LTE Software eNodeB

Version: 2018-07-10

Table of Contents

1 Introduction

LTEENB is a LTE base station (eNodeB) implemented entirely in software and running on a PC. The PC generates a baseband signal which is sent to a radio front end doing the digital to analog conversion. The reverse is done for the reception.

LTEENB interfaces with a LTE Core Network thru the standard S1 interface. In particular the Amarisoft Core Network software (LTEMME) can easily be connected to it to build a highly configurable LTE test network.

LTEENB also supports NB-IoT cells.

2 Features

2.1 PHY layer

- LTE release 14 compliant.
- FDD and TDD configurations.
- Supported bandwidths: 1.4, 3, 5, 10, 15 and 20 MHz.
- Handle several cells in intra-band or inter-band configurations.
- Transmission modes: 1 (single antenna) and 2 to 10 (MIMO 4x2).
- Wideband CQI/PMI reports.
- HARQ support.
- Timing measurement thru the PRACH.
- Closed-loop UE power control.
- Frequency based MMSE equalizer.
- Highly optimized software turbo decoder.
- PAPR reduction support.
- Support of other radio heads can be added with an external shared library.
- Positioning Reference Signals (PRS) support.
- CSI-RS support.
- Multi-cluster PUSCH allocation.
- PUCCH 3 and PUCCH channel selection support.
- Carrier Aggregation support with cross carrier scheduling (tested with 3 DL channels, supports up to 5 DL channels).
- Mixed FDD-TDD Carrier Aggregation support.
- CoMP testing features (DMRS scrambling identity and QCL parameters can be selected).
- 256QAM DL support for PDSCH and MBMS.
- Support of release 11 TDD special subframe configurations 7 and 9.

2.2 Protocol layer

- LTE release 14 compliant.
- Implements the MAC, RLC, PDCP and RRC layers.
- Proportionally fair MAC scheduler with QoS support.
- Support of full and half duplex UEs.
- DRX support.
- Number of active users only limited by the available bandwidth.
- Fully configurable System Information Blocks.
- Integrity check and encryption using AES, Snow3G and ZUC algorithms.
- Support of RRC measurement with measurement gap.
- Supports intra eNodeB, S1 or X2 handovers.
- QoS support with user selectable DRB configuration for each QCI.
- ROHC support (RTP, UDP and IP v1 profiles).
- Public Warning System (CMAS/ETWS) support.
- MBMS support.

- Support of all ciphering and integrity protection algorithms including ZUC. Note that ciphering is subject to export rules depending on your country.
- Category 0 UE support.
- eDRX support.

2.3 LTE-M

- Release 14 compliant.
- Category M1 UE support.
- TM6 and TM9 support.
- FDD only (FDD and HD-FDD UEs are supported).
- Support of multiple CE levels (only CE-Mode A is supported).
- Support of message repetition for MPDCCH, PDSCH and PUSCH in full duplex. No message repetition for PUCCH and PRACH.
- Support of localized MPDCCH transmission.
- EPDCCH support for Category M1.
- No frequency hopping.
- Bandwidth must be >= 5 MHz for cells that also need to support Category 0 and above UEs.

2.4 NB-IoT

- NB-IoT release 14 compliant.
- Single-tone and multi-tone category NB1 and NB2 UE support.
- 15 kHz and 3.75 kHz subcarrier spacing are supported.
- All operation modes (in-band, guard band and standalone) are supported.
- Multiple NB-IoT and LTE cells can be used at the same time in the same eNodeB.
- Support of multiple coverage levels.
- Support of all NPDCCH, NPDSCH, NPUSCH and NPRACH configurations.
- Support of control plane CIoT optimization.
- Support of multi-DRB mode.

2.5 Downlink channel simulator

- Real time operation.
- High quality white Gaussian noise generator.
- Support the AWGN, EPA, EVA, ETU and MBSFN 3GPP channels.
- MIMO operation with the 3GPP correlation matrixes.
- User defined constant or Rayleigh paths with custom MIMO correlation matrixes.

2.6 Network interface

- Standard S1AP and GTP-U interfaces to the Core Network. Several PLMNs and S1 interfaces can be used simultaneously.
- X2AP interface between eNodeBs.
- M1 and M2 interfaces for MBMS.
- IPv6 support.

2.7 User interface

- Configurable logging system for all channels with built-in text decoders.
- Wireshark MAC-LTE capture.
- Plots for QAM constellations and channel response.
- Remote API using WebSocket.
- Command line monitor.
- Test commands to initiate handover and to dynamically change the power level of each cell.

3 Requirements

3.1 Hardware requirements

- A fast PC:
  - For best performances, a quad core Intel Core i5 or i7 CPU with AVX2 support (Haswell architecture or later) is recommended. Support of the SSE4.1 instruction set extension is required to run the software.
  - At least 2 Gigabit Ethernet ports.
  - At least 2 GB of RAM.
  - At least 1 GB of hard disk space.
  - The video adapter does not matter.
- Radio front end:
  - Amarisoft PCIe SDR
  - Ettus Research USRP N2x0 (SBX daughterboard). For MIMO 2x2, a second N2x0 with the SBX daughterboard and a USRP MIMO cable are needed.
  - Ettus Research USRP B2x0.
  - Ettus Research USRP X3x0.
  - Lime Microsystem LimeSDR
  - Nutaq PicoSDR 2x2.
- Appropriate antennas for the intended LTE frequencies or cables and attenuators to connect to a UE.
- Any commercial UE compatible with LTE FDD or TDD. All LTE FDD and TDD frequency bands are supported. If you use the Amarisoft Core Network, the device must accept test USIM cards (sim locked devices may not accept them).
- A test USIM card. Test USIM cards from Anritsu are supported by the default configuration. Other test USIM cards should work as well provided their IMSI and secret key are known.

3.2 Software requirements

- A 64 bit Linux distribution. Fedora 26 is the officially supported distribution.
  The following distributions are known as compatible:
  - Fedora 17 to 27
  - Cent OS 7
  - Ubuntu 12 to 16

  Other distributions can be used provided the radio frontend drivers are available for them.

- The Amarisoft LTE Core Network (another Core Network can be used, but we only explain here how to quickly set up the Amarisoft Core Network).

4 Off-The-Shelf package

If you ordered the OTS package, you don't need installation so you can skip next chapter.
When booting, MME end eNB are automatically started within a screen.
If you are not familiar with screen here is what you must know:

- To have access to consoles, log on the machine with root access, then type:

  ```
  screen -r
  ```

- To access MME monitor:

  ```
  CTRL-A + 0
  ```

- To access eNB monitor:

  ```
  CTRL-A + 1
  ```

- To exit screen:

  ```
  CTRL-A + d
  ```

5 Installation

The radio front end must be connected to one gigabit Ethernet port (don't use a switch to connect them to avoid potential packet losses). The other Ethernet port can be connected to the local network if necessary.

It is not recommended to run LTEENB in a virtual machine because it has hard real time constraints.

We also assume that you have some Linux and LTE knownledge.

5.1 Linux setup

LTEENB and LTEMME use the SCTP protocol for which the necessary packages are not usually installed. In order to install them, do as root user:

- Fedora

  ```
  yum install lksctp-tools kernel-modules-extra
  ```

- Ubuntu

  ```
  sudo apt-get install lksctp-tools linux-image-extra-3.13.0-24-generic
  ```

  Note that linux-image-extra package name mya differ depending on your kernel version.

and reboot the PC in case the Linux kernel was upgraded too.

As openssl libraries have no standard naming in Fedora and Cent OS, you need to create the following symbolic links on those distributions:

```
ln -s /usr/lib64/libcrypto.so.10 /usr/lib64/libcrypto.so.1.0.0

ln -s /usr/lib64/libssl.so.10 /usr/lib64/libssl.so.1.0.0
```

On recent Fedora distributions you might have to install the compat-openssl10 package:

```
sudo dnf install compat-openssl10
```

5.2 Linux setup for best performance

LTEENB requires a lot of CPU power and it has hard real time requirements (a maximum latency of 3 ms is required).

In order to get the lowest latency, it is recommended to set up the `performance` frequency governor for each CPU core. An example is included in the `lte_init.sh` script given with LTEENB.

Some buggy drivers are known to block the CPU during a few tens of ms. When it happens, LTEENB displays `UHD status: L=X U=Y S=Z`. One known problem is the DRM KMS cable polling. The script `lte_init.sh` disables it automatically.

Other drivers such as Wifi controllers can give the same problem. In order to avoid such problems, remove all unnecessary peripherals from the PC.

5.3 RRH setup

Please refer to sub section of your radio frontend to set it up.
When configured, you will have to select it (See [RRH selection], page 9).

5.3.1 Amarisoft PCIe SDR

Read the PCIe SDR documentation (`trx_sdr.pdf`).

5.3.2 Ettus Research USRP

Read the UHD Compatible RF frontends documentation (`trx_uhd.pdf`).

5.3.3 Lime Microsystems LimeSDR

In the dedicated package, see README file and execute install.sh script.

5.3.4 Nutaq PicoSDR 2x2

Nutaq PicoSDR is already preinstalled.

5.4 UE setup

Insert the test USIM card in the device.
Enable data connection and roaming in the configuration of your device.

With LTE, no Access Point Name (APN) is necessary because a default one is always provided by the network.
However, some UEs insists on having the same APN name as the MME to enable IP connectivity.

As a result, start by removing any APN stored in the UE and manualy add APN (Only APN name is required) as defined in MME configuration file (Default is test123).

If possible, disable 2G (GSM) and 3G (WCDMA) access to have only LTE access system in order to speed up the network search.

If possible, limit the LTE frequency bands used by the device to the one you want to use in order to speed up the network search.

5.5 LTEMME installation

Decompress the LTEMME archive to a convenient place. The executable `ltemme` can be launched from this directory.

5.5.1 Basic LTEMME configuration

The main configuration file is `config/mme.cfg`. It uses a superset of the JSON syntax.

If your USIM card has different parameters from Anritsu, you need to put its IMSI and secret key in the `ue_db` section.

You also need to change the DNS address that will be given to the UE to match the DNS address of your local network (parameter `dns_addr`).

5.5.2 License key installation

LTEMME needs a license key file to run. *It is associated to your PC, so if you replace it or change its hardware configuration you must contact Amarisoft to get a new license key.*

The following steps are needed to get this license file:

- Run LTEMME:

  ```
  ./ltemme config/mme.cfg
  ```

  It says that the license key is not present and prints a 16 digit hexadecimal code.
- Send by mail to `delivery@amarisoft.com` this hexadecimal code to your contact at Amarisoft. You will get back the `ltemme.key` license key file.
- Copy the `ltemme.key` file to the `${HOME}/.amarisoft/` directory (`${HOME}` is the home directory of the `root` user). You can use the shell variable `AMARISOFT_PATH` to change this path.

Once the license key is installed, `ltemme` should start normally.

5.6 LTEENB installation

Decompress the LTEENB archive to a convenient place. The executable `lteenb` can be launched from this directory.

5.6.1 Basic LTEENB configuration

The main configuration file is `config/enb.cfg`. It uses a superset of the JSON syntax.

The default setup is for a 10 MHz LTE eNodeB. The main parameter you need to change right now is the actual frequency you want to use. You need to be sure that no interference is present on the frequency you are using and that you have the legal right to use it (in most countries it is illegal to transmit on LTE frequency bands without an explicit authorization).

In order to reduce interferences, it is recommended to use a high frequency band such as the 2600 MHz band (band 7) in Europe.

The parameter `dl_earfcn` gives the EARFCN of the center frequency for the downlink. The corresponding uplink frequency is automatically choosen. The center frequency can be arbitrarily choosen provided the transmitted spectrum fully lies inside the choosen band. So if `f` is the center frequency, `B` the LTE bandwidth, `f_min` and `f_max` the band limits, the following relation must hold:

```
f_min + 0.5 * B <= f <= f_max - 0.5 * B
```

The EARFCN is the frequency expressed in 100 kHz units starting from an offset depending on the selected band. You can have the band parameters and do convertions between EARFCNs and frequencies at `http://niviuk.free.fr/lte_band.php` or by looking at the section 5.7.3 of 3GPP TS 36.101.

5.6.2 RRH selection

To select appropriate RF frontend to use, please execute following command:

```
./config/rf_select.sh <type>
```

Where `type` is your frontend type:

- sdr
- n2x0
- b2x0
- x3x0
- sdr
- nutaq

NB: you can lanch following command to see available frontends:

```
./config/rf_select.sh
```

5.6.3 License key installation

LTEENB needs a license key file to run. *It is associated to your PC, so if you replace it or change its hardware configuration you must contact Amarisoft to get a new license key.*

The following steps are needed to get this license file:

- Run LTEENB:

  ```
  ./lteenb config/enb.cfg
  ```

  It says that the license key is not present and prints a 16 digit hexadecimal code.
- Send by mail this hexadecimal code to your contact at Amarisoft. You will get back the `lteenb.key` license key file.

- Copy the `lteenb.key` file to the `${HOME}/.amarisoft/` directory (`${HOME}` is the home directory of the `root` user). You can use the shell variable `AMARISOFT_PATH` to change this path.

Once the license key is installed, `lteenb` should start normally.

5.7 Initial testing

Customize and start the `lte_init.sh` script as `root` user to configure the network and CPU governors.

Start the LTEMME software as `root` user. `root` priviledges are needed to set up the virtual network interface.

```
./ltemme config/mme.cfg
```

In another terminal, start the LTEENB software as `root` user. `root` priviledges are needed to use real time scheduling priority.

```
./lteenb config/enb.cfg
```

The base station is now running. Type `s1` in the command line monitor of LTEENB to verify that it is connected to LTEMME. If it is the case, type `t` to enable the MAC traces (the traces are automatically disabled once you press return).

Turn on the UE and keep it at a few meters of the eNodeB antenna. It starts scanning the LTE bandwidth. After a few minutes, it should detect the eNodeB signal and transmit its first PRACH signal to the base station. You should get a trace like:

```
PRACH: cell=01 seq=X ta=Y snr=Z dB
```

Then the UE will *attach* to the simulated LTE core network and get its IP address. If it works, then the device will indicate it is connected.

The core network maintains a persistent database storing all the parameters of the configured UEs. It is by default in `config/lte_ue.db`. If the initial connection is OK, you can get the IP address of the UE from this file. You can then try to ping it from the PC.

Then if the local network is correctly configured on the PC, the UE can access to your local network (and internet if your local network allows it).

6 Troubleshooting

There are many parameters in an LTE setup, so there are many cause of problems. Here are a few ones we noticed during our tests:

6.1 LTEENB does not start

LTEENB must be launched as root so that it can use real time scheduling.

