Overview of OAI L1

- OAI RAN (openairinterface5g) Software Architecture
  - Functional entities
    - Focus today on L1 procedures for eNodeB
  - Real-time-scheduling
Current vRAN Roadmap in OAI

OAI Functional Splits

OSS/BSS/MEC

LTE/NR/NB PDCP

LTE MAC-RLC

LTE-L1

LTE MODEM

LTE/NR/NB RRC

LTE MAC-RLC

NR MAC-RLC

NR-L1

NR MODEM

NR RRC

LTE-L1

LTE MODEM

NB-IoT RRC

NB-IoT MAC-RLC

NR-L1

NR MODEM

Radio-Cloud Center (RCC)

Radio-Access Unit (RAU)

Remote Radio-Unit (RRU)

(c) Eurecom 2018
Current Functional Entities

- OAI currently implements the following entities in openairinterface5g
  - LTE-MODEM (eNB 36.211 OFDM modulation/demodulation)
  - LTE-L1 (eNB 36.211/212/213)
  - LTE-MACRLC (eNB 36.321/322)
  - LTE-PDCP (eNB PDCP/GTPU 36.323)
  - LTE-RRC (eNB RRC/SCTP 36.331)
- Each entity comprises
  - a northbound interface (backhaul/midhaul/fronthaul and configuration)
  - a southbound interface (midhaul/fronthaul and configuration)
  - one or two management interfaces
  - Three computing nodes
    - Radio Cloud Center (RCC): multiple RRC/PDCP entities
    - Radio-Access Unit (RAU): multiple MACRLC entities with medium-latency midhaul and L1 entities with low-latency fronthaul.
    - Remote Radio-Unit (RRU): Equipment at radio site. Varying degrees of processing elements depending on fronthaul/midhaul interface.
- Each entity has a configuration which is a local file or received via the management interface
- default interface between all entities is implemented using a UDP socket. Transport is configurable via a dynamically-loadable networking device
**RRU/RAU**

- **OAI current lower-layer functional split**
  - a network of radio units (L1-low)
  - a precoding function and switching function
  - Regular (virtualized) eNB functions

Example: RAU with NGFI_IF1pp xhaul (MAC/PHY split) northbound, NGFI_IF4p5 fronthaul southbound, 2 vCell logical interfaces (2 L1/L2 instances, or 1 L2 instance and 2 CCs), 4 RRUs with NGFI_IF4p5
OAI RAN (CU-DU) Architecture

L1 Instance 0

RU0

PHY TX \(n + 4\)

PHY RX \(n\)

RU1

PHY TX \(n + 4\)

PHY RX \(n\)

RU2

PHY TX \(n + 4\)

PHY RX \(n\)

RU3

PHY TX \(n + 4\)

PHY RX \(n\)

RU4

MAC/RLC/PDCP Instance 0

1ms TIEC

DCI0, Transport Blocks

DL/UL Scheduler

MAC TX

RLC TX

PDCP TX

PDCP RX

PDCP TX

PDCP RX

PDCP TX

PDCP RX

L1 Instance 1

RU5

PHY TX

PHY RX

RU6

MAC/RLC/PDCP Instance 1

1ms TIEC

DCI0, Transport Blocks

DL/UL Scheduler

MAC TX

RLC TX

PDCP TX

PDCP RX

PDCP TX

PDCP RX

PDCP TX

PDCP RX

(c) Eurecom 2018
RU and L1 Instances

- **Radio Unit (RU)** is
  - an entity managing a set of physical antennas. It can have a local RF unit or remote RF unit
  - performs precoding of multiple eNB TX streams and OFDM modulation (TX) and demodulation (RX) (part of 36.211)

- **L1 Instance** (indexed by Mod_id, or enb_mod_id) is a separate set of threads and contexts for the eNB/gNB procedures. There is one MAC/RLC entity associated to all :1 component carriers.

