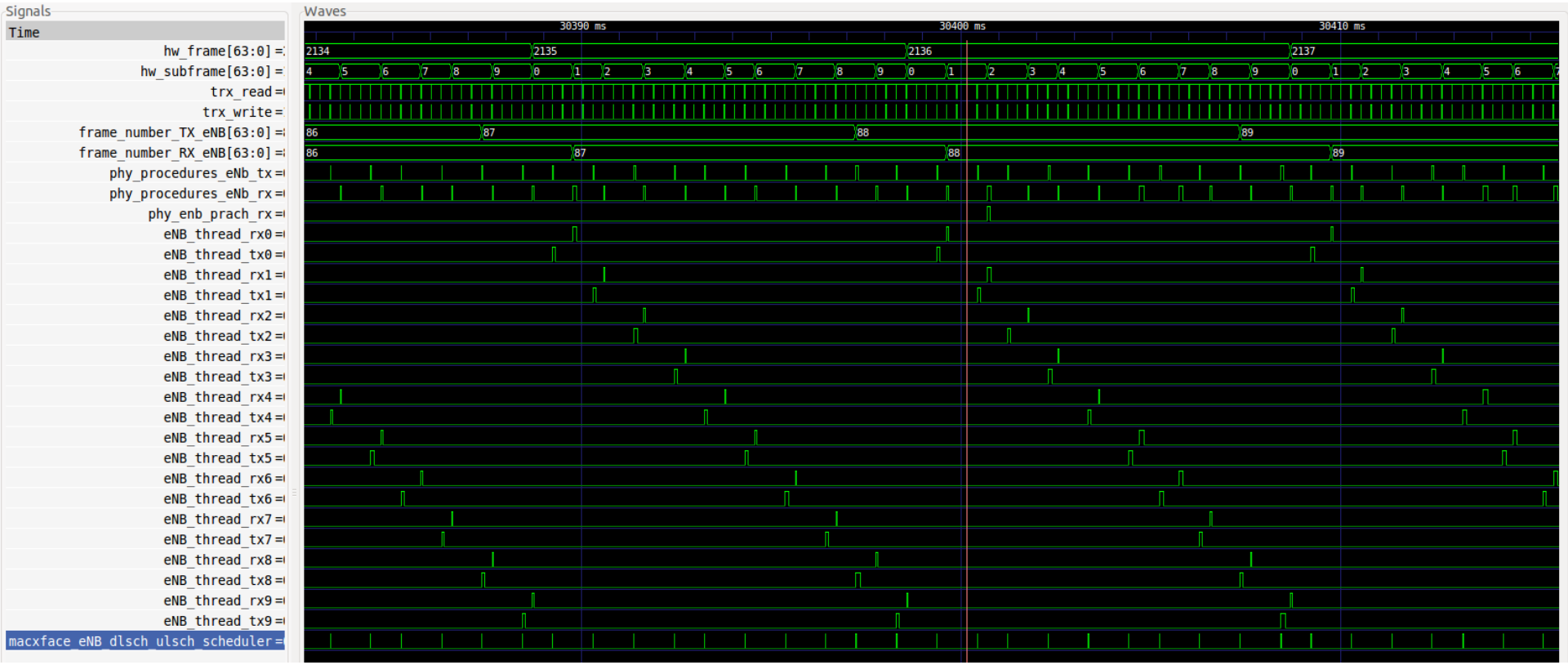
- OAILOG_FUNC_IN(pROTO)
- OAILOG_FUNC_RETURN(pROTO,RC)
- OAILOG_EMERGENCY(pROTO, args…)
- OAILOG_ALERT(pROTO, args…)
- OAILOG_CRITICAL(pROTO, args…)
- OAILOG_ERROR(pROTO, args…)
- OAILOG_WARNING(pROTO, args…)
- OAILOG_NOTICE(pROTO, args…)
- OAILOG_INFO(pROTO, args…)
- OAILOG_DEBUG(pROTO, args…)
- OAILOG_TRACE(pROTO, args…)

- **Source**
  - openair-cn/SRC/UTIL/

```
LOGGING :
  {
    # OUTPUT choice in { "CONSOLE"}
    # `path to file` must start with '.' or '/'
    # if TCP stream choice, then you can easily dump the traffic on the remote or local host: nc -l `TCP port num` > received.txt
    OUTPUT          = "CONSOLE";

    # COLOR choice in { "yes", "no" } means use of ANSI styling codes or no
    COLOR           = "yes";                                    # TODO

    # Log level choice in { "EMERGENCY", "ALERT", "CRITICAL", "ERROR", "WARNING", "NOTICE", "INFO", "DEBUG", "TRACE"}
    SCTP_LOG_LEVEL    = "TRACE";
    S1AP_LOG_LEVEL    = "TRACE";
    NAS_LOG_LEVEL     = "TRACE";
    MME_APP_LOG_LEVEL = "TRACE";
    S6A_LOG_LEVEL     = "TRACE";
    UTIL_LOG_LEVEL    = "TRACE";
    MSC_LOG_LEVEL     = "ERROR";
    ITTI_LOG_LEVEL    = "ERROR";

    # ASN1 VERBOSITY: none, info, annoying
    # for S1AP protocol
    ASN1_VERBOSITY    = "none";
  };
};
```