If some librairies needed by LTEENB are not present, it means you need to use another Linux distribution or to upgrade it.

6.2 UHD library ABI compatibility mismatch

If you get this kind of message while starting *lteenb*:

```
'trx_uhd_3.4.0.so' does not exist
```

It means that current UHD version on your system is not supported.
Follow `trx_uhd.pdf` documentation to install proper version.

6.3 The license key file is not correct

You need to contact Amarisoft to get a valid license key and/or a USB dongle.

6.4 Many messages 'UHD status: L=X U=Y S=Z' are displayed.

These messages indicate that there are underflows or overflows errors when communicating with the USRP device. The most likely explanation is that not enough CPU time is available. You can launch

```
top -H
```

To see which processes and threads use the CPU time. Normally only 2 LTEENB threads should use about 50% of one CPU core time each when the eNodeB is idle. The following can be done to help:

- Remove unnecessary drivers and peripherals. See [Linux setup for best performance], page 7.
- Launch `lteenb` as `root` so that it can use real time scheduling.
- Don't connect the USRP to the PC thru a switch to avoid packet losses.
- Ensure that your CPU is fast enough. It should be at least a 4 core i5 or i7 Haswell architecture or later.
- If your CPU is too slow, consider using a smaller LTE bandwidth (the CPU load is proportional to the bandwidth).
- If the errors happen during high traffic, be sure you have disabled the `debug` log output (`log_options` option). It generates a lot of data and takes some CPU time.
- If the errors happen during high traffic, consider limiting the eNodeB uplink bit rate. See [CPU load limitation], page 14.

For best performance, you can also remove unused daemons or cron jobs, in particular:

- The automatic upgrade done by PackageKit or similar.
- The various scripts in `/etc/cron.{hourly|daily|weekly}` which take some time and which are not strictly needed: `mlocate`, `tmpwatch`, `man-db`, `prelink`.

If the PC is only used as server, it is better to boot the PC in text mode by default. If `systemd` is used by your Linux distribution (Fedora), it is done by changing the

link `/etc/systemd/system/default.target` to `/lib/systemd/system/runlevel3.target`. If `init` is used, it is done by modifying `/etc/inittab` to use the run level 3 as the default run level:

```
id:3:initdefault:
```

6.5 The initial PRACH signal is not received.

This is the most critical step. If you don't get the initial PRACH signal, it indicates that something is wrong in your configuration. Here are a few important points:

- Check that your EARFCN is correct and in a band without interference. Warning: the EARFCN corresponds to the frequency of the *center* of the bandwidth. Use a cable and an attenuator if you want to avoid interfering with another network or if you don't have the authorization to transmit on the corresponding frequency.
- Check that your UE is correctly configured: LTE must be enabled on the right frequency band.
- The UE may not get a good enough signal or may saturate. Try to move it closer or further from the base station antenna. Some UE have better performance with some frequency bands, so try another frequency band supported by the UE.
- Only if you have problems after changing the frequency band, make sure that you use a different `cell_id` in `enb.cfg`. The UE memorizes the last parameters and won't search another frequency if the Cell Identity is not modified.

If none of the previous point helps, consider trying another frequency band (change the EARFCN and update `cell_id` in `enb.cfg`).

6.6 The initial PRACH is received, but the UE is never attached.

Look at the `/tmp/enb0.log` log file. There can be several problems. The normal steps are RRC connection, then NAS messages are exchanged to attach the UE and authenticate it.

The possible problems are:

- Radio problems. In this case, no NAS messages are seen in the logs. Try to lower the TX gain of the radio frontend ([tx_gain], page 28, parameter) to reduce the TX/RX interferences due to the lack of proper duplexer.
- Invalid IMSI configured. In this case, the dialog stops after the NAS attach request message.
- SIM authentication error. In this case, the dialog ends in the Authentication request/Authentication response NAS dialog. It means you don't have the correct secret key configured.
- Security configuration error. In the case, the dialog stops after the NAS security mode command. It indicates that the UE does not accept to be configured without integrity check and encryption. You need to find another UE which is more tolerant (we never hit this case, but it might happen).
- Immediate NAS detach after NAS attach complete. In this case, the UE does not accept something in the network configuration. It can happen if it requested IPv6 (we only support IPv4 in the default configuration) or if the APN is not correct. It can also happen because of an invalid UE database in the core network emulation. Try to turn off and on the device several times to see if the problem persists (there are various timeouts and number of NAS attempts which can be triggered and solve the issue).
- Unexpected `PDN connectivity request`. In this case, you see `PDN connectivity request` and `PDN connectivity reject` in the logs. Remove any explicit Access Point Name (APN)

in the UE configuration (the core network emulation only supports configuring a default PDN in the initial attach).

6.7 The initial attach is OK but `ping` is not working.

If you get here, the device indicates that the LTE/4G connection is up but the ping to the UE from the PC does not work. You can also try a ping from the device to the PC (the PC can be pinged on 192.168.3.1).

Radio problems can still be the explication if the radio conditions are too bad. The symptom of this case is that you see many PRACH signals coming from the UE. Try to reduce the TX power with the [tx_gain], page 28, parameter.

Another explanation can be that the UE does not accept roaming. Try to enable it on the UE.

Another explanation can be bugs in the UE (or its PC driver if it is a USB dongle) in case you changed the LTE configuration (we noticed it in some cases). The symptom is that the IP packets are truncated when doing `tcpdump` on the UE side. In case of doubt, just turn off and on the UE (and the corresponding PC if it is a USB dongle) to start from a clean state.

6.8 The ping is working but no Internet access is possible from the UE.

The most likely explanation is that the IP forwarding/masquerading is not configured correctly on the PC. You need to look at the IP table configuration (/sbin/iptables -n -v -L) and correct it if the `lte_init.sh` configuration is not enough. Use of `tcpdump` or `wireshark` on the different interfaces can help to locate the problem.

Another possibility is that the DNS address given to the UE is not correct (try to ping using IP address instead of host names).

Verify that roaming is activated on the UE. The UE may not accept roaming and avoid IP access even if the ping is working.

7 Advanced Configuration

7.1 Logging

The eNodeB and the Core Network can output the messages of all the layers to log files. See the `log_options` option to select the layer to output and the level of verbosity. The log filenames are defined with the `log_filename` option.

You can also use Wireshark to monitor:

- S1 (S1AP with embedded NAS, GTP-U) link the eNodeB and the Core Network.
- M2 (M2AP, GTP-U) link between the eNodeB and MBMS Gateway.
- X2 (X2AP) link between eNodeBs.

7.2 Changing the LTE bandwidth

It is configured with the `n_rb_dl` parameter giving the number of resource blocks. Use `n_rb_dl` = 6, 15, 25, 50, 75, 100 for the bandwiths 1.4, 3, 5, 10, 15, 20 MHz.

The SIB2 configuration should also be modified in `sib_sched_list`. Preconfigured files are available for the available bandwidths: `sib23_rb6.asn`, `sib23_rb15.asn`, `sib23_rb25.asn`, `sib23_rb50.asn`, `sib23_rb75.asn` and `sib23_rb100.asn`.

Notes:

- The 1.4 MHz bandwidth needs a specific scheduler configuration, so use the configuration file `enb-1_4mhz.cfg` instead.
- The CPU load is proportional to the LTE bandwidth.
- Not all LTE bands allow all LTE bandwidths. For example, bands 7 (2.6 GHz) and band 20 (Europe 800 EDD) do not support the bandwidths of 1.4 and 3 MHz.

7.3 CPU load limitation

In order to avoid using too much CPU time, it is possible to limit the uplink MCS (`pusch_max_mcs`) and the number of iterations of the turbo decoder (`pusch_max_its`). Normally it is only critical for the larger LTE bandwidths (20 MHz). The symptom of too high CPU use are many messages `UHD status: L=X U=Y S=Z`.

7.4 UE connection traces

By default or when using the `t` monitor command, the eNodeB displays the status of the connection with the UEs. It stops displaying them when you press return.

```
                    ----DL----------------------- --UL----------------------------------------
     UE_ID CL RNTI C cqi ri  mcs retx txok brate  snr puc1  mcs rxko rxok brate    turbo phr  pl
         1 01 003d 1  12  1 21.1    0  361 1.77M  9.1  8.5 14.7    3  779 2.06M 1/3.5/10  10 100
```

`UE_ID` S1 eNodeB UE identity, unique among all cells.

`CL` Lower 7, 8 or 10 bits of the cell identity (hexadecimal).

`RNTI` C-RNTI of the UE (hexadecimal).

`C` Number of aggregated DL cells.

`cqi` Channel Quality Indicator, between 0 (bad) and 15 (very good). If there are several aggregated DL cells, the minimum cqi is displayed.

`ri` Rank Indicator (number of layers for MIMO). If there are several aggregated DL cells, the minimum rank indicator is displayed.

`mcs` Average Modulation and Coding Scheme.

`retx` Number of transport block retransmissions.

`txok` Number of successfully transmitted transport blocks.

`brate` Average bitrate (at the MAC layer), in bits per second.

`snr` is the measured Signal to Noise Ratio for the uplink from the PUSCH reference signals and the SRS.

`puc1` is the measured Signal to Noise Ratio for the last PUCCH1.

`rxko` Number of received uplink transport blocks with CRC errors.

`rxok` Number of received uplink transport blocks without CRC error.

`turbo` gives the minimum, average and maximum number of iterations of the turbo decoder.

`phr` is the content of the last Power Headroom MAC control element sent by the UE. It is expressed in dB. Negative values indicate that the UE could not transmit with the required power.

`pl` Uplink Path Loss in dB. It is measured from the reported PHR and the measured uplink power level. It is meaningful only if the RF interface correctly reports the absolute received power level.

7.5 UE Power control

The eNodeB does dynamic UE power control (see the `dpc`) option. However, it is better to have a good initial UE power to avoid retransmissions or interferences. So it is important to correctly set the various power settings in the SIBs. In particular, to avoid generating too much interference and to limit the battery drain, the SIB1 `p-Max` parameter (maximum allowed power for the UE in dBm) should be set to a low enough value (a few dBm).

It is also useful to tune the value of `referenceSignalPower` (power per carrier of the reference signal in dBm) in SIB2 if the RF interface does not provides its transmit power thru the TRX driver. It is used by the UE to compute the path loss and to adjust its own transmit power.

7.6 Multi-cell support

The eNodeB can run several LTE cells. The cells can be configured individually and share the same S1 interface with the Core Network.

7.6.1 Intra-band multi-cell

A configuration example is given in `config/multi_cell.cfg` for two adjacent 5 MHz cells in band 7.

The monitor command `cell_gain` changes the relative DL power for a given cell. For example, use

```
cell_gain 1 -100
```

to mute the first cell (cell_id = 1). Use:

```
cell_gain 1 0
```

to restore the default output power.

Constraints:

- The full transmitted spectrum must lie inside the maximum output bandwidth permitted by the radio head (hence 40 MHz for the USRP N2x0 and 56 MHz for PCIe SDR card).

- The difference of the center frequencies of each cell must be a multiple of 300 kHz (hence the difference of their EARFCN must be a multiple of 3).
- The difference between the center frequency of each cell and the average of center frequencies must be a multiple of 15 kHz.
- The number of cells that could be configured in a frequency band depends on the total bandwidth of the lte band and the configured bandwidth of each cell + the offsets.
- The cells must have the same `prach-ConfigIndex` (SIB2), i.e. their PRACH must have the same duration and transmitted in the same subframes.
- Multiple cells can be set at the same frequency provided their physical cell identity (`n_id_cell` property) and PRACH `rootSequenceIndex` (SIB2) are different to minimize the inter-cell interferences. In the current version, there is no resource reservation among the cells, so a performance degradation happens if they transmit at the same time in the same resource blocks. So it is currently better to use cells at different frequencies.

Let's take the following example to configure 3 cells in band 7:

```
cell 1 DL frequency: 2627 MHz
cell 2 DL frequency: 2642 MHz
cell 3 DL frequency: 2657 MHz
```

$$\texttt{average\_dl\_freq} = (2627 + 2642 + 2657)/3 = 2642 \text{ MHz}$$

$$\texttt{cell1\_freq\_offset} = 2627 - 2642 = -15 \text{ MHz}$$

$$\texttt{cell2\_freq\_offset} = 2642 - 2642 = 0 \text{ MHz}$$

$$\texttt{cell3\_freq\_offset} = 2657 - 2642 = 15 \text{ MHz}$$

```
cell 1 DL EARFCN: 2820
cell 2 DL EARFCN: 2970
cell 3 DL EARFCN: 3120
```

$$\texttt{cell1\_cell2\_earfcn\_offset} = 2820 - 2970 = -150$$

$$\texttt{cell1\_cell3\_earfcn\_offset} = 2820 - 3120 = -300$$

$$\texttt{cell3\_cell2\_earfcn\_offset} = 3120 - 2970 = 150$$

We can observe that the difference between the the center frequency of each cell and the average of center frequencies is indeed a multiple of 15 kHz and the difference between the DL EARFCNs are a multiple of 3.

7.6.2 Inter-band multi-cell

A configuration example is given in `config/enb-ca.cfg` for two SISO 5 MHz cells in bands 2 and 4. This configuration could be tested with:

- Two URSP N210 connected with a MIMO cable
- Two PCIe SDR cards synchronized with a USB cable

This configuration also enables carrier aggregation for release 10 UEs.

7.7 Handover support

In the current version, handover is supported as an experimental feature. Intra-eNodeB, S1 or X2 handovers are supported. The handover can be manually initiated with the `handover` monitor command or automatically initiated based on UE measurement.

Test case for intra-eNodeB handover:

1. Start lteenb with the multi cell configuration `config/multi_cell.cfg`.
2. Wait until the UE connects to a cell (use the `t` command to active the MAC traces or use the monitor command `ue` to list the connected UEs).

3. Start a long network transfer or a ping to the UE.
4. Reduce the power by 10 dB on the serving cell. If the UE is on cell 1:

   ```
   cell_gain 1 -10
   ```

   After some time the UE will make a handover to cell 2 (check it with the `t` command by looking at the `CL` column).
5. Increase the power of cell 1 and reduce the power of cell 2:

   ```
   cell_gain 1 0 ; cell_gain 2 -10
   ```

   After some time the UE will make a handover to cell 1.
6. You can also force a handover with the `handover` monitor command by giving the UE ID and the Physical Cell Identity (and optionally the DL EARFCN) of the target cell.

The behavior is similar with S1 and X2 handovers, but at least two eNodeBs (and two radio heads) are needed to use it.