- **L1 Component Carrier** (indexed by CC_id) is
  - a software entity managing the L1 procedures (36.213,36.212,36.211) and can act on
    * sectored antenna component
    * Rel10+ component carrier
    * virtual cell for DAS or Massive-MIMO array
  - each L1 instance is managed by one or two threads which operate on a subframe (TX and RX) and can have a local RU or remote RU
  - if a remote radio unit the eNB performs the 36.213 specifications only (HARQ, etc.) and connects to the remainder via the IF2 midhaul interface.
RU and L1 Instances

- RU may have both an if_device for fronthaul and an rf_device for interconnection with a local RF unit.
- If the rf_device is absent, it must have a southbound fronthaul interface (either IF5 or IF4p5) depending on the local processing of the remote RU.
- If the if_device is absent, it must have a southbound RF interface and rf_device.
- Three types of L1 processing are performed by the RU:
  - Subset of common L1 procedures from 36.211 specifications
  - Fronthaul compression/decompression
  - Framing
- On TX:
  - A-law compression for (NGFI_RAU_IF4p5, NGFI_RAU_IF5)
  - A-law decompression (for NGFI_RRU_IF4p5 and NGFI_RRU_IF5)
  - OFDM modulation and cyclic prefix insertion (for NGFI_RRU_IF4p5, NGFI_RAU_IF5, 3GPP_eNodeB_BBU, 3GPP_eNodeB)
  - Precoding (for NGFI_RAU_IF5, NGFI_RAU_IF4p5, 3GPP_eNodeB_BBU, 3GPP_eNodeB)
RU and L1 Instances

- on RX
  - A-law compression for (NGFI_RRU_IF4p5, NGFI_RRU_IF5)
  - A-law decompression (for NGFI_RAU_IF4p5 and 3GPP_eNodeB_BBU)
  - cyclic prefix removal, frequency-shifting, OFDM demodulation, PRACH DFT (for NGFI_RRU_IF4p5, NGFI_RAU_IF5, 3GPP_eNodeB_BBU, 3GPP_eNodeB)

- On TX path
  - L1 instances/component carriers operate on a set of logical antenna ports (0-3 for TM1-6, 4 for eMBMS, 5 for TM7, 6 for positioning, 7-8 for TM8, etc.)
  - each L1 instance has a list of RUs and the logical antenna ports are mapped to the physical antennas attached to the RUs via the precoding function
RADIO SEGMENT (RU)
RU Procedures (monolithic eNB/gNB)
TX Precoding

- Spatio-temporal filtering for multi-cell (vCell) and multi-user transmission. Input and output are frequency-domain signals.
  
- can be applied to Rel-10/11/12/13 physical channels and Rel-8 common channels
  
  - UE-specific precoding (TM7-10)
  - vCell-specific precoding (PDCCH + TM1-6) for groups of UEs
  - PMCH vCells
  
- Precoding applicable to
  
  1. indoor DAS
  2. outdoor co-localized arrays (e.g., Massive-MIMO)
  3. outdoor CoMP
TX Precoding (monolithic eNB/gNB)
LAYER 1 HIGH SEGMENT (ENB/GNB)
Layer 1 TX per eNB instance

IF2 split points

eNodeB 3GPP or NGFI-LCC RF4+5

generate_pilots_sl

generate_pss/sss/p

36.211

CS-RS/PSS/SSS/PBCH REs

openair1/SCHED/phy-procedures_lte_enB.c:common-signal-procedures()

generate_phich_top()

PHICH REs

PHICH REs

generate_phich()

PDCCH REs

36.212, 36.211

pdcch_scrambling()

pdcch_modulation

pdcch_interleaving

36.211

PDSCH REs

generate_dci_top()

d1sch_scrambling()

d1sch_modulation()

36.212

openair1/SCHED/phy-procedures_lte_enB.c:dci_procedures()

d1sch_coding() 36.212

d1sch_encoding() 36.212

d1sch_scrambling()

d1sch_modulation() 36.211

[2-4 layers]

openair1/SCHED/phy-procedures_lte_enB.c:pdsch_procedures()