OAI time analyzer
Format: VCD Value Change Dump

- **tracks the execution time of each function working as a common profiler for performance improvement.**
  - code optimization, bottleneck detection, and processing time measurements
- **Output format is read by gtkwave to view the signal transition and timing**

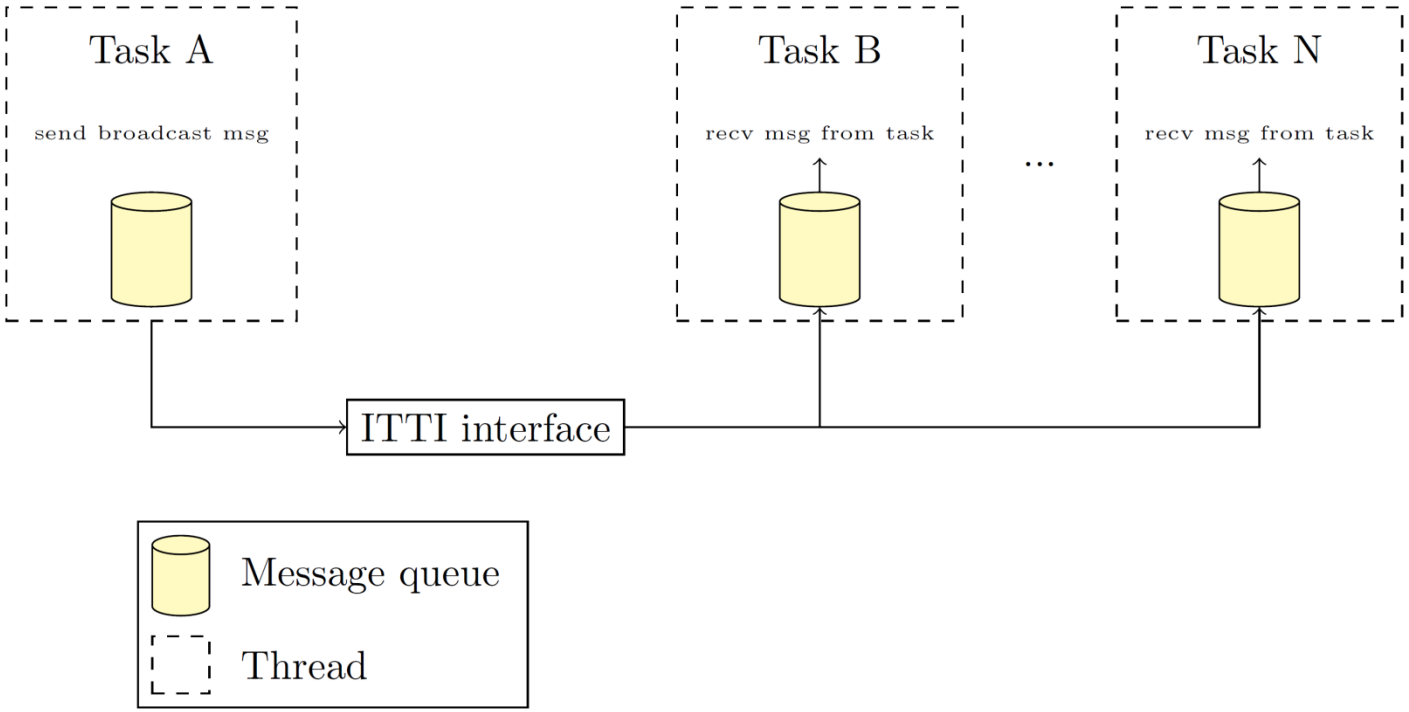


OAI time analyzer API
Format: VCD Value Change Dump

```
#include "UTIL/LOG/vcd_signal_dumper.h"
Main(){
VCD_SIGNAL_DUMPER_INIT("/tmp/openair_dump.vcd");
 VCD_SIGNAL_DUMPER_DUMP_VARIABLE_BY_NAME(VCD_SIGNAL_DUMPER_VARIABLES_HW_FRAME, frame);

VCD_SIGNAL_DUMPER_DUMP_FUNCTION_BY_NAME(VCD_SIGNAL_DUMPER_FUNCTIONS_ENB_DLSCH_ULSCH_SCHEDULER,VCD_FUN
CTION_IN);
…

VCD_SIGNAL_DUMPER_DUMP_FUNCTION_BY_NAME(VCD_SIGNAL_DUMPER_FUNCTIONS_ENB_DLSCH_ULSCH_SCHEDULER,VCD_FUN
CTION_OUT);


 VCD_SIGNAL_DUMPER_CLOSE();
}
```