In all the cases, it is important to have a valid neighbour cell configuration for each cell (`ncell_list` property) so that the source eNodeB can deduce target the Cell Identity from the target Physical Cell Identity and DL EARFCN.

7.8 MIMO support

MIMO is currently supported with following radio frontends:

- USRP N2x0 with their SBX daughterboards (Two device are needed for MIMO).
- USRP B2x0.
- USRP X3x0 (Only up to 10Mhz bandwidth with gigabit Ethernet link).
- PCIe SDR.
- Nutaq PicoSDR 2x2.

The configuration `mimo-2x2-5mhz.cfg` demonstrates a 5 MHz MIMO configuration using transmission mode 3 (large delay CDD).

The configuration `mimo-2x2-20mhz.cfg` demonstrates a 20 MHz MIMO configuration with transmission mode 3.

If you use N2x0 with this configuration, please note that there is only one N2x0 connected by ethernet (second is using MIMO cable).

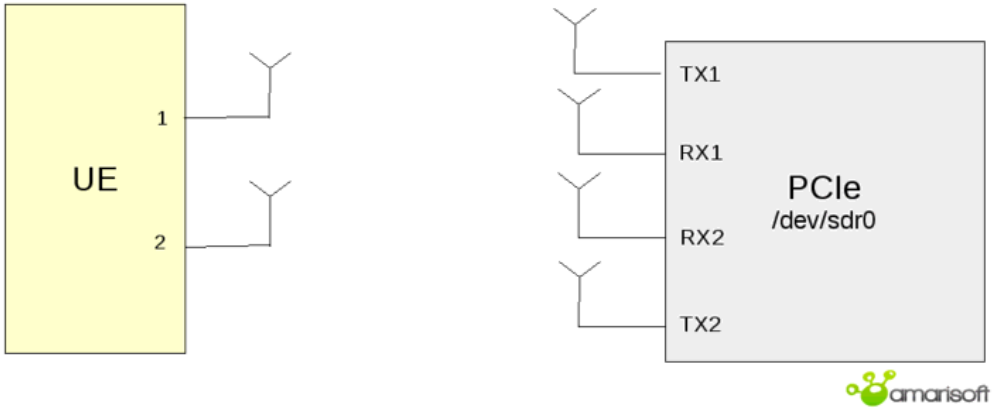
Note: the current UHD driver uses a lot of CPU time when MIMO is enabled. If it is an issue in your tests, follow patched version installation inside your `trx_uhd.pdf` documentation.

7.9 MIMO environment setup

7.9.1 Over the air

7.9.1.1 PCIe SDR setup

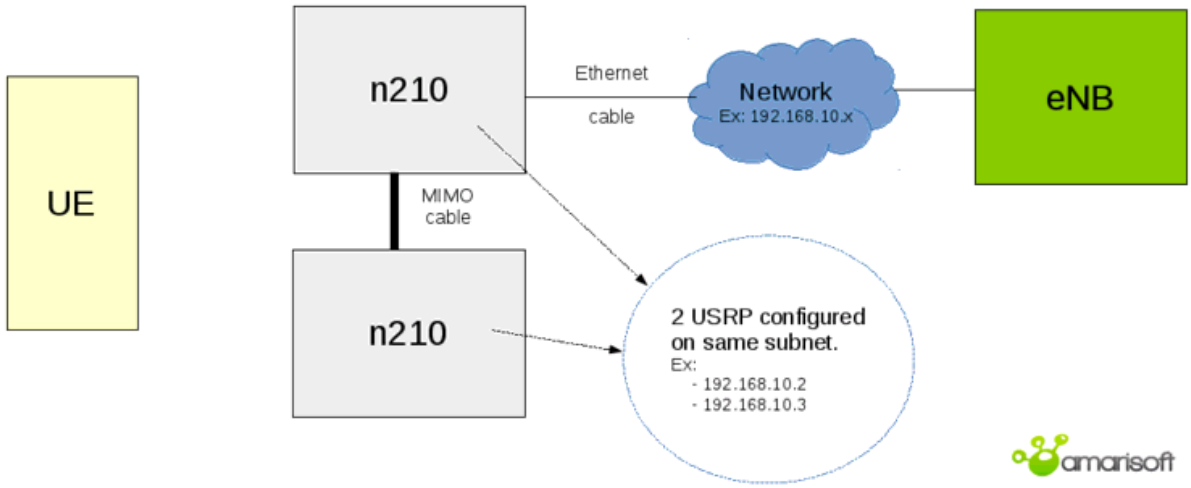
The following diagram depicts how to set up your MIMO environment with antennas using one single PCIe SDR card in FDD. You can simplify the setup by removing the antenna connected to RX2 if there is no MIMO in UL.



For TDD mode, you only need to connect antennas on the TX1 and TX2 connectors.

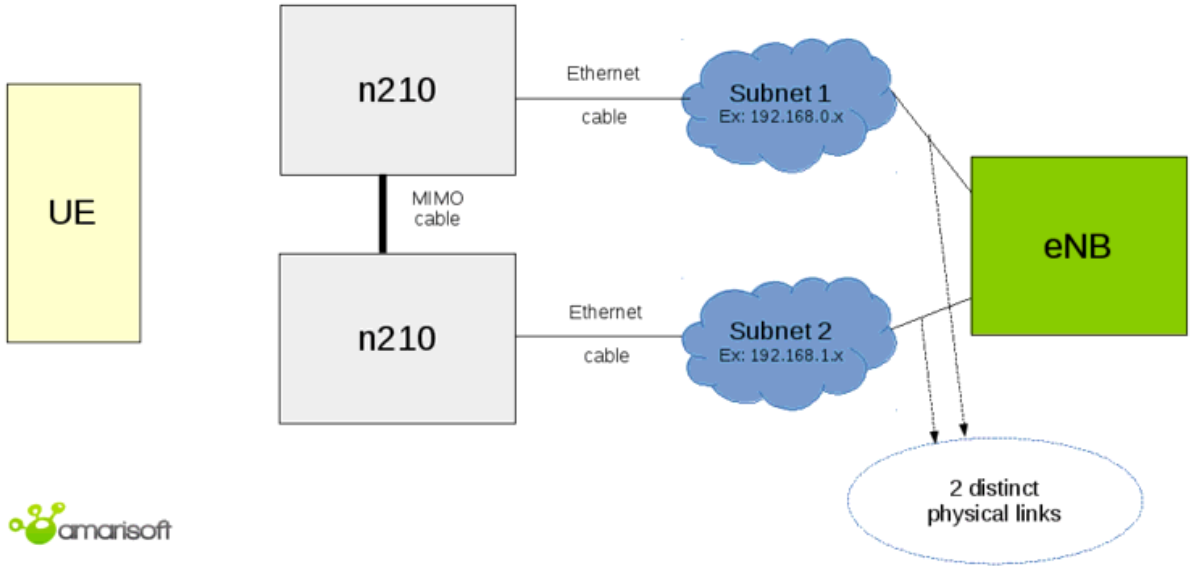
7.9.1.2 N2x0 setup

To setup your MIMO environment with N2x0 device, here is a detailed diagram of how to proceed.



This configuration may have a bottle neck at ethernet side because USRP speed is limited to 1Gbps.
That's why for 20Mhz MIMO configuration, downlink sampling rate is limited to 8 (`dl_sample_`

`bits` parameter).
You can remove this constraint with the following diagram:

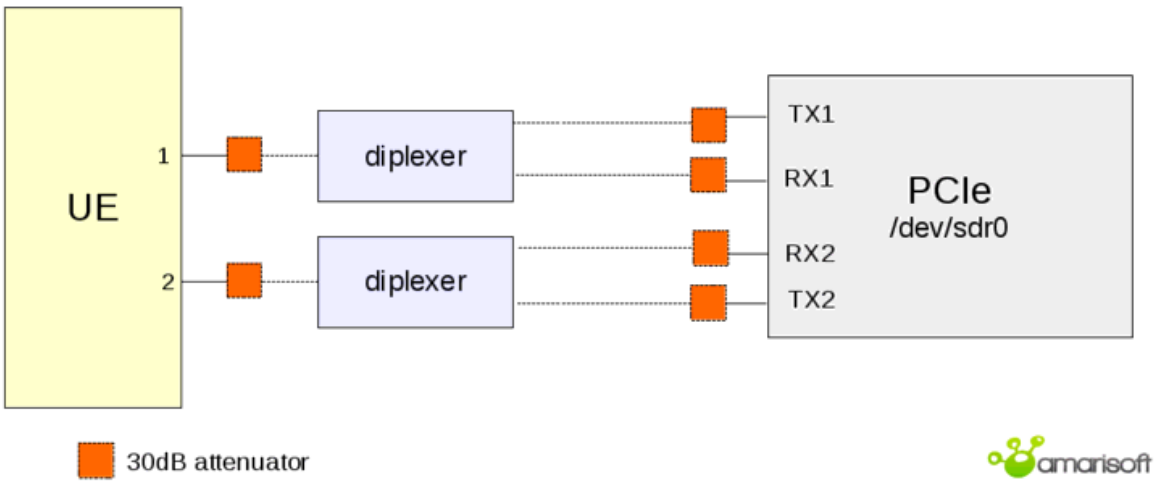


7.9.2 Using cable

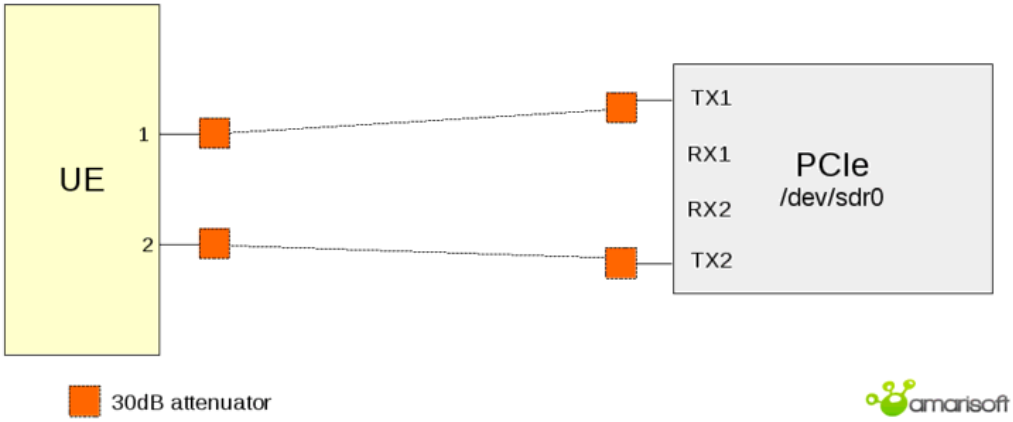
Note that the diagrams provided below are only examples.
You may adapt depending on the UE.

7.9.2.1 PCIe SDR setup

If you are using FDD mode, the general case will be as follows. Note that if there is no MIMO in UL, you can simplify by removing the RX2 connection, thus connecting the TX2 directly to antenna 2 at UE side.

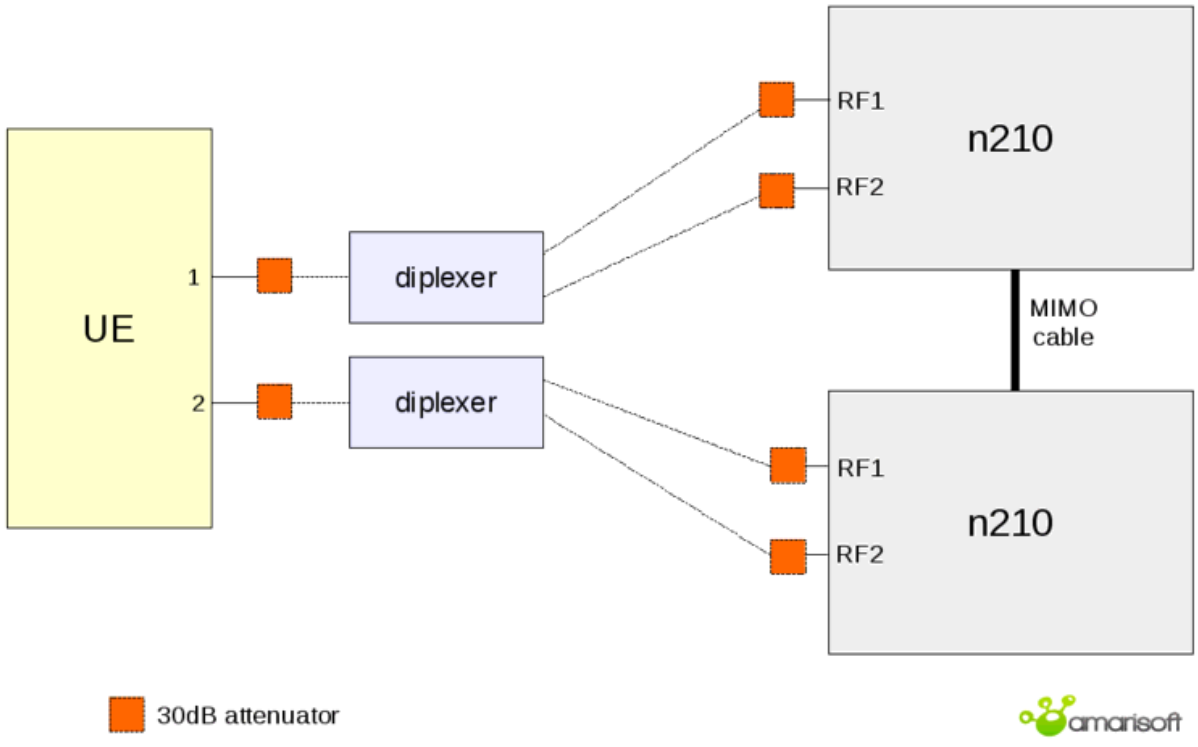


For TDD mode, you only need to connect TX1 and TX2.