MCH_PDU

DLSCH_F

MAC

DLSCH_F

DLSCH_F

DCI_PDU

(c) Eurecom 2018
Layer 1 RX per eNB instance
RU/eNodeB OAI PUSCH Procedures

\[
R_{r,l} = \text{DFT}_{N_{\text{fft}}}(r_{r,l} \odot F_{7.5}), r = 0, 1, \ldots, R - 1, l = 0, 1, \ldots, N_{\text{symb}} - 1
\]

\[
R_{\text{ext},r,l}(n) = R_{r,l}(12\text{firstPRB} + n), n = 0, 1, \ldots, 12N_{\text{PRB}} - 1
\]

\[
\hat{H}_{r,l} = R_{\text{ext},r,l} \odot \text{DRS}^\dagger_2(\text{cyclicShift}, n_{\text{DMRS}}(2), n_{\text{PRS}}), (eNB\_pusch\_vars\rightarrow \text{ulsch}\_\text{xdataF}\_\text{ext})
\]

\[
R_{\text{comp},r,l} = \hat{H}_{r} \odot R_{\text{ext},r,l} 2^{-\log2 |H_{\text{max}}|}, \hat{H}_{r} = \frac{1}{2}(\hat{H}_{r,3} + \hat{H}_{r,10})
\]

\[
R_{\text{comp},0,l} = \frac{1}{R} \sum_{r=0}^{R-1} R_{\text{comp},r,l}
\]

\[
R_{\text{comp},0,l}(n) = R_{\text{comp},0,l}(n) \hat{Q}_8 \left( \frac{1}{|\hat{H}(n)|^2 + I_0} \right), \hat{H}(n) = \sum_{r=0}^{R-1} \hat{H}_{r}(n)
\]

\[
r_{\text{comp},0,l} = \text{IDFT}_{12N_{\text{PRB}}}(R_{\text{comp},0,l})
\]

QPSK: \[\lambda_l(2n) = \text{Re}(r_{\text{comp},0,l}(n)), \lambda_l(2n + 1) = \text{Im}(r_{\text{comp},0,l}(n))\]

16QAM: \[\lambda_l(4n) = \text{Re}(r_{\text{comp},0,l}(n)), \lambda_l(4n + 2) = \text{Im}(r_{\text{comp},0,l}(n))\]

\[\lambda_l(4n + 1) = |\text{Re}(r_{\text{comp},0,l}(n))| - 2|h(n)|, \lambda_l(4n + 3) = |\text{Im}(r_{\text{comp},0,l}(n))| - 2|h(n)|\]
RU/eNodeB PRACH procedures

- PRACH detection is a quasi-optimal non-coherent receiver for vector observations (multiple antennas)
- correlation is done in the frequency-domain, number of correlations (in the example above 2) depends on zeroCorrelationConfig configuration parameter
- peak-detection (for delay estimation) is performed in each NCS time-window
eNodeB PUCCH (1/1A/1B) Procedures

- PUCCH1 detection is a quasi-optimal non-coherent receiver (energy detector) for vector observations (multiple antennas) for scheduling request. Care is taken to handle residual frequency-offset.

- PUCCH1A/1B detection is quasi-coherent based on a rough channel estimate obtained on the 3 symbols without data modulation.

- In both cases, correlation is done in the frequency-domain
OAI IF1” – MAC/PHY Interface

- OAI IF1” is the interface between the 36.321 Medium-Access (MAC) Layer Procedures and the 36.213 Physical Layer Procedures. It links several PHY instances to one MAC instance.
- It is a configurable (dynamically loadable) module which can implement an (N)FAPI P5/P7 or a simpler interface.
OAI IF1” – MAC/PHY

- The PHY end uses three basic messages
  - CONFIG_REQ: this provides the cell configuration and UE-specific configuration to the PHY instances. This comprises the following FAPI P5/P7 messages
    1. CONFIG.request
    2. UE_CONFIGREQUEST
  - UL_INDICATION: This is an uplink indication that sends all UL information received in one TTI, including PRACH, if available. It also provides the subframe indication for the DL scheduler. It maps to the following FAPI P7 messages
    1. SUBFRAME.indication
    2. HARQ.indication
    3. CRC.indication
    4. RX_ULSCH.indication
    5. RX_SR.indication
    6. RX_CQI.indication
    7. RACH.indication
    8. SRS.indication
  - SCHEDULE_REQUEST: This message contains the scheduling response information and comprises the following FAPI P7 messages
    1. DL_CONFIG.request
    2. UL_CONFIG.request
    3. TX.request
    4. H1_DCI0.request

- The module is registered both by PHY and MAC and can implement different types of transport (NFAPI, function call, FAPI over UDP, etc.). During registration, function pointers for the different messages are provided for the module to interact with either PHY or MAC or both if they are executing in the same machine. Note that for a networked implementation (e.g. NFAPI), there are north and south components running in different machines.
OAI IF1” – MAC/PHY