- **Two type of signals**
  - variables
  - Functions
- **Used with gtkwave to view the signal transition and timing**
- **Source code**
  - Openair2/UTILS/LOG
- **To enable use option “-V”, then open with gtkwave and precnfigured file**
  - eNB_exmimo2.gtkw eNB_usrp.gtkw rrh.gtkw ue_exmimo2.gtkw

Inter-task interface (ITTI)



- **Intra-process communication system through async message passing**
- **Source code:**
  - common/utils/itti

Inter-task interface (ITTI)

- **Task = thread + Queue intra-process communication**
  - #define TASK_DEF(name of the task, priority , queue size)
- **Thread management, Task priority, Timer service**
- **Message definitions**

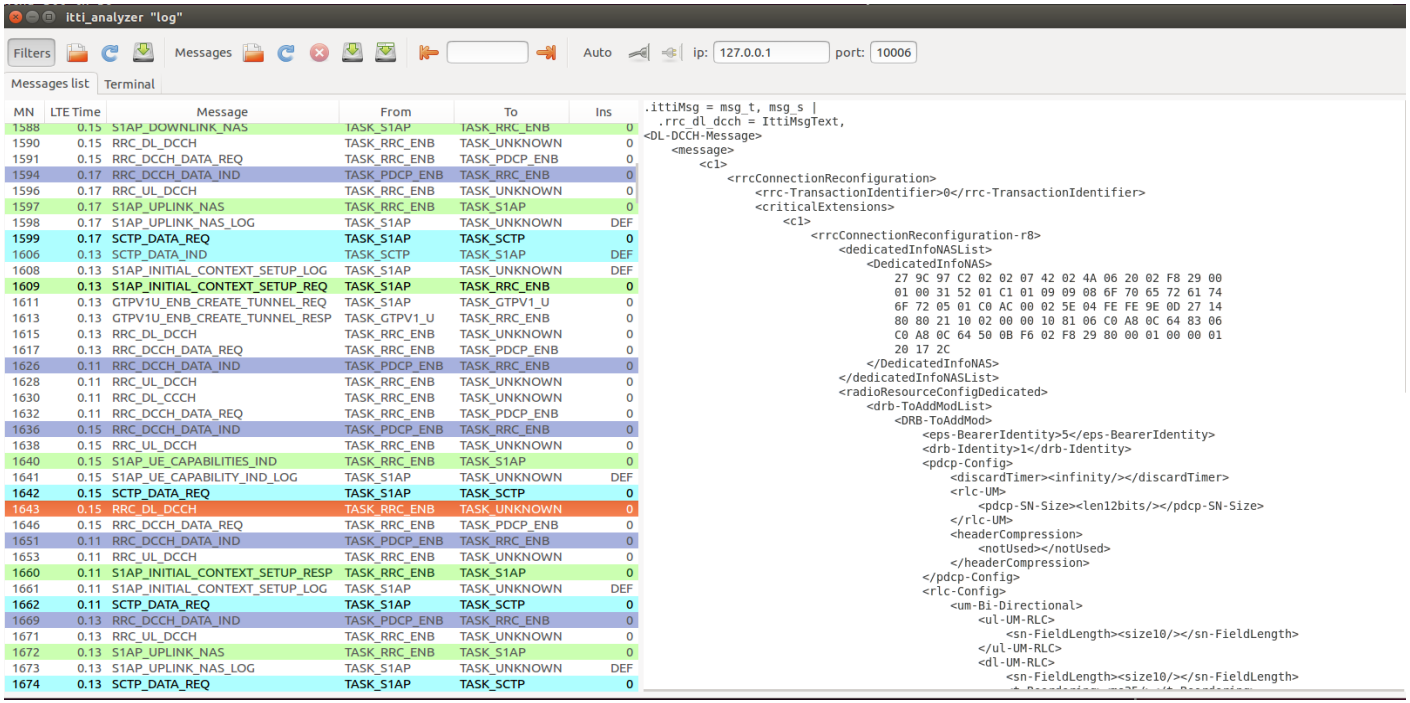
```
MESSAGE_DEF ( S1AP_SCTP_NEW_MESSAGE_IND ,
TASK_PRIORITY_MED , S1apSctpNewMessageInd
s1apSctpNewMessageInd )

Typedef struct {
uint8_t * buffer ; ///< SCTP buffer
3 uint32_t bufLen ; ///< SCTP buffer length
4 int32_t assocId ; ///< SCTP physical
} s1apSctpNewMessageInd
```