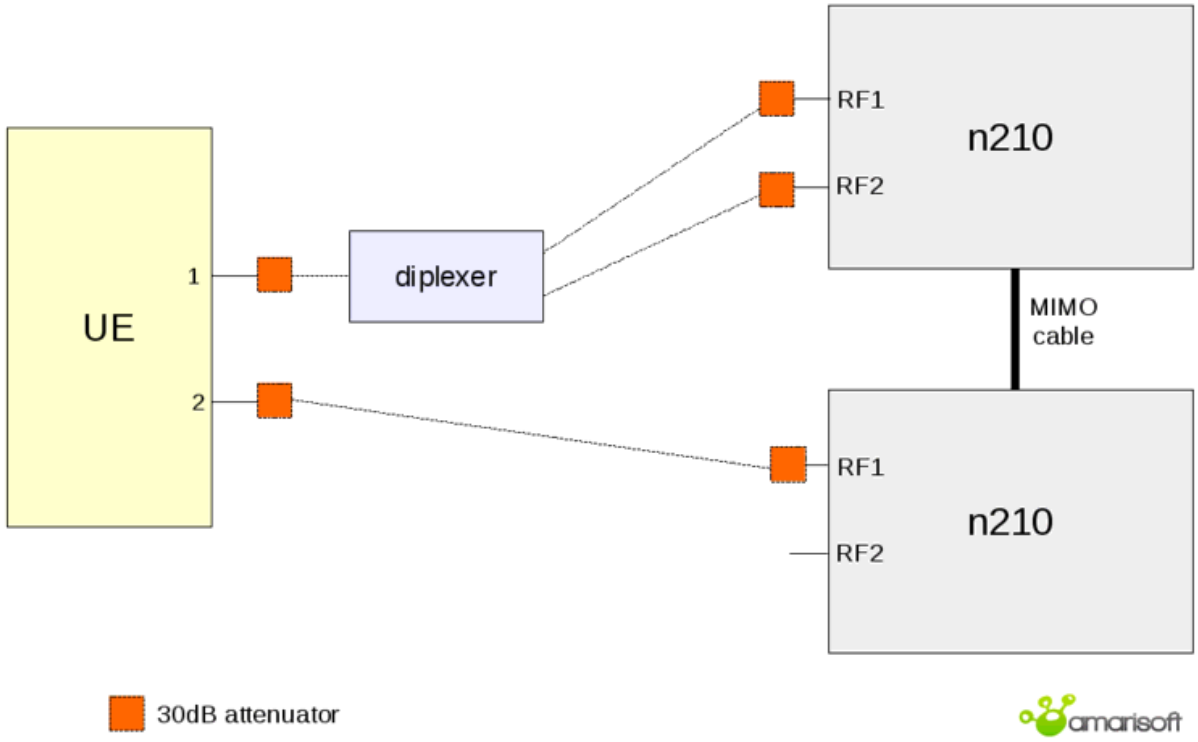


7.9.2.2 N2x0 setup

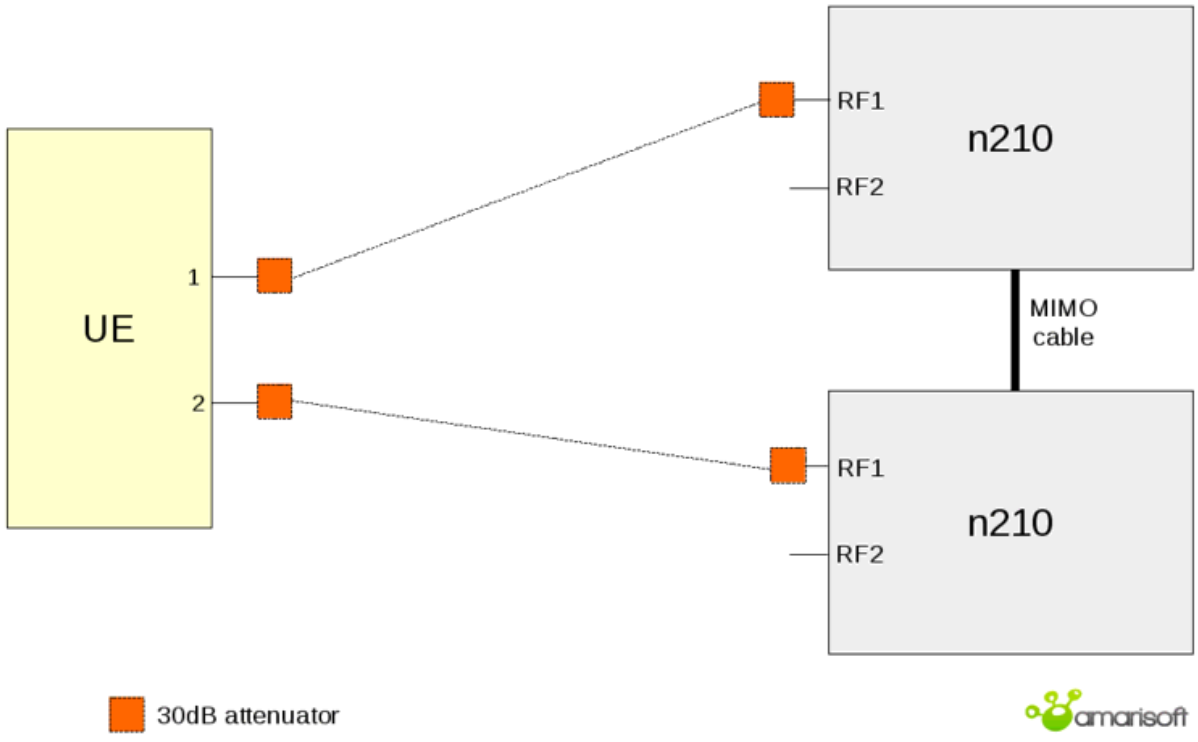
If you are using FDD mode, the general case will be:



On most UE, second antenna is only used for RX so you can simplify with:

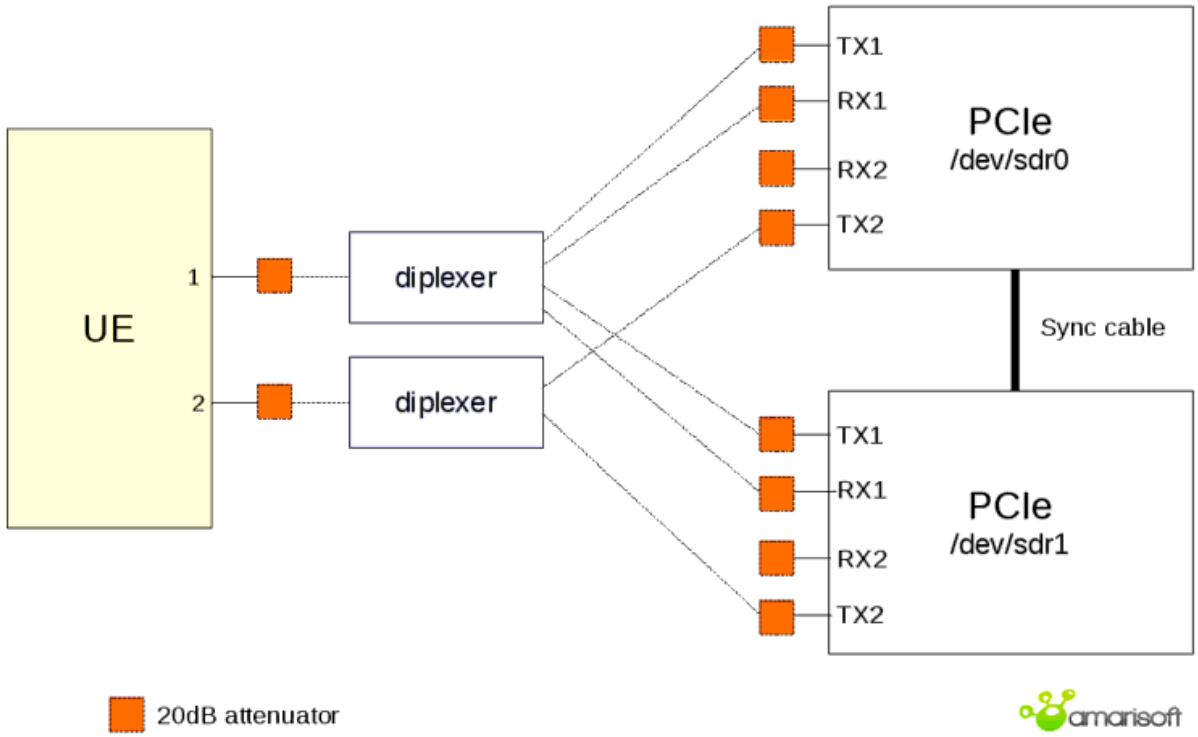


If your are using TDD mode, only one antenna is necessary per USRP.
Thus, you only need following diagram:



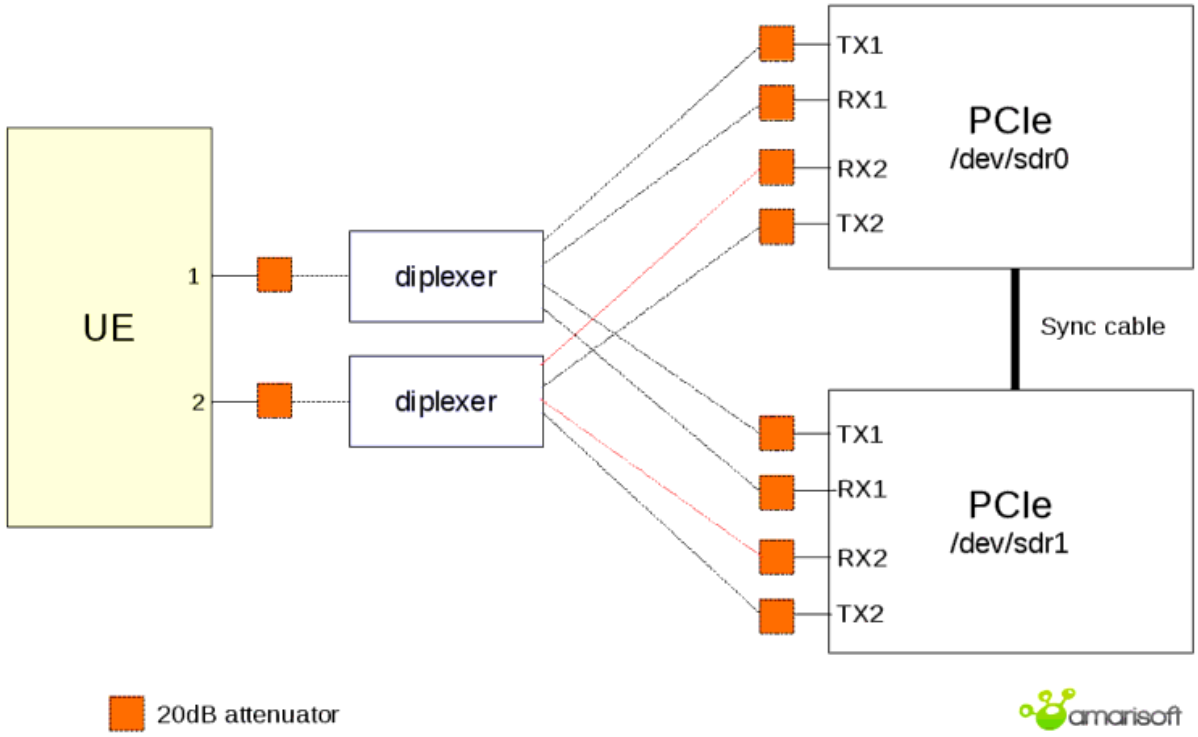
7.10 Carrier Aggregation support

CA is currently supported with PCIe SDR radio frontend. Intra-band CA could be handled with one single PCIe SDR card with the same constrains specified for intra-band multi-cell support, See [Intra-band multi-cell], page 15. For inter-band CA, one PCIe SDR card is required per band. The following diagram depicts the environment setup for DL inter-band CA with 2 carriers and MIMO in DL.



The configuration `enb-2cc-mimo.cfg` is an example of DL CA with 2 carriers in band 2 and band 4 (20+20 MHz) and 2x2 DL MIMO configuration using transmission mode 3.

If you would like to enable CA in UL as well, you should connect the RX2 connectors as below.



7.11 TDD support

The configuration file `enb-tdd.cfg` is an example of TDD configuration. The eNodeB supports all 7 UL/DL TDD configurations.

7.12 Category M1

The eNodeB supports Category M1 UEs (Bandwidth Reduced UEs). They only receive or transmit on a 1.4 MHz bandwidth so they cannot use all the standard LTE signals. In particular, BR specific system information blocks are necessary. The configuration file `enb-catm1.cfg` is an example of Category M1 configuration. See [Bandwidth Reduced parameters], page 53.

7.13 NB-IoT

The eNodeB supports NB-IoT cells. They use a 200 kHz bandwidth which can be inside an existing LTE cell (in-band operation mode), at its edge (guard band mode) or completely independent (standalone mode). An example of standalone NB-IoT configuration is in file `enb-nbiot.cfg`. An example of in-band NB-IoT configuration is in file `enb-nbiot-inband.cfg`.

8 Configuration reference

8.1 Configuration file syntax

The main configuration file uses a syntax very similar to the Javascript Object Notation (JSON) with few extensions.

1. Supported types:
   - Numbers (64 bit floating point). Notation: `13.4`
   - Complex numbers. Notation: `1.2+3*I`
   - Strings. Notation: `"string"`
   - Booleans. Notation: `true` or `false`.
   - Objects. Notation: `{ field1: value1, field2: value2, .... }`
   - Arrays. Notation: `[ value1, value2, .... ]`
2. The basic operations `+`, `-`, `*` and `/` are supported with numbers and complex numbers. `+` also concatenates strings. The operators `!`, `||`, `&&`, `==`, `!=`, `<`, `<=`, `>=`, `>` are supported too.
3. The numbers `0` and `1` are accepted as synonyms for the boolean values `false` and `true`.
4. {} at top level are optional.
5. " for property names are optional.
6. Properties can be duplicated.
   Merge will be done by recursively overriding values considering reading direction.

```
{
    value: "foo",
    value: "bar",
    sub: {
        value: "foo"
    },
    sub: {
        value: "bar"
    }
}
```

   Will be equivalent to:

```
{
    value: "bar",
    sub: {
        value: "bar"
    }
}
```

7. Files can be included using *include* keyword (must not be quoted) followed by a string (without :) representing the file to include (path is relative to current file) and terminating by a comma.
   Arrays can't be included.
   Merge will be done as for duplicate properties.
   If *file1.cfg* is:

```
    value: "foo",
    include "file2.cfg",
    foo: "foo"
```

   And *file2.cfg* is:

```
    value: "bar",
```

```
    foo: "bar"
```

Final config will be:

```
{
    value: "bar",
    foo: "foo"
}
```

8. A C like preprocessor is supported. The following preprocessor commands are available:

   `#define` *var expr*
   : Define a new variable with value *expr*. *expr* must be a valid JSON expression. Note that unlike the standard C preprocessor, *expr* is evaluated by the preprocessor.

   `#undef` *var*
   : Undefine the variable *var*.

   `#include` *expr*
   : Include the file whose filename is the evaluation of the string expression *expr*.

   `#if` *expr* Consider the following text if *expr* is true.