- The PHY-layer timing is assumed to be
  1. wait for subframe indication \( n \) from HW
  2. trigger PRACH if \( n \) has PRACH (parallel thread)
  3. trigger UE specific RX procedures for \( n \) if \( n \) is UL
  4. assemble UL_INDICATION and send to MAC
  5. wait for SCHEDULE_REQUEST
  6. do TX procedures if \( n + 4 \) is TX and RX programming if \( n + 4 + k \) is UL

- The MAC-layer timing is assumed to be
  1. do all UL processing for subframe \( n \) if \( n \) is UL after unraveling of UL_INDICATION in MAC module
  2. wait for call to eNB_dlsch_ulsch_scheduler
  3. do DL scheduling for \( n + 4 \) if it is DL
  4. do UL scheduling for \( n + 8 \) if it is UL
  5. return from eNB_dlsch_ulsch_scheduler
  6. let MAC module form SCHEDULE_REQUEST
RU AND LAYER 1 PROCESS SCHEDULING
RU Process Scheduling

- Threads (all in targets/RT/USER/lte-ru.c)
  - `ru_thread`: Thread per RU which sequentially performs
    * read from south interface (RF or IF fronthaul)
    * RX processing for subframe \( n \) (if necessary). This can spawn a
      worker thread for OFDM demod (`fep_thread`)
    * wakeup L1s that are waiting for signal (if necessary). Note: this must
      block if subsequent TX processing is in the same thread
    * if \#cores \( \leq 4 \)
      - do RU TX processing for subframe \( n + 4 \) (if necessary). Note
        that this can spawn multiple worker threads for precoding and
        OFDM (`feptx_thread`)
      - send TX signal out of fronthaul interface
  - `ru_thread_tx`: per RU performing the TX portion above. This is used
    when \#cores > 4 and is triggered upon completion of L1_thread_tx.
  - `ru_thread_prach / ru_thread_prach_br`: Thread for PRACH
    processing in remote RU (DFT on RX, IF4p5 RRU). Up to 4 thread for
    PRACH BR (LTE-M) processing. One per CE level.
  - `ru_thread_asynch`: Thread for asynchronous reception from fronthaul
    interface (TX direction in RRU).
L1 Process Scheduling

- Threads (all in targets/RT/USER/lte-enb.c)
  - multi RX/TX thread mode (optional)
    - \texttt{L1\_thread\_rx}: threads per CC/Instance which does RX procedures for subframe $n$
    - \texttt{L1\_thread\_tx}: TX procedures for subframe $n+4$ (if $n+4$ is DL when TDD).
  - In the case of a common RU-L1 RX/TX thread (default if single RU/eNB with less than 4 processor cores)
    - \texttt{calls L1\_top}: procedure per CC/Instance which sequentially
      - blocks on signal from RU
      - RX/TX processing for subframe $n$ and $n+4$
      - return to \texttt{ru\_thread} (function)
  - \texttt{L1\_thread\_prach}: Thread per CC\_id/Instance for PRACH processing
  - \texttt{L1\_thread\_prach\_br}: Thread per CC\_id/Instance for PRACH BR (LTE-M) processing, one per CE level
Timing (1-4 cores, single RU/L1 entity)

\begin{figure}
\centering
\includegraphics[width=\textwidth]{timing_diagram.png}
\caption{Diagram showing timing for 1-4 cores with a single RU/L1 entity.}
\end{figure}
Timing (≥8 cores, 2 RU, 1 L1)

LTE HARQ periodicity (FDD, TDD can be longer)

ru_thread0
ru_thread1
fep_thread0
fep_thread1
l1_thread_rx
td_thread
l1_thread_tx
te_thread
ru_thread_tx0
ru_thread_tx1
feptx_thread0
feptx_thread1
ru_thread_prach
Multi-threading lower-layer operations

- **Front-end Processing (RU)**
  - Parallelizing even/odd slots in Fourier Transforms (TX and RX in RU)

- **Back-end Processing (L1)**
  - Parallelizing Segments in Turbo-encoder / Rate-Matching
  - Parallelizing Segments in Rate-Matching Inversion / Turbo-Decoder

- **Run worker threads in parallel to main thread in “single-thread” mode**
RU/L1 I/O and Processing Flow

- Piece of ru_t hread:

  // This is a forever while loop, it loops over subframes which are scheduled by incoming samples from HW devices
  while (!oai_exit) {
      ...