- **ITTI can wait for**
  - Messages
  - Timeout
  - External events such as sockets (FD)

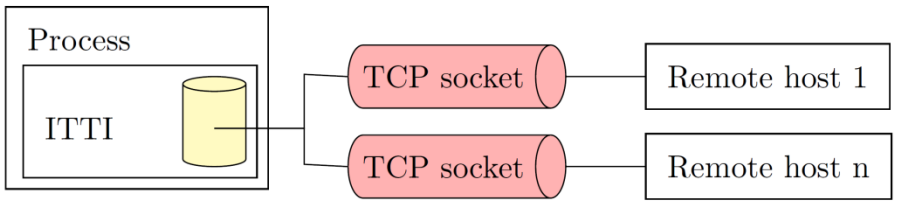
Task message handling

```
void * s1ap_mme_thread ( void * args ) {
while (1) {
  receive_msg ( TASK_S1AP , & receivedMessage );
 assert ( receivedMessage != NULL );
 switch ( receivedMessage -> messageId ) {
        case S1AP_SCTP_NEW_MESSAGE_IND :
         // Some processing
         break ;
        default :
         S1AP_DEBUG (" Unkwnon message ID %d\n", receivedMessage -> messageId );
         break ;
        }
 free ( receivedMessage );
 receivedMessage = NULL ;
 }
return NULL ;
}
```

ITTI Analyzer
Analyzing protocols PDU/SDU logs



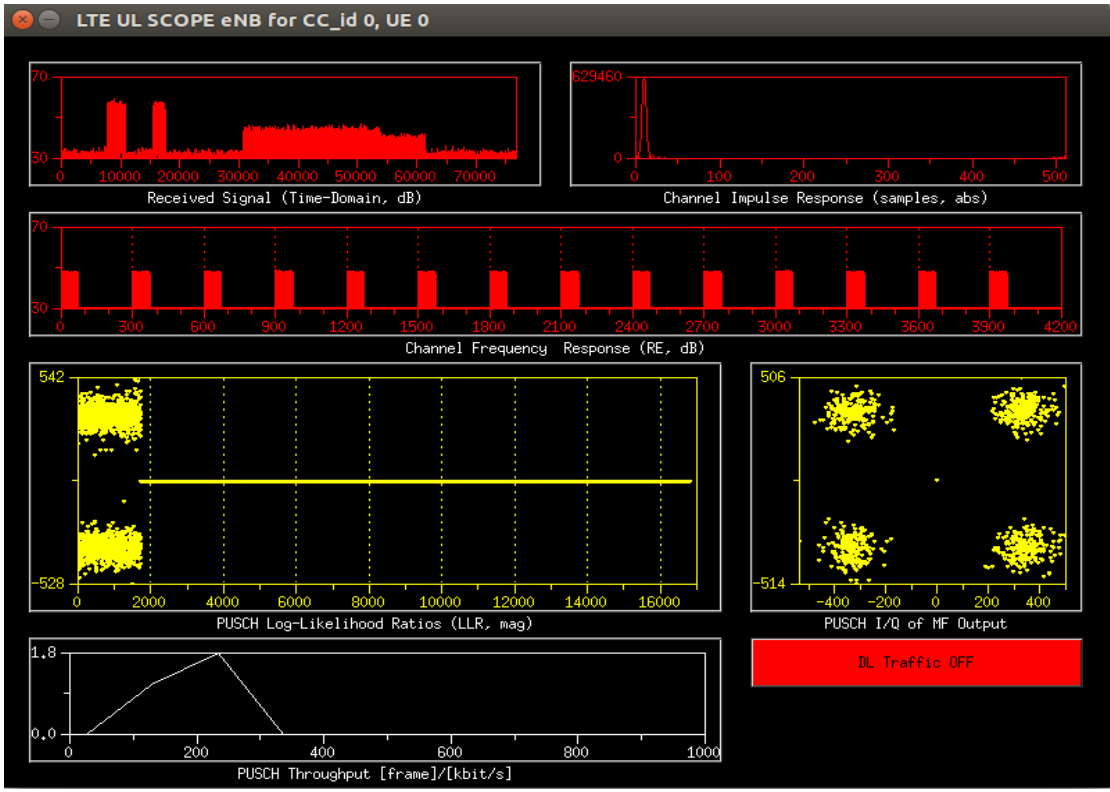
- **Complementary to Wireshark**
- **source file**
  - common/utils/itti_analyzer