   `#else` Alternative of `#if` block.

   `#elif` Composition of `#else` and `#if`.

   `#endif` End of `#if` block.

   `#ifdef` *var*
   : Shortcut for `#if defined(var)`

   `#ifndef` *var*
   : Shortcut for `#if !defined(var)`

   In the JSON source, every occurrence of a defined preprocessor variable is replaced by its value.

9. Backquote strings: JSON expression can be inserted in backquote delimited strings with the `${expr}` syntax. Example: `` `abc${1+2}d` `` is evaluated as the string `"abc3d"`. Preprocessor variables can be used inside the expression.

The System Information Blocks use the ASN.1 GSER syntax defined in RFC 3641 (Generic String Encoding Rules for ASN.1 Types). The description of the exact content of the System Information Blocks can be found in 3GPP TS 36.331 (RRC).

8.2 Global properties

`log_filename`
: String. Set the log filename. If no leading `/`, it is relative to the configuration file path. See [Log file format], page 87.

`log_options`
: String. Set the logging options as a comma separated list of assignments.

  - *layer*.level=*verbosity*. For each layer, the log verbosity can be set to `none`, `error`, `info` or `debug`. In debug level, the content of the transmitted data is logged.
  - *layer*.max_size=*n*. When dumping data content, at most `n` bytes are shown in hexa. For ASN.1, NAS or Diameter content, show the full content of the message if `n > 0`.

- *layer*.payload=[0|1]. Dump ASN.1, NAS, SGsAP or Diameter payload in hexadecimal.
- *layer*.key=[0|1]. Dump security keys (NAS and RRC layers).
- *layer*.crypto=[0|1]. Dump plain and ciphered data (NAS, RRC and PCDP layers).
- phy.signal=[0|1]. Dump binary received signal data of the physical layer to another file (*log_filename*.bin). The currently available data are QAM constellations (PUSCH, PDSCH) and channel response information (PUSCH, SRS). The GUI can be used to display them. Note: the size of the binary signal data is larger than the textual logs, so they should be enabled only when needed.
- phy.rep=[0|1]. Log the NPUSCH/NPDCCH/NPDSCH repetitions in each subframe (NB-IoT eNodeB only).
- time=[sec|short|full]. Display the time as seconds, time only or full date and time (default = time only).
- file=cut. Close current file log and open a new one.
- file.rotate=now. Rename current log with timestamp and open new one.
- file.rotate=*size*. Rename current log every time it reaches *size* bytes open new one. Size is an integer and can be followed by K, M or G.
- file.path=*path*. When log rotation is enabled, move current log to this path instead of initial log path.
- bcch=[0|1]. Enable or disable BCCH log. The BCCH is always transmitted, so it gives large logs when enabled.
- append=[0|1]. (default=0). If 0, truncate the log file when opening it. Otherwise, append to it.

Available layers are: `phy`, `mac`, `rlc`, `pdcp`, `rrc`, `nas`, `s1ap`, `x2ap`, `m2ap`, `gtpu`

`pcap` Optional object. Gives the Wireshark capture options. The `mac-lte-framed` protocol using link-layer type 147 is supported (`http://wiki.wireshark.org/MAC-LTE`). In order to enable it in Wireshark, go to the menu `Edit->Preferences->protocols->DLT_USER->Edit->New` and add the DLT type 147 (User 0) with payload protocol `mac-lte-framed`.

`filename` String. Filename in which the capture is stored.

`bcch` Optional boolean (default = false). If true, the BCCH SI PDUs are logged. It is disabled by default because the capture size increases even when the eNodeB is idle.

`max_data_len` Optional integer (default = 65536). Maximum captured MAC PDU length per packet.

`use_pipe` Optional boolean (default = false). Capture in a pipe instead of a regular pipe. This allow live capture with wireshark: `wireshark -k -i <filename>`. Note that it can be used only once as wireshark requires initial header.

`enb_name`

Optional string. Set eNB name used in S1 connection setup request.

`gtp_addr`

String. Set the IP address (and optional port) on which the GTP-U packets are received. The default port is 2152. It is normally the IP address of the network interface connected to the core network.

`gtp_payload_mtu`
: Optional integer (range 68 to 16384, default = 1500). MTU in bytes for the GTP-U payload. Do not forget to update the network interface MTU accordingly for optimal performance. For example with a GTP MTU of 1500 bytes, interface should have a MTU of at least 1564 bytes.

`mme_list` Array of objects. List of MME to which the eNodeB is connected. Each object contains the following properties:

`mme_addr`
: String. Set the IP address (and optional port) of S1AP SCTP connection to the MME. The default port is 36412.

  Syntax:

  - `"1.2.3.4"` (use default port)
  - `"1.2.3.4:5678"` (use explicit port)
  - `"2001:db8:0:85a3::ac1f:8001"` (IPv6 address and default port)
  - `"[2001:db8:0:85a3::ac1f:8001]:5678"` (IPv6 address and explicit port)

`gtp_ext_addr`
: Optional string. Set the IP address on which the Core Network should transmit the GTP-U packets. It is the same as gtp_addr by default. It can be different if eNodeB is behind a NAT.

`s1ap_bind_addr`
: Optional string. IP address and optional port on which the S1AP SCTP connection is bound.

`qci_dscp_mapping`
: Optional array of objects. Allows to define a specific IP differentiated services code point for a given QCI. QCI not explicitly configured use the default DSCP value 0.
  Each object must contain the following properties:

  `qci` Integer (range 1 to 254). QCI value.

  `dscp` Integer (range 0 to 63). DSCP value.

For backward compatibility, if `mme_list` is omitted, then a single MME is assumed and the properties `mme_addr`, `gtp_ext_addr` and `s1ap_bind_addr` are expected at the top level.

`x2ap_bind_addr`
: Optional string. IP address and optional port on which the X2AP SCTP connection is bound.

`x2_peers` Optional array of strings. IP addresses and optional port of other eNodeBs to establish X2 connections. The default port is 36422.

`mbmsgw_addr`
: Optional string. Set the IP address (and optional port) of the MBMS Gateway for the M2 connection. The default port is 36443.

`mbms_gtp_u_port`
: Optional integer. GTP-U local port number used to receive the MBMS packets. The default port is 2152.

`mcc` String. The MCC part of the PLMN (3 digits). This property is obsolete. Use `plmn_list` instead.

`mcc` String. The MNC part of the PLMN (2 or 3 digits). This property is obsolete. Use `plmn_list` instead.

`enb_type` Optional enumeration: macro, short_macro, long_macro or home (default = macro). Select between macro or home eNodeB.

`enb_id` Integer. The 18 bit (short macro), 20 bit (macro), 21 bits (long macro) or 28 bit (home) eNodeB global identifier

`rf_driver`
Object. Parameters of the radio driver. See [Radio driver configuration], page 31.

`tx_gain` Float or array of floats. Transmit gain in dB. The range is device dependent. For the PCIe SDR board, the range is between 0 and 89.75 dB. For the USRP N2x0 device with the SBX daughterboard, the range is 0 to 31.5 dB. With an array of floats a different gain is specified for each channel.

`rx_gain` Float or array of floats. Receive gain in dB. The range is device dependent. For the PCIe SDR board, the range is between -11 and 77 dB (the exact limits depend on the RX frequency). For the USRP N2x0 device with the SBX daughterboard, the range is 0 to 31.5 dB. With an array of floats a different gain is specified for each channel.

`com_addr` Optional string. Address of the WebSocket server remote API. See [Remote API], page 69.
If set, the WebSocket server for remote API will be enabled and bound to this address.
Default port is 9000.
Setting IP address to 0.0.0.0 will make remote API reachable through all network interfaces.

`com_name` Optional string. Sets server name. ENB by default

`com_ssl_certificate`
Optional string. If set, forces SSL for WebSockets. Defines CA certificate filename.

`com_ssl_key`
Optional string. Mandatory if *com_ssl_certificate* is set. Defines CA private key filename.

`com_ssl_peer_verify`
Optional boolean (default is false). If *true*, server will check client certificate.

`license_server`
Configuration of the Amarisoft license server to use.
Object with following properties:

`server_addr`
String. IP address of the license server.

`name` Optional string. Text to be displayed inside server monitor or remote API.

`tag` Optional string. If set, server will only allow license with same tag.

Example:

```
license_server: {
    server_addr: "192.168.0.20"
}
```

`cell_list`
: Array of object. Each element gives the configuration of a cell. The property `cell_default` gives a default value for each property. See [Cell configuration], page 32.

`cell_default`
: Optional Object. Gives a default value for the cell configuration.

`nb_cell_list`
: Optional array of object. Each element gives the configuration of a NB-IoT cell. See [NB-IoT cell configuration], page 59.

8.3 Advanced properties

`internal_time_ref`
: Optional enumeration: os_clock or rf_frontend (default = os_clock). Selects the time source for the SIB16. `fr_frontend` selects the time from the RF frontend. It is the normal choice when the RF frontend is time synchronized. `os_clock` uses the OS clock and derives the number leap seconds by using the `right/UTC` Unix time zone.

`rf_frontend_time_offset`
: Optional integer (default = 0). Gives the difference in ms between the time given by the rf frontend and the International Atomic Time (TAI).

`frame_epoch`
: Optional integer (default = 0). Gives the TAI time in ms at which the LTE frame 0 starts. Use 0 to have the frame 0 start at 1970-01-01 00:00:00 TAI. Use 315964819000 to have the frame 0 start at 1980-01-06 00:00:19 TAI (GPS time = 0).

`dl_freq` Optional float. Tuning frequency in MHz for the downlink. It is automatically set to the average of the DL center frequency of each cell.

In the multi-cell case, if the radio head has a degraded output near the center of the transmitted spectrum (which is the case for zero IF TX or RX architectures), it is interesting to move the center of the transmitted spectrum outside the spectrum of every cell or in the middle of the spectrum of a given cell.

In this case, the `dl_freq` property can be used. It must be set so that for each cell `dl_freq - dl_cell_freq` is a multiple of 15 kHz (`dl_freq_cell` is assumed to be the center frequency of a cell).

Note: if you want to use non standard LTE frequencies, use the `custom_freq_band` option.

`ul_freq` Optional float. Tuning frequency in MHz for the uplink. It is automatically set to the average of the UL center frequency of each cell.

Same remark as `dl_freq`.

`channel_dl`
: Optional object. Set the channel simulator configuration. See [Channel simulator], page 67.

`n_antenna_dl`
: Optional integer. Set the number of DL antennas. Only useful if the channel simulator is used to set a different number of physical DL antennas at the output of the channel simulator. See [Channel simulator], page 67.

`sample_rate`
: Optional float. Sample rate in MHz. It is normally automatically set depending on the radio head capabilities and selected cell bandwidth.

`sample_rate_num`
: Optional integer. Main sample rate used for the LTE signal processing in 1.92 MHz units (hence 3 means 5.76 MHz). It is normally automatically set depending on the radio head capabilities and selected cell bandwidth. If the resulting rate is different from `sample_rate`, a fractional sample rate interpolator is used to convert the sample rate.

`tx_pad_duration`
: Optional integer (default = 35). Duration (in 1/1.92 us units) of the zero sample burst sent before the start of the downlink burst in TDD. It corresponds to the power amplifier ramp up duration. The appropriate value depends on the radio head.

`tx_time_offset`
: Optional integer (default = 0). Time offset (in samples) for the downlink. It may be needed to compensate internal delays in the radio head. In a normal setup, this value should be set to zero.

`rx_ta_offset`
: Optional float (default = 2.0). Time offset (in 1/1.92 us) for the uplink. With a well calibrated radio head (i.e. where the TRX timestamps take into account the internal radio head delays), it gives the PRACH timing advance of a UE close to the eNodeB. A small non zero value (such as 2) is recommended.

`tdd_legacy_timing`
: Optional boolean (default = true). If true, the downlink frame starts 39/1.92 us after the GPS origin. Otherwise, the downlink frame starts at the GPS origin. For interoperability purposes with other eNodeBs, it is better to set it to false. The default value is true for compability with previous versions.

`tx_gain_offset`
: Optional float. Set the TX digital gain. Warning: do not change it unless you know what you do because a too high value introduces saturation in the output. The default value is -12 dB if no DL channel simulator is used and -21 dB otherwise.

`custom_freq_band`
: Optional object or array of objects. Define a non standard LTE frequency band. Standard bands can also be overriden by this option. If the uplink information is not provided, it is assumed to be the same as the downlink (TDD band). Use an array of objects if you want to define more than one custom band.

  `band` Range: 1 to 256.

  `dl_earfcn_min`
  : Range: 0 to 262143.

  `dl_earfcn_max`
  : Range: 0 to 262143.

  `dl_freq_min`
  : Float. Low DL frequency in MHz.

  `ul_earfcn_min`
  : Optional integer. Range: 0 to 262143.

  `ul_earfcn_max`
  : Optional integer. Range: 0 to 262143.

  `ul_freq_min`
  : Optional Float. Low UL frequency in MHz.

`papr_reduction`
: Optional object. Define the parameters for Peak to Average Power Ratio (PAPR) reduction. It is only useful if you use a high power amplifier. In the current version it takes a significant amount of CPU time, so it is only usable for LTE bandwidth <= 10 MHz.

  The following properties are available:

  `enabled` Boolean. If true, PAPR reduction is enabled.

  `a_max` Float. Set the cut-off level in dB relative to the Reference Signal power.

  `evm_max` Float. Set the maximum Error Vector Magnitude (EVM) for 64QAM.

  `oob_points`
  : Array of floats. Each pair of number defines a point of the maximum allowed distorsion curve. The first number is the frequency offset in MHz from the edge of the LTE spectrum. The second number is the power level in dB. The actual curve is linearly interpolated between the points.

  An example of use is given in the configuration `enb-1_4mhz.cfg`.

`rate_bucket_duration`
: Optional. Range 50 to 1000 (default = 100). Duration in ms for the average bit rate estimation. It is used to enforce the UE Aggregate Maximum Bit Rate and GBR ERAB Maximum Bit Rate.

`sched_rate_duration`
: Optional. Range 5 to 1000 (default = 50). Period in ms for the average bit rate estimation for the MAC scheduler.

`sched_metric`
: Optional enumeration: pf, rr, mt (default = pf). Set the MAC scheduler metric. Available possibilities:

  `pf` Proportionally fair

  `rr` Round-robin

  `mt` Maximum throughput

`sched_latency_for_prb_max`
: Optional. Range: 5 to 1000 (default = 50). Approximate maximum latency in ms. It is used to limit the maximum number of UEs per TTI.