  // synchronization on input FH interface, acquire signals/data and block
  if (ru->fh_south_in) ru->fh_south_in(ru, &frame, &subframe);
  else AssertFatal(1==0, "No front haul interface at south port");
  if (((ru->do_prach>0) && (is_prach_subframe(fp, proc->frame_rx, proc->subframe_rx)==1)) {
      wakeup_prach_ru(ru);
  }
  else if (((ru->do_prach>0) && (is_prach_subframe(fp, proc->frame_rx, proc->subframe_rx)>1)) {  
      wakeup_prach_ru_br(ru);
  }

(c) Eurecom 2018
// do RX front-end processing (frequency-shift, dft) if needed
if (ru->feprx) ru->feprx(ru);

// At this point, all information for subframe has been received on FH interface

// wakeup all eNB processes waiting for this RU
if (ru->num_eNB>0) wakeup_eNBs(ru);
if(get_nprocs() <= 4 || ru->num_eNB==0){
    // do TX front-end processing if needed (precoding and/or IDFTs)
    if (ru->feptx_prec) ru->feptx_prec(ru);
    // do OFDM if needed
    if ((ru->fh_north_asynch_in == NULL) && (ru->feptx_ofdm))
        ru->feptx_ofdm(ru);
    if(!emulate_rf){
        // do outgoing front haul (south) if needed
        if ((ru->fh_north_asynch_in == NULL) && (ru->fh_south_out))
            ru->fh_south_out(ru);
        if (ru->fh_north_out) ru->fh_north_out(ru);
    }
}
Function to acquire samples

- From RF - r x r f
  - Either to HW or emulated for timing analysis of RU/L1 procedures
- From fronthaul receive function
  - f h_i f 5_sout h_i n (time-domain interface)
  - f h_i f 4p5_sout h_i n (frequency-domain interface)
 Inline function to wakeup RU component of prach procedures
  - Uses condition variable ru->proc. cond_prach
  - Thread code located in ru_thread_prach
    • Invokes frequency-domain translation only if RRU (rx_prach)
    • Invokes complete PRACH procedures if RAU

 _br is for LTE-M PRACH processing (up to 4 threads)
Top-level for RX front-end processing

- 7.5 kHz frequency-translation
- FFTs
- Processing is located in (single-thread) `openair/SC/ED/ruprocesses.c`
  * `fepfull()`: single-thread for processing of 1 subframe
  * `rup_fepfull2thread()`: 2 threads, one for each slot in the subframe
**wakeup_eNBs**

- Function to invoke the L1 entity procedures for subframe \( n \) (RX) and \( n+4 \) (TX)

- **If \# cores \( \leq 4 \)**
  - Call function `eNB_t_op` (targets/RT/USER/lte-enb.c)

- **If \# cores > 4**
  - Wakeup `eNB_thread_rx (ru->wakeup_rxtx)`
  - The is the function in targets/RT/USER/lte-enb.c: `wakeup_rxtx()`
    - Called by each RU that is attached to the L1 instance
    - When all RUs have called this function for a particular subframe, the L1 entity is woken up
Both initially call a function rxtx()
- Invokes full PRACH/PRACH_BR procedures (if RAU)
- phy_procedures_eNB_uespec_RX
  - UE-specific processing of subframe (UCI,SRS,ULSCH)
- If # cores >4 (in eNB_thread_rxtx)
  - Synchronization: wait for TX thread from previous subframe to complete (avoids race conditions)
- Run L2 procedures
  - Call UL_indication to send RX data up to L2
- if # cores <=4 (in eNB_top)
  - Run TX procedures for n+4
  - Wakeup TX FH thread to transmit to RF/fronthaul

If in eNB_thread_rxtx
- Wakeup eNB_thread_tx (wakeup_tx)
- wakeup_tx schedules tx_thread()
- tx_thread() does
  - TX procedures for n+4
  - Unlocks potentially waiting RX for n+1
  - Wakeup TX FH thread to transmit to RF/fronthaul