Softscope

- **Monitor PHY layer for both eNB and UE**
- **The tool plots**
  - received signal power, channel impulse response, channel frequency response, channel frequency response, LLRs, throughput and I/Q components (e.g., 4-QAM constellation)

- **source file**
  - openair1/PHY/TOOLS



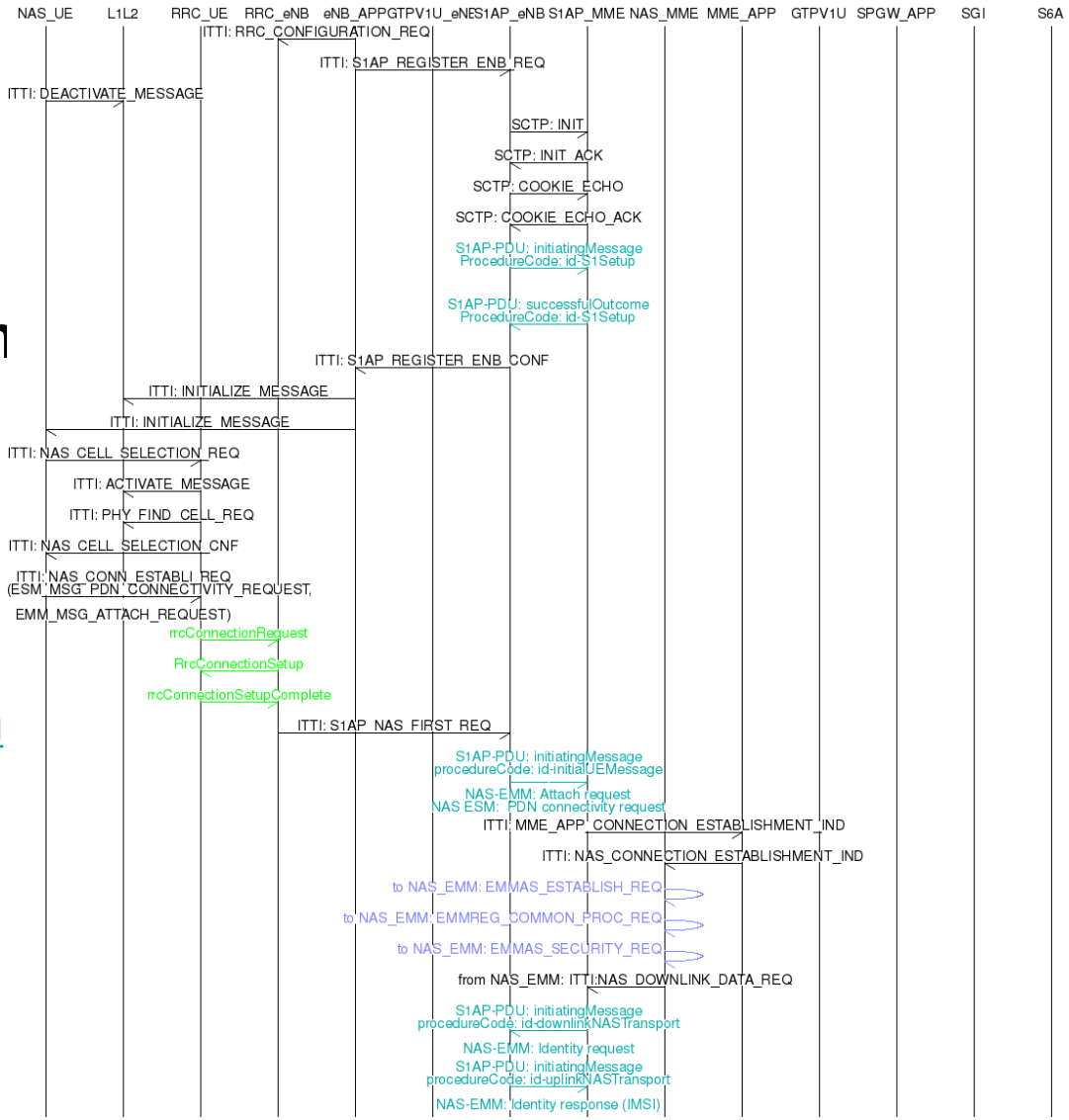
MAC/PHY statistics

- **Online statistics for the status of the network**
  - successful transmissions, errors per HARQ per round, average throughput, ULSCH/DLSCH errors per HARQ process (8 in LTE FDD) per round (4 is maximum).
- **Source file**
  - Openair1/PHY/LTE_TRANSPORT/print_stats.c
  - Openair2/LAYER2/openair2_proc.c

```
[eNB PROC] eNB 0/1 Frame 134: RX Gain 120 dB, I0 -97 dBm (23,0) dB
[eNB PROC] Subband I0:  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0

[eNB PROC] PERFORMANCE PARAMETERS
[eNB PROC] Total DLSCH bits successfully transmitted 12 kbits in 135 frame(s)
[eNB PROC] Total DLSCH average system throughput 0 kbps
[eNB PROC] Total DLSCH successful transmissions 25 in 135 frame(s)

[eNB PROC] UE 0 (c7b4) Power: (46,0) dB, Po_PUSCH: (-91,0) dBm, Po_PUCCH (-85/-108) dBm, Po_PUCCH1 (-96,-109) dBm,  PUCCH1 Thres -108 dBm
   harq 0: DL mcs 28, UL mcs 10, UL rb 6, delta_TF 269
   harq 1: DL mcs 22, UL mcs 9, UL rb 3, delta_TF 269
   harq 2: DL mcs 0, UL mcs 10, UL rb 6, delta_TF 269
   harq 3: DL mcs 0, UL mcs 10, UL rb 6, delta_TF 269
   harq 4: DL mcs 0, UL mcs 10, UL rb 6, delta_TF 269
   harq 5: DL mcs 0, UL mcs 10, UL rb 6, delta_TF 269
   harq 6: DL mcs 0, UL mcs 10, UL rb 6, delta_TF 269
   harq 7: DL mcs 0, UL mcs 10, UL rb 6, delta_TF 197
[eNB PROC] Wideband CQI: (0,0) dB
[eNB PROC] Subband CQI:     0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