`automatic_ue_info_request`
: Optional boolean (default = false). If set, the eNB will send a UE Information message if the UE indicates the availability of information in RRC Connection Setup Complete, RRC Connection Reestablishment Complete or RRC Connection Reconfiguration Complete message.

8.4 Radio driver configuration

The `name` property selects the driver. The corresponding DLL file name is `trx_`*`name`*`.so`. It is searched in the lteenb executable directory, in the path configured in the `path` property. The following drivers are currently available:

`dummy` Dummy driver. Can be used to measure the RX to TX latency.

`sdr` Amarisoft PCIe SDR driver.

`uhd` Ettus Research UHD driver for USRP N2x0, B2x0 and X3x0 series.

`lms7002m` Lime MicroSystem LimeSDR platform driver.

`perseus` Nutaq driver for PicoSDR 2x2.

If you don't have and need one of these drivers, please contact `customer@amarisoft.com` and ask for it.

8.4.1 Dummy driver

No specific properties are available.

8.4.2 UHD driver

Please check Amarisoft UHD documentation delivered within package.

8.4.3 SDR driver

Please check Amarisoft SDR documentation delivered within package.

8.5 Cell configuration

8.5.1 Basic parameters

These parameters are the most important ones and must usually be modified when a new cell is added.

`plmn_list`

Array of objects or strings. List of PLMNs broadcasted by the eNodeB. At most 6 PLMNs are supported. Each element of the array is either a PLMN (5 or 6 digit string) or an object containing the following properties:

`plmn` String. PLMN (5 or 6 digits).

`reserved` Boolean. True if the cell is reserved for operator use.

`cp_ciot_opt`

Optional boolean (default = false). Indicates if PLMN supports CP-CIoT EPS optimisation.

`attach_without_pdn`

Optional boolean (default = false). Indicates if PLMN supports attach without PDN connectivity.

When `reserved` is not provided, its default value is `false`.

`dl_earfcn`

Range: 0 to 262143. Set the DL EARFCN. See `http://niviuk.free.fr/lte_band.php` to convert between the center frequency and EARFCN. When several cells share the same radio front end, the difference of their center DL frequency must be a mutiple of 300 kHz (i.e. the difference of their DL EARFCN must be a multiple of 3). Also, the difference between the DL center frequency of each cell and the average of DL center frequencies must be a multiple of 15 kHz.

`ul_earfcn`

Optional. Range: 0 to 262143. Set the UL EARFCN. If not provided, the default DL/UL gap is used (i.e. ul_earfcn = dl_earfcn + 18000). `ul-CarrierFreq` in SIB2 is automatically set to the corresponding value. When several cells share the same radio front end, the difference of their center UL frequency must be a mutiple of 300 kHz (i.e. the difference of their UL EARFCN must be a multiple of 3). Also,

the difference between the UL center frequency of each cell and the average of UL center frequencies must be a multiple of 15 kHz.

`multi_band_list`
Optional array of integers. List the additional bands supported by the cell, in decreasing priority order (MFBI feature). The downlink and uplink frequency of the cell must exist in all these bands.

`cell_id` Range: 0 to 1023. 7 bit (long macro), 8 bit (macro) or 10 bit (short macro) cell identifier. For a macro eNodeB, the 28 bit cell identifier is the concatenation of `enb_id` and `cell_id`.

`tac` Range: 0 to 65535. Tracking Area Code of the cell.

`n_id_cell`
Range: 0 to 503. Physical cell identifier. Each neighbour cell operating on the same frequency must have a different physical cell identifier modulo 3.

`root_sequence_index`
Range: 0 to 837. Set the PRACH root sequence index. It must be different for each neighbour cell operating on the same frequency and sharing the same PRACH configuration.

`ncell_list`
Optional array of objects. List of neighbour cells. Used for to convert the physical cell identity and EARFCN to a cell identity in case of handover. Each neighbour cell is defined by the following properties:

`n_id_cell`
Range: 0 to 503. Physical cell identity.

`dl_earfcn`
Optional. Range 0 to 262143. DL EARFCN. If not present, it is assumed to be the same as the current cell.

`plmn` Optional string. PLMN of the cell (5 or 6 digits). The default is the same PLMN as the eNB.

`cell_id` Integer. 28 bit cell identifier.

`tac` Range: 0 to 65535. Tracking Area Code.

`type` Optional string. Can be "macro" (default) for macro eNB, "short_macro" for short macro eNB, "long_macro" for long macro eNB or "home" for home eNB. Only used for S1 handover.

`allowed_meas_bandwidth`
Optional integer 6, 15, 25, 50, 75 or 100. Defines the allowed measurement bandwidth to be used for this cell. If the field is not present, it uses the serving cell downlink bandwidth.

`antenna_port_1`
Optional boolean. Indicates if antenna port 1 is used by the cell. If the field is not present, it uses the serving cell configuration.

`neigh_cell_config`
Optional integer, range 0 to 3, default to 1 (means 'no MBSFN subframes are present in all neighbour cells'). Sets the neighbour cell information as specified in TS 36.331. It must be the same for all cells belonging to the same frequency. The allowed values are:

`0` Not all neighbour cells have the same MBSFN subframe allocation as the serving cell on this frequency, if configured, and as the PCell otherwise

`1` No MBSFN subframes are present in all neighbour cells

`2` The MBSFN subframe allocations of all neighbour cells are identical to or subsets of that in the serving cell on this frequency, if configured, and of that in the PCell otherwise

`3` Different UL/DL allocation in neighbouring cells for TDD compared to the serving cell on this frequency, if configured, and compared to the PCell otherwise

`individual_offset`
Optional enumeration: -24, -22, -20, -18, -16, -14, -12, -10, -8, -6, -5, -4, -3, -2, -1, 0, 1, 2, 3, 4, 5, 6, 8, 10, 12, 14, 16, 18, 20, 22 or 24. Individual offset in dB given to the UE in the Measurement Object for the corresponding cell.

`multi_band_list`
Optional array of integers. List the additional bands supported by the cell, in decreasing priority order (MFBI feature). The downlink and uplink frequency of the cell must exist in all these bands. If not present, it will take the `multi_band_list` configuration from the serving cell.

`n_rb_dl` Integer. Range: 6 to 100. Set the number of DL resource blocks. The corresponding LTE bandwidth can be deduced from the following table:

`6` 1.4 MHz

`15` 3 MHz

`25` 5 MHz

`50` 10 MHz

`75` 15 MHz

`100` 20 MHz

Note: It is always necessary to modify the SIB2 when changing the LTE bandwidth.

`pucch1_sr_detect_threshold`
Optional float. SNR threshold in dB to filter Scheduling Request detection in PUCCH format 1/1A/1B.

`pucch1_an_detect_threshold`
Optional float. SNR threshold in dB to filter HARQ ACK/NACK detection in PUCCH format 1/1A/1B.

8.5.2 Advanced parameters

`n_antenna_dl`
Enumeration: 1, 2, 4 or 8. Number of DL antennas. It must be the same for all NB-IoT and LTE cells sharing the same RF port. Currently 1 (SISO) 2 (MIMO 2x2) or 4 (MIMO 4x4) are supported.

`n_antenna_ul`
Enumeration: 1, 2, 4 or 8. Number of UL antennas. It must be the same for all NB-IoT and LTE cells sharing the same RF port.

`n_antenna_pbch`
: Optional enumeration: 1, 2 or 4 (default = n_antenna_dl). Number of PBCH antennas. Must be <= n_antenna_dl.

`rf_port`
: Optional integer (default = 0). This parameter selects the RF port when several cells on different RF interfaces or RF bands are handled by the eNodeB. The number of supported RF ports depends on the radio head. For example, each N210 counts as one RF port.

`cell_gain`
: Optional float (default = 0). Downlink cell gain in dB. Must be between -200 and 0 (included).

`rx_epre_in_dbfs`
: Optional boolean (default = false). In the logs, the EPRE (Energy Per Resource Element) is displayed in dBm if the RF interface provides its reference receive power and if rx_epre_in_dbfs = false. Otherwise it is displayed in dBFS (Decibels relative to Full Scale).

`rx_epre_offset`
: Optional float (default = 0). Offset in dB applied to all the receive EPRE measurements.

`manual_ref_signal_power`
: Optional boolean (default = false). If the RF interface provides its transmit power, then SIB2.referenceSignalPower is automatically set. If manual_ref_signal_power is true, then SIB2.referenceSignalPower is never automatically set by the eNodeB.

`cyclic_prefix`
: Enumeration: normal or extended. Set the DL cyclic prefix.

`uldl_config`
: (TDD only) Range: 0 to 6. TDD frame configuration.

`sp_config`
: (TDD only) Range: 0 to 9. TDD special subframe configuration. The special subframe 7 (with extended cyclic prefix) and 9 (with normal cyclic prefix) from the release 11 are supported.

`long_range`
: Optional boolean (default = false). If true, enable a proprietary Amarisoft extension to extend the cell range (modified UEs are necessary). Only FDD mode is supported. PRACH format 1 or 3 must be used and SIB2.zeroCorrelationZoneConfig must be set to zero. The HARQ round-trip time is increased from 8 to 10 ms.

`power_p_sync`
: Optional float (default = 0). Set the relative power in dB of the Primary Synchronization Signal.

`power_s_sync`
: Optional float (default = 0). Set the relative power in dB of Secondary Synchronization Signal.

`power_pcfich`
: Optional float (default = 0). Set the relative power in dB of PCFICH.

`power_pbch`
: Optional float (default = 0). Set the relative power in dB of PBCH.

`power_phich`
: Optional float (default = 0). Set the relative power in dB of PHICH.

`power_pdcch`
: Optional float (default = 0). Set the relative power in dB of PDCCH.

`power_pdsch_si`
: Optional float (default = 0). Set the relative power in dB of PDSCH when transmitting SI/RA/P-RNTI information.

`phich_duration`
: Enumeration: normal or extended. Set the PHICH duration.

`phich_resource`
: Enumeration: 1/6, 1/2, 1 or 2. Set the PHICH resource.

`sib1`
: String. Filename of the textual ASN.1 content (GSER syntax) for SIB1. The fields `plmn-Identity`, `trackingAreaCode`, `cellIdentity` and `freqBandIndicator` are automatically modified by the eNodeB.

`sib_sched_list`
: Array of strings. Filenames of the other SIBs in the same order as the scheduling list in SIB1. The first entry must contain the SIB2. The content is in textual ASN.1 (GSER syntax).

  The eNodeB uses parameters from SIB2 for its configuration. The SIB2 field `rootSequenceIndex` is automatically modified by the eNodeB.

`si_coderate`
: Float. Maximum code rate for System Information Blocks (SIBs).

`rar_coderate`
: Optional float. Maximum code rate for Random Access Response (RAR) (default = same as `si_coderate`).

`paging_coderate`
: Optional float. Maximum code rate for paging messages (default = same as `si_coderate`).

`paging_cat0_coderate`
: Optional float. Maximum code rate for paging messages for category 0 UEs (default = same as `paging_coderate`).

`si_pdcch_format`
: Range: 2 to 3. Log2 of the number of CCEs for PDCCH for SIBs.

`rar_pdcch_format`
: Optional. Range: 2 to 3. Log2 of the number of CCEs for PDCCH for RAR (default = same as `si_pdcch_format`).

`paging_pdcch_format`
: Optional. Range: 2 to 3. Log2 of the number of CCEs for PDCCH for paging (default = same as `si_pdcch_format`).

`paging_cat0_pdcch_format`
: Optional. Range: 2 to 3. Log2 of the number of CCEs for PDCCH for paging for category 0 UEs (default = same as `paging_pdcch_format`).

`rar_backoff`
: Optional. Range: -1 to 15. If set to -1, no Backoff Indicator is sent in the Random Access Response message. Values 0 to 15 refer to the index of table 7.2-1 found in 3GPP 36.321.

`sib_sfn_offset`
: Optional array of integer. If present, set the offsets of the SIBs in the SI window. At most 4 offsets are allowed. If it not present, the eNodeB uses default values.

`n_symb_cch`
: Range: 0 to 4. Number of OFDM symbols for PDCCH. 0 means to automatically adjust the number of OFDM symbols. 0 should not be used in a cell where cross carrier PDCCH signalling is enabled.

`half_duplex_ue`
: Optional boolean (default = false). If true, the support of HD-FDD UEs is enabled. Because it introduces some restrictions in the choice of scheduling parameters and because HD-FDD UEs are not commercially deployed, the feature is disabled by default.

`allow_cat0_ue`
: Optional boolean (default = false). If true, category 0 UEs (release 12) can connect to the eNodeB. The corresponding SIB1 field is set and the scheduler takes the category 0 UE scheduling restrictions into account.

`edrx`
: Optional boolean (default = false). If true, extended idle mode DRX support is activated in the cell, and Hyper Sub Frame Number value is scheduled in SIB1.

`pdsch_dedicated`
: Object. PDSCH dedicated configuration. Currently shared by all UEs. The following properties are defined:

  `p_a`
  : Enumeration: -6, -4.77, -3, -1.77, 0, 1, 2, 3. Set the `p_a` parameter which sets the PDSCH average power.

  `dmrs`
  : Optional array of 2 integers. Range: 0 to 503. Specifies the DMRS scrambling identity when transmission mode 10 is used (release 11).

  `qcl_operation`
  : Optional enumeration: `typeA` or `typeB` (default = `typeB`). Select the PDSCH Quasi Co-Location parameter when transmission mode 10 is used (release 11). Note that this parameter does not modify the eNodeB behavior, it just changes the value of the corresponding RRC field.

`pdcch_format`
: Optional. Range: 0 to 3. If defined, force for number of CCEs for UE specific PDCCH to `2^pdcch_format`. Otherwise it is computed from the reported CQI.

`initial_cqi`
: Range: 1 to 15. This CQI value is assumed when none is received from the UE.

`pucch_dedicated`
: Optional object. PUCCH dedicated configuration. Currently shared by all UEs.

  `tdd_ack_nack_feedback_mode`
  : (TDD only) Enumeration: `bundling` or `multiplexing`. Define the ACK/NACK feedback mode for TDD.

  `tdd_ack_nack_feedback_mode_r10`
  : (TDD only) Optional enumeration: `bundling`, `multiplexing`, `cs`, `pucch3`. Select the ACK/NACK feedback mode for release 10 TDD UE. `cs` means channel selection. By default it is the same as `tdd_ack_nack_feedback_mode`.

`ack_nack_feedback_mode_ca`
: Optional enumeration: `cs`, `pucch3`. Select the ACK/NACK feedback mode when two serving cells are enabled (carrier aggregation). When more than two serving cells are enabled, `pucch3` is always used.

`n1_pucch_an_cs_count`
: Optional integer (default = 0). Select the number of PUCCH 1 resources used for PUCCH 1B channel selection. It is only useful when `ack_nack_feedback_mode_ca` is set to `cs`. This value limits the number of UEs which can be scheduled in the same TTI while doing carrier aggregation.

`n3_pucch_an_n_rb`
: Optional integer (default = 0). Select the number of UL resources blocks reserved for PUCCH 3 signalling. It is only necessary if PUCCH 3 ACK/NACK feedback is selected for carrier aggregation or TDD.

`pusch_dedicated`
: Object. PUSCH dedicated configuration. Currently shared by all UEs. The following properties are defined:

`beta_offset_ack_index`
: Range: 0 to 15.

`beta_offset_ri_index`
: Range: 0 to 12.

`beta_offset_cqi_index`
: Range: 2 to 15.

`pusch_msg3_delta_power`
: Optional. Range: -6 to 8 (default = 0). Relative power for Msg3 (=CCCH RRC Connection Request) in dB. It is internally rounded to an even value.

`pusch_msg3_mcs`
: Range: 0 to 15. MCS for Msg3 (=CCCH RRC Connection Request).

`pusch_mcs`
: Integer or array of 10 integers (range: -1 to 28). Force the PUSCH MCS (test feature). If an array is provided, it provides the PUSCH MCS for each subframe. Use -1 not to force the MCS in a given subframe.

`pusch_fixed_rb_alloc`
: Optional boolean or array of 10 booleans. Force fixed PUSCH RB allocation in all or a selected set of subframes. If an array is provided, a value true at the index value i of the array indicates that a fixed PUSCH RB allocation is used in subframe number i.

  The parameters `pusch_fixed_rb_start` and `pusch_fixed_l_crb` are used for the fixed allocation. `pusch_fixed_l_crb` must be of the form `2^n1*3^n2*5^n3`. PUSCH are allocated only if they don't overlap with PUCCH or PRACH, so care must be taken when defining the range. In some cases, PUSCH retranmissions may use other RBs.

`pusch_fixed_rb_start`
: Optional integer or array of 10 integers. First RB for fixed PUSCH allocation. If an array is provided, it gives the first RB for each subframe (see `pusch_fixed_rb_alloc`).

  For a cell configured for category M1 UEs, `pusch_fixed_rb_start` and `pusch_fixed_l_crb` give the allocation inside a narrow band (hence `pusch_fixed_rb_start + pusch_fixed_l_crb <=` 6).

`pusch_fixed_l_crb`
: Optional integer or array of 10 integers. Number of consecutive RBs for fixed PUSCH allocation. If an array is provided, it gives the number of consecutive RBs for each subframe (see `pusch_fixed_rb_alloc`).

`pusch_fixed_rb_forced`
: Optional boolean (default = false). If true, the eNodeB schedules the PUSCH with fixed RB allocation even if it collides with PUCCH/PRACH or another PUSCH.

`pusch_multi_cluster`
: Optional boolean (default = false). If true, enable multi-cluster PUSCH resource allocation for the UEs supporting it (release 10). Note: this is a UE test feature, so the multi cluster allocation is not optimized by the scheduler.

`pusch_max_mcs`
: Optional. Range: 0 to 28 (default = 28). CPU load limitation: maximum MCS allocated by the eNodeB for PUSCH. Smaller MCS give a smaller bitrate and a smaller CPU load.

`pusch_max_its`
: Optional. Range 1 to 20 (default = 10). CPU load limitation: set the maximum number of iterations of the turbo decoder. A higher value gives a lower frame error rate but a higher CPU load.

`force_full_bsr`
: Optional boolean (default = false). If true, the eNodeB considers the UE always indicates a full buffer size. Hence the UE is scheduled as often as possible for PUSCH transmission.

`force_dl_schedule`
: Optional boolean (default = false). If true, the eNodeB considers there is always DL data waiting for transmission. Hence the UE is scheduled as often as possible for PDSCH transmission.

`pdsch_mcs`
: Integer or array of 10 integers (range: -1 to 28). Force the PDSCH MCS (test feature). If an array is set, it provides the PDSCH MCS for each subframe. Use -1 not to force the MCS in a given subframe.

`pdsch_mcs_from_cqi`
: Integer or array of 16 integers (range: -1 to 28). Force the PDSCH MCS (test feature).
If an array is set, it provides the PDSCH MCS according to the CQI reported by UE. Use -1 not to force the MCS for a given CQI.

`pdsch_fixed_rb_alloc`
: Optional boolean or array of 10 booleans. Force fixed PDSCH RB allocation using the parameters `pdsch_fixed_rb_start` and `pdsch_fixed_l_crb`. If an array is provided, it selects the fixed PDSCH allocation for each subframe.

  For a cell configured for category M1 UEs, fixed PDSCH RB allocation is only possible in subframes where the PDSCH MCS is fixed (see `pdsch_mcs`).

`pdsch_fixed_rb_start`
: Optional integer or array of 10 integers. First RB for fixed PDSCH allocation (see `pdsch_fixed_rb_alloc`). If an array is provided, it provides the first RB for each subframe.

For a cell configured for category M1 UEs, `pdsch_fixed_rb_start` and `pdsch_fixed_l_crb` give the allocation inside a narrow band (hence `pdsch_fixed_rb_start + pdsch_fixed_l_crb <= 6`).

`pdsch_fixed_l_crb`
: Optional integer or array of 10 integers. Number of consecutive RBs for fixed PDSCH allocation (see `pdsch_fixed_rb_alloc`). If an array is provided, it provides the consecutive RBs for each subframe.

`rrc_procedure_filter`
: Optional object. Allows to define the eNB behavior for a list of RRC procedures. Each property name represents a RRC procedure. The ones currently supported are `rrc_connection_request` and `rrc_connection_reestablishment_request`. Each property value is an enum: `treat` (UE message is processed), `ignore` (UE message is ignored) or `reject` (UE message is rejected).
By default all procedures are treated.

Example:

```
rrc_procedure_filter: {
    rrc_connection_request: "treat",
    rrc_connection_reestablishment_request: "reject"
}
```

`rach_ignore_count`
: Optional integer. Indicates how many consecutive RACH attempts are ignored by the eNB.

`transmission_mode`
: Optional. Range: 1 to 6 (default = 1). Set the DL transmission mode (same for all UEs). The values of 1 and 2 are equivalent and automatically adjusted to 1 or 2 depending on the number of DL antennas. The corresponding transmission modes are:

`1`	Single antenna port.
`2`	Transmit diversity.
`3`	Large delay CDD.
`4`	Closed-loop spatial multiplexing.
`5`	Multi-user MIMO.
`6`	Closed-loop spatial multiplexing using single transmission layer.

Notes:

- Transmission modes 2 to 6 are only usable when `n_antenna_pbch >= 2` (more than one DL antenna).
- Transmission modes 3 and 4 need rank indicator reporting for proper operation (see the `m_ri` parameter).
- The current MAC scheduler does not schedule several UE at the same time when using transmission mode 5.

`codebook_subset_restriction`
: Optional string. Bit string giving the allowed code book indexes for transmission modes 3, 4, 5, 6. The number of bits is given by TS.36 213 table 7.2-1b. The default value is all ones (i.e. all code book indexes are allowed).

`transmission_mode_opt`
: Optional integer (default = 0). Range: 0 or 7 to 10. If the UE supports the indicated transmission mode, it is enabled with the first RRC connection reconfiguration. The value 0 is used to keep the initial transmission mode selected by `transmission_mode`. The available optional transmission modes are:

  `7` Antenna port 5 (UE specific, release 8).

  `8` Dual layers, antenna ports 7 and 8 (UE specific, release 9).

  `9` Up to 8 layers, antenna ports 7 to 14 (UE specific, release 10).

  `10` Up to 8 layers, antenna ports 7 to 14 (UE specific, CoMP, release 11).

  The transmission modes 8, 9 and 10 require at least two DL antennas and need rank indicator reporting for proper operation (see the `m_ri` parameter). Moreover transmission modes 9 and 10 need a proper CSI-RS configuration.

`codebook_subset_restriction_opt`
: Optional string. Bit string giving the allowed code book indexes for transmission modes 8, 9 or 10. The number of bits depends on the selected transmission mode and number of DL antennas:

  `tm8, 2 antennas:`
  6 bits

  `tm8, 4 antennas:`
  32 bits

  `tm9 or tm10, 2 antennas:`
  6 bits

  `tm9 or tm10, 4 antennas:`
  64 bits

  `tm9 or tm10, 8 antennas:`
  109 bits

`n_scid` Optional integer (default = 0). Range 0 to 1. Force the scrambling identifier when antenna ports 7 or 8 are used.

`ue_specific_port`
: Optional integer (default = 7). Range 7 to 8. When single layer transmission is used with transmission mode 8, force the corresponding antenna port.

`csi_rs_nzp`
: Optional object. Specifies the Non-Zero Power Channel-State Information Reference Signals (CSI-RS) sent by the eNodeB for release 10 UEs. The following fields are defined:

  `period` Enumeration: 5, 10, 20, 40, 80. Period (in ms) of the CSI-RS.

  `offset` Range: 0 to period - 1. Offset (in ms) of the CSI-RS.

  `n_antenna`
  Integer: 1, 2, 4 or 8. Must be less than the number of DL antennas.

  `resource_config`
  Integer. Selected CSI-RS resource configuration. The exact range depends on the selected cyclic prefix and frame structure. See tables 6.10.5.2-1 and 6.10.5.2-2 from TS 36.211.

`p_c` Range: -8 to 15. Relative power in dB compared to the cell specific reference signal.

`csi_rs_zp`
Optional object. Specifies the Zero Power Channel-State Information Reference Signals reserved by the eNodeB for release 10 UEs. The following fields are defined:

`period` Enumeration: 5, 10, 20, 40, 80. Period (in ms) of the CSI-RS ZP.

`offset` Range: 0 to period - 1. Offset (in ms) of the CSI-RS ZP.

`resource_config_list`
Range: 0 to 65535. Bit mask of the selected zero CSI-RS ZP configurations. The first configuration is in bit 15. The corresponding configurations are given in tables 6.10.5.2-1 and 6.10.5.2-2 from TS 36.211 (column with 4 antennas).

`csi_rs_im`
Optional object. Specifies the Channel-State Information Reference Signals reserved by the eNodeB for Interference Measurement for release 11 UEs (CSI-RS IM). The following fields are defined:

`period` Enumeration: 5, 10, 20, 40, 80. Period (in ms) of the CSI-RS IM.

`offset` Range: 0 to period - 1. Offset (in ms) of the CSI-RS IM.

`resource_config`
Integer. Selected CSI-RS IM resource configuration. The exact range depends on the selected cyclic prefix and frame structure. See tables 6.10.5.2-1 and 6.10.5.2-2 from TS 36.211.

The CSI-RS IM must completely overlap with the configured CSI-RS ZP.

`dl_256qam`
Optional boolean (default = false). If true, allow 256QAM DL support for the UE supporting it (release 12).

`sr_period`
Enumeration: 5, 10, 20, 40, 80, 2, 1, 0. Scheduling Request period in ms. Currently when `half_duplex_ue` is true it must be >= 40. The special value 0 means that no Scheduling Request resource is allocated hence the UE uses a PRACH instead.

`dsr_trans_max`
Optional enumeration: 4, 8, 16, 32, 64 (default = 64). Set the dsr-TransMax parameter (maximum number of scheduling request transmissions).

`cqi_period`
Enumeration: 2, 5, 10, 20, 40, 80, 160, 1, 32, 64, 128. CQI/PMI report period in ms. Currently when `half_duplex_ue` is true it must be >= 32.

`m_ri` Optional enumeration: 0, 1, 2, 4, 8, 16, 32 (default = 0). If different from zero, Rank Indicator (RI) reporting is done every `m_ri` CQI/PMI reports. RI should only be used with transmission modes 3, 4, 8, 9 and 10.

`ap_cqi_period`
Optional integer (default = 0). Approximate period (in ms) for the aperiodic CQI reporting. 0 indicates that aperiodic CQI reporting is disabled. Note: aperiodic CQI is currently not supported with carrier aggregation.

`ap_cqi_rm`
Optional enumeration: rm12, rm20, rm22, rm30, rm31. Aperiodic CQI reporting mode.

`simultaneousAckNackAndCQI`
: Optional boolean (default = true). If true, enable simultaneous ACK/NACK and CQI reporting. With normal cyclic prefix, PUCCH format 2A/2B are used.

`simultaneousAckNackAndCQI_format3`
: Optional boolean (default = false). If true, enable simultaneous ACK/NACK and CQI reporting with PUCCH format 3 (release 11).

`srs_dedicated`
: Object. SRS configuration. Currently the same for all UEs except for `srs-ConfigIndex` and `freqDomainPosition` which are dynamically allocated for each UE. The following properties are defined:

  `srs_period`
  : Enumeration: 2, 5, 10, 20, 40, 80, 160, 320. SRS period in ms. Currently when `half_duplex_ue` is true it must be >= 40.

  `srs_bandwidth`
  : Range: 0 to 3. SRS bandwidth.

  `srs_hopping_bandwidth`
  : Range: 0 to 3. SRS hopping bandwidth.

`mac_config`
: Object. MAC configuration. Currently the same for all UEs. The following properties are defined:

  `ul_max_harq_tx`
  : Maximum number of HARQ transmissions for uplink.

  `dl_max_harq_tx`
  : Maximum number of HARQ transmissions for downlink.

  `ul_max_consecutive_retx`
  : Optional Integer (default = 30). Maximum number of UL retransmissions after which the UE is disconnected.

  `dl_max_consecutive_retx`
  : Optional Integer (default = 30). Maximum number of DL retransmissions after which the UE is disconnected.

  `time_alignment_tx_timer`
  : Optional integer from 0 to 10240 (default = 500). Transmit the UL time alignment information every `time_alignment_tx_timer` ms. The value 0 means infinity.

  `time_alignment_timer_dedicated`
  : Optional integer (default = 0). Time alignment timer dedicated. 0 means infinity. Note: `time_alignment_tx_timer` must be used to set the UL time alignment transmission period.

  `periodic_bsr_timer`
  : Optional integer (default = 20). Periodic BSR timer value.

  `retx_bsr_timer`
  : Optional integer (default = 320). Retransmission BSR timer value.

  `periodic_phr_timer`
  : Optional integer (default = 500). Periodic PHR timer value.

  `prohibit_phr_timer`
  : Optional integer (default = 200). Prohibit PHR timer value.

`dl_path_loss_change`
: Optional enumeration: `dB1`, `dB3`, `dB6`, `infinity` (default = `dB3`). DL path loss change value.