[eNB PROC] DL TM 1, DL_cqi 15, DL_pmi_single 0
[eNB PROC] DL Subband CQI: 15 15 15 15 15 15 15  0  0  0  0  0  0
[eNB PROC] Timing advance 12 samples (3 16Ts), update -1
[eNB PROC] Mode = PUSCH(3)
[eNB PROC] UE_id_mac = 0, RRC status = 4
[eNB PROC] SR received/total: 3922/4139 (diff 217)
[eNB PROC] ULSCH errors/attempts per harq (per round):
   harq 0: 1/2068 (fer 0) (1/2068, 1/1, 1/1, 1/1)
   harq 1: 0/2070 (fer 0) (0/2070, 0/0, 0/0, 0/0)
   harq 2: 1/2068 (fer 0) (1/2068, 1/1, 1/1, 1/1)
   harq 3: 0/2070 (fer 0) (0/2070, 0/0, 0/0, 0/0)
   harq 4: 0/2070 (fer 0) (0/2070, 0/0, 0/0, 0/0)
   harq 5: 0/2071 (fer 0) (0/2071, 0/0, 0/0, 0/0)
   harq 6: 0/2071 (fer 0) (0/2071, 0/0, 0/0, 0/0)
   harq 7: 0/2071 (fer 0) (0/2071, 0/0, 0/0, 0/0)
[eNB PROC] ULSCH errors/attempts total 2/16559 (2/16559, 2/2, 2/2, 2/2):
[eNB PROC] DLSCH errors/attempts per harq (per round):
   harq 0: 0/19 (19/0/19, 0/0/0, 0/0/0, 0/0/0)
   harq 1: 0/6 (6/0/6, 0/0/0, 0/0/0, 0/0/0)
   harq 2: 0/0 (0/0/0, 0/0/0, 0/0/0, 0/0/0)
   harq 3: 0/0 (0/0/0, 0/0/0, 0/0/0, 0/0/0)
   harq 4: 0/0 (0/0/0, 0/0/0, 0/0/0, 0/0/0)
   harq 5: 0/0 (0/0/0, 0/0/0, 0/0/0, 0/0/0)
   harq 6: 0/0 (0/0/0, 0/0/0, 0/0/0, 0/0/0)
   harq 7: 0/0 (0/0/0, 0/0/0, 0/0/0, 0/0/0)
[eNB PROC] DLSCH errors/attempts total 0/25 (0/25, 0/0, 0/0, 0/0):
[eNB PROC] DLSCH total bits from MAC: 12kbit
[eNB PROC] DLSCH total bits ack'ed: 12kbit
[eNB PROC] DLSCH Average throughput (100 frames): 0kbps
[eNB PROC] Transmission Mode 1
[eNB PROC] RB Allocation on Sub-bands: 1 0 0 0 0 0 0
[eNB PROC] Total Number of Allocated PRBs = 2
```

Message Sequence Chart API

- **Represents the internal function calls across layers and entities in a form of chart**
- **It is an ITTI task**
- **Make use of MSC lib**
  - http://www.mcternan.me.uk/mscgen/

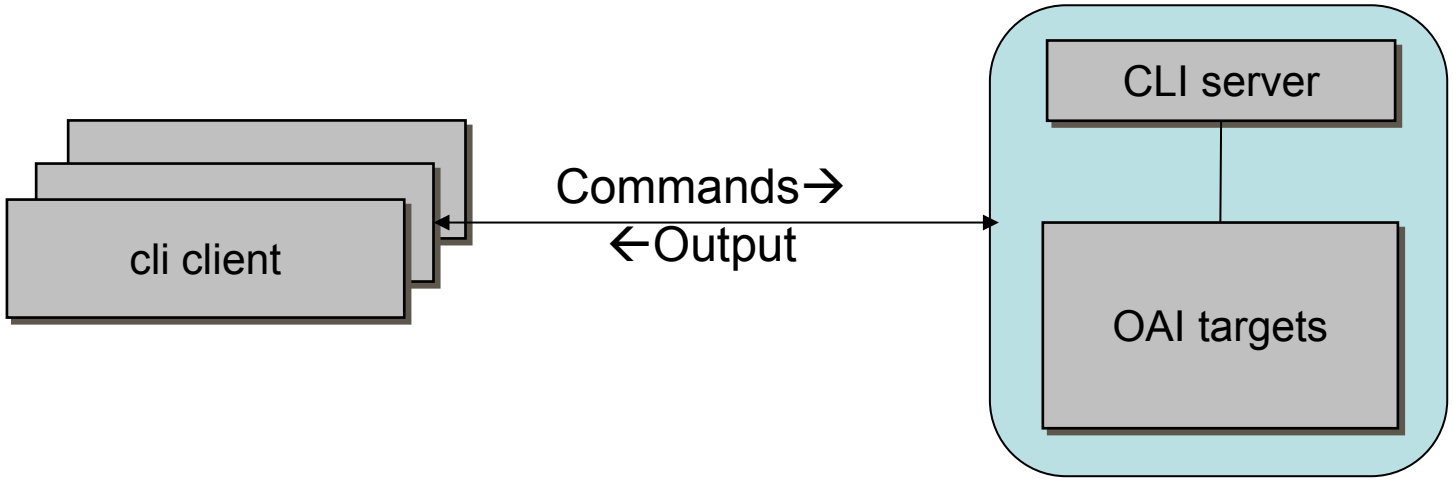


MSC API

- **MSC_LOG_EVENT(pROTO, fORMAT, aRGS...)**
  - Failure events, new UE attached, congestion, ….
- **MSC_LOG_RX_MESSAGE(rECEIVER, sENDER, bYTES, nUMbYTES, fORMAT, aRGS...)**
- **MSC_LOG_RX_DISCARDED_MESSAGE(rECEIVER, sENDER, bYTES, nUMbYTES, fORMAT, aRGS...)**
- **MSC_LOG_TX_MESSAGE(sENDER, rECEIVER, bYTES, nUMbYTES, fORMAT, aRGS...)**
- **MSC_LOG_TX_MESSAGE_FAILED(sENDER, rECEIVER, bYTES, nUMbYTES, fORMAT, aRGS...)**
- **Example**
  - MSC_LOG_TX_MESSAGE(MSC_S1AP_ENB, MSC_S1AP_MME,NULL,0,
    MSC_AS_TIME_FMT" S1AP_NAS_FIRST_REQ eNB %u UE %x",
    MSC_AS_TIME_ARGS(ctxt_pP),
    ctxt_pP->module_id,
    ctxt_pP->rnti);
- **Usage**
  - usage: msc_gen [-h] [--dir DIR] [--profile PROFILE] [--no_message NO_MESSAGE][--no_pdu NO_PDU] [--no_event NO_EVENT] [--type TYPE]
    - Dir:Directory where msc logs can be found
    - Profile : E-UTRAN, EPC
    - type: 'png', 'eps', 'svg' or 'ismap'
- **Source code**
  - Openair-cn/SRC/UTIL/MSC/ and openair-cn/SCRIPTS/msc_gen
  - Openairinterface5g/ommon/util/msc and openairinterface5g/targets/SCRIPTS/msc_gen

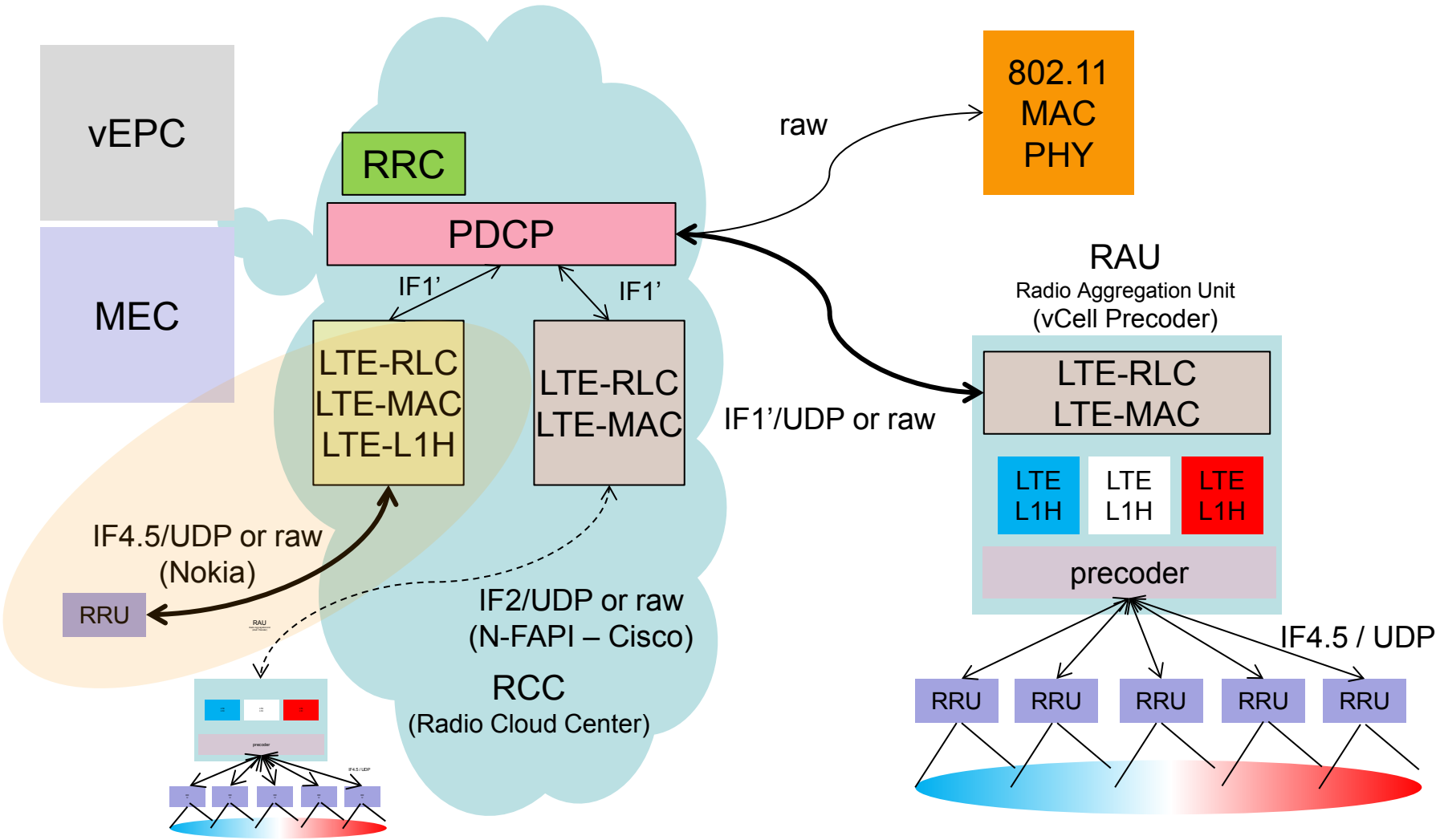
CLI

- **Allow interactive interface with the OAI**
  - Debugging
  - Monitoring
  - Configuration
- **Now only available in oaisim with limited commands**
- **Plans**
  - Extend the commands
  - Apply to all OAI targets: lte-softmodem, RRH



BACKUP

Splits under construction in OAI Community



Key Ingredients (How does OAI work)

- **Real-time extensions to Linux OS**
  - Today we rely on the lowlatency kernel provided by Ubuntu (since Ubuntu 14.04)
  - In earlier Ubuntu versions RTAI was used
- **Real-time data acquisition to/from PC**
  - ExpressMIMO uses DMA to transfer signals in and out of PC memory without hogging CPU -> very efficient
  - USRP transfers data over USB and therefore requires extra CPU time for (de-)packetization of signals
- **Highly optimized DSP routines running on Intel GPP**
  - Exploiting vector processing (SIMD)
  - 64-bit MMX $\rightarrow$ 128-bit SSE2/3/4 $\rightarrow$ 256-bit AVX2
  - OAI features fastest FFT and Turbo decoder of its kind
- **Multi-threaded parallel processing**

OSA Strategic Areas