`drx_config`
: Optional object. If present, configure the DRX parameters. The following properties are defined:

  `on_duration_timer`
  : Range: 1 to 200. DRX on duration timer (in PDCCH subframes). If the value is small, it may be necessary to disallow half duplex UE from connecting to the eNodeB (set `half_duplex_ue` to `false`) in order to relax the constraints on the allocation of SRS/CQI/SR.

  `drx_inactivity_timer`
  : Range: 1 to 2560. DRX inactivity timer (in PDCCH subframes).

  `drx_retransmission_timer`
  : Range: 1 to 33. DRX retransmission timer (in PDCCH subframes).

  `long_drx_cycle`
  : Range: 10 to 2560. Duration of the long DRX cycle (in subframes). Must be a multiple of `mas_gap_period`.

  `short_drx_cycle`
  : Optional. Range: 2 to 640. If present, configuration the duration of the short DRX cycle (in subframes). `long_drx_cycle` must be a multiple of `short_drx_cycle`.

  `drx_short_cycle_timer`
  : Optional. Range: 1 to 16. If the short DRX cycle is configured, set the short DRX cycle timer.

`data_inactivity_timer`
: Integer. Value in seconds of the data inactivity monitoring timer. 0 means that the timer is deactivated.

`sr_prohibit_timer`
: Optional integer. Timer in number of SR periods used to delay the transmission of a Scheduling Request.

`logical_channel_sr_prohibit_timer`
: Optional integer. Timer in number of subframes used to delay the transmission of a Scheduling Request for logical channels enabled by the `logicalChannelSR_Prohibit` parameter in `drb_config` object.

`cyclic_shift_dci`
: Optional. Range: 0 to 7 (default = 0). Set the DCI 0 cyclic_shift_dci parameter.

`dpc`
: Optional boolean (default = false). Enable dynamic UE power control.

`dpc_pusch_snr_target`
: Optional float. Must be present if `dpc` is true. Set the PUSCH SNR target for the dynamic UE power control.

`dpc_pucch_snr_target`
: Optional float. Must be present if `dpc` is true. Set the PUCCH SNR target for the dynamic UE power control.

`p_srs_offset`
: Optional. Range 0 to 15 (default = 3). SRS power offset. The configured value is `-10.5 + 1.5 * p_srs_offset` dB.

`snr_to_mcs_offset`
: Optional float. This offset is added to the estimated uplink SNR to compute the PUSCH MCS. The default value depends on the eNodeB configuration.

`ul_snr_adapt_fer`
: Optional float (default = 0.01). This value defines the UL PER targeted by the eNB link adaptation algorithm. By default it applies an error rate of 1%.

`cqi_adapt_fer`
: Optional float (default = 0.01). This value defines the DL PER targeted by the eNB link adaptation algorithm. By default it applies an error rate of 1%.

`cipher_algo_pref`
: Array of integers. Set the preferred algorithms for RRC and User Plane encryption in decreasing order of preference. If none match the UE capabilities, then EEA0 (no encryption) is selected.

  List of supported algorithms:

  `1` EEA1 (Snow 3G)

  `2` EEA2 (128 bit AES)

  `3` EEA3 (ZUC)

  If encryption is necessary, for best performance use AES (EEA2) as first choice if your CPU supports the AES NI Intel instruction set (use the `hwcaps` monitor command and see if AES is displayed). Otherwise use Snow3G (EEA1) or ZUC (EEA3).
  Note that ciphering is subject to export rules depending on your country.

`integ_algo_pref`
: Array of integers. Set the preferred algorithms for RRC integrity check in decreasing order of preference. If none match the UE capabilities, then EIA0 (no integrity check) is selected.

  List of supported algorithms:

  `1` EIA1 (Snow 3G)

  `2` EIA2 (128 bit AES)

  `3` EIA3 (ZUC)

  For best performance, use AES (EIA2) as first choice if your CPU supports the AES NI Intel instruction set (use the `hwcaps` monitor command and see if AES is displayed). Otherwise use Snow3G (EIA1) or ZUC (EIA3).

`inactivity_timer`
: Integer. Send RRC connection release after this time (in ms) of network inactivity.

`srb_config`
: Optional array of objects. Allows to override some parameters of the default configuration specified in 3GPP 36.331 chapter 9.2.1. If unset, the eNB will already change maxRetxThreshold value to 32, t-Reordering value to 45 ms and t-PollRetransmit to 60 ms. Each object contains the following fields:

  `id` Integer: 1 or 2. Contains the SRB identity.

`maxRetxThreshold`
: Enumeration: 1, 2, 3, 4, 6, 8, 16, 32 (default 32). maxRetxThreshold value.

`t_Reordering`
: Enumeration: 0, 5, 10, 15, 20, 25, 30, 35, 40, 45, 50, 55, 60, 65, 70, 75, 80, 85, 90, 95, 100, 110, 120, 130, 140, 150, 160, 170, 180, 190, 200, 1600 (default 45). t-Reordering timer value in ms.

`t_PollRetransmit`
: Enumeration: 5, 10, 15, 20, 25, 30, 35, 40, 45, 50, 55, 60, 65, 70, 75, 80, 85, 90, 95, 100, 105, 110, 115, 120, 125, 130, 135, 140, 145, 150, 155, 160, 165, 170, 175, 180, 185, 190, 195, 200, 205, 210, 215, 220, 225, 230, 235, 240, 245, 250, 300, 350, 400, 450, 500, 800, 1000, 2000, 4000 (default 60). t-PollRetransmit timer value in ms.

`drb_config`
: String or Array. Array of objects containing the DRB configuration for each QCI value. If a string is given, the array is read from the corresponding filename. See [DRB configuration], page 55.

`meas_config`
: Optional string. Filename of the textual ASN.1 context (GSER syntax) of the `measConfig` field of the `RRCConnectionReconfiguration` message (see TS 36.331). It is used to set the parameters of the RRC measurements. If no filename is given, no `measConfig` field is transmitted to the UEs.

`meas_config_desc`
: Optional object. If present, and if `meas_config` object is not present, eNB will dynamically build the measurement configuration sent to the UE based on the content of this object and the list of neighbour cells defined in `ncell_list` object. It will create A1 and A2 events for the serving cell (if inter frequencies neighbour cells exist, or if intra frequency neighbour cells exist for a BR UE), and an A3 event for each serving and neighbour frequencies. At the beginning, gaps are not activated. When A2 event report is triggered, if `meas_gap_config` is set to gp0 or gp1, gaps are activated. When A1 event report is triggered, gaps are released. This object contains the following fields:

  `a1_report_type`
  : Enumeration, rsrp or rsrq. Defines the measurement type requested for the A1 report.

  `a1_rsrp` Integer, range from -140 to -43. RSRSP threshold value in dBm. Used if `a1_report_type` is set to rsrp.

  `a1_rsrq` Integer, range from -40 to -6. RSRQ threshold value in 0.5dB steps. Used if `a1_report_type` is set to rsrq.

  `a1_hysteresis`
  : Integer, range from 0 to 30. A1 hysteresis in 0.5dB steps used for the measurement report triggering condition.

  `a1_time_to_trigger`
  : Enumeration: 0, 40, 64, 80, 100, 128, 160, 256, 320, 480, 512, 640, 1024, 1280, 2560 or 5120. Time in ms during which the A1 event condition must be met before triggering the measurement report.

`a2_report_type`
: Enumeration, rsrp or rsrq. Defines the measurement type requested for the A2 report.

`a2_rsrp`
: Integer, range from -140 to -43. RSRSP threshold value in dBm. Used if `a2_report_type` is set to rsrp.

`a2_rsrq`
: Integer, range from -40 to -6. RSRQ threshold value in 0.5dB steps. Used if `a2_report_type` is set to rsrq.

`a2_hysteresis`
: Integer, range from 0 to 30. A2 hysteresis in 0.5dB steps used for the measurement report triggering condition.

`a2_time_to_trigger`
: Enumeration: 0, 40, 64, 80, 100, 128, 160, 256, 320, 480, 512, 640, 1024, 1280, 2560 or 5120. Time in ms during which the A2 event condition must be met before triggering the measurement report.

`a3_report_type`
: Enumeration, rsrp or rsrq. Defines the measurement type requested for the A3 report.

`a3_offset`
: Integer, range from -30 to 30. A3 offset in 0.5dB steps used for the measurement report triggering condition.

`a3_hysteresis`
: Integer, range from 0 to 30. A3 hysteresis in 0.5dB steps used for the measurement report triggering condition.

`a3_time_to_trigger`
: Enumeration: 0, 40, 64, 80, 100, 128, 160, 256, 320, 480, 512, 640, 1024, 1280, 2560 or 5120. Time in ms during which the A3 event condition must be met before triggering the measurement report.

`a3_force_meas_id_on_pcell_earfcn`
: Optional boolean (default = false). Forces an A3 measurement identity for the primary cell even if no neighbour intra frequency cells are declared in `ncell_list` object.

`rsrp_filter_coeff`
: Optional enumeration: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 11, 13,15, 17 or 19 (default = 4). Coefficient used for the RSRP layer 3 filtering done in RRC (see 3GPP 36.331 5.5.3.2 for details).

`rsrq_filter_coeff`
: Optional enumeration: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 11, 13,15, 17 or 19 (default = 4). Coefficient used for the RSRQ layer 3 filtering done in RRC (see 3GPP 36.331 5.5.3.2 for details).

`meas_gap_config`
: Optional enumeration: none, gp0, gp1. Configuration of the measurement gap. Must be provided if `meas_config` or `meas_config_desc` is present.

`ho_from_meas`
: Optional boolean (default = false). If true, the eNodeB triggers a handover when an A3 or A5 RRC measurement event is received from the UE, or when a periodical measurement indicates a neighbour cell RSRP higher than or equal to the serving cell RSRP.

`t304` Enumeration: 50, 100, 150, 200, 500, 1000, 2000 (default = 1000). T304 timer for handover.

`pws_max_segment_len`
Optional integer (default = 32). Set the maximum CMAS/ETWS message segment length in byte. It is needed in order to limit the size of the corresponding SIB messages.

`pws_si_periodicity`
Enumeration: 8, 16, 32, 64, 128, 256, 512 (default = 16). Set the periodicity (in frames) of the transmission of the CMAS/ETWS SIB messages.

`prs` Optional object. Contains the optional Positioning Reference Signals (PRS) configuration. If not present, no PRS are generated. PRS parameters are defined in TS 36.211 and TS 36.355.

`prs_bandwidth`
Integer. Bandwidth (in Resource Blocks) of the PRS. From 6 to `n_rb_dl`.

`prs_period`
Enumeration: 160, 320, 640, 1280. Give the periodicity (in subframes) of the PRS.

`prs_offset`
Integer (0 to prs_period - 1). Give the time offset of the PRS.

`numdl_frames`
Integer: 1, 2, 4 or 6. Number of consecutive subframes in which the PRS are sent.

`prs_muting_info`
Optional string. Bit string containing the PRS muting pattern. Its length must be 2, 4, 8 or 16.

`power_prs`
Optional float (default = 0). Relative power in dB of the PRS.

`mbms` Optional object. MBMS configuration. See [MBMS configuration], page 58.

`sib16_enable`
Optional boolean (default = false). If true, enable SIB16 (time information broadcast). Note: the broadcasted UTC is currently taken from the eNB internal time (see `internal_time_ref` parameter), so it might not be GPS accurate. The time zone and daylight saving time are taken from the system time.

`sib16_si_periodicity`
Enumeration: 8, 16, 32, 64, 128, 256, 512 (default = 32). Set the periodicity (in frames) of the transmission of the SIB16 messages.

`ueinfo_extension`
Optional boolean (default = false). If set, eNB will send UE information within S1AP initial UE message.
The informations are presented as a S1AP-PROTOCOL-IES item in InitialUEMessage with an ID = 1000.
Its ASN.1 definition is:

```
UEInformationExtension ::= SEQUENCE {
    timing-advance      INTEGER,
    snr                 INTEGER (0..255)
```

```
}
```

Where:

`timing advance`
: UE timing advance expressed in unit of TS

`snr`
: 8 bit value representing SNR in range of -63.5 to +64 dB by step of 0.5 dB (i.e 0 is -63.5 dB and 255 is 64 dB).

Example of EU initial message:

```
initiatingMessage: {
  procedureCode id-initialUEMessage,
  criticality ignore,
  value {
    protocolIEs {
      {
        id id-eNB-UE-S1AP-ID,
        criticality reject,
        value 1
      },
      ...
      {
        id 1000,
        criticality ignore,
        value {
          timing-advance 1,
          snr 169
        }
      }
    }
  }
}
```

`rrc_redirect`
: Array of strings or objects. Each string is the filename of the textual ASN.1 content (GSER syntax) of a redirection information.
These will define redirection parameter within RRC Connection Release sent by eNB to the UE (cf 3GPP TS 25.331)
To send this redirection, you need to send to eNB a S1AP DownlinkNASTransport message and add a S1AP-PROTOCOL-IE item with an ID of 1001 (Please refer to MME documentation and `attach_reject_filter` parameter to use it).
Its ASN.1 definition is:

```
UERedirectExtension ::= SEQUENCE {
    type                INTEGER (0..255)
}
```

Where:

`type`
: Index of the redirection configuration in the rrc_redirect array.

Here is an example of the incoming downlink NAS transport message:

```
initiatingMessage: {
  procedureCode id-downlinkNASTransport,
  criticality ignore,
  value {
```

```
            protocolIEs {
              ...
              {
                id 1001,
                criticality ignore,
                value {
                  type 0
                }
              }
            }
          }
        }
```

If *rrc_redirect* is the following: `["redirect.asn"]`.
And *redirect.asn* is:

```
geran: {
    startingARFCN  10,
    bandIndicator  dcs1800,
    followingARFCNs explicitListOfARFCNs: {12, 42}
}
```

The UE will be sent the following RRC connection release message:

```
        {
          message c1: rrcConnectionRelease: {
            rrc-TransactionIdentifier 0,
            criticalExtensions c1: rrcConnectionRelease-r8: {
              releaseCause other,
              redirectedCarrierInfo geran: {
                startingARFCN 10,
                bandIndicator dcs1800,
                followingARFCNs explicitListOfARFCNs: {
                  12,
                  42
                }
              }
            }
          }
        }
```

Alternatively, *rrc_redirect* can be an array of objects. Each object contains the following fields:

`plmn` String. PLMN of the location area.

`lac` Range 0 to 0xffff. LAC of the location area.

`filename` String. Filename of the textual ASN.1 content (GSER syntax) of a redirection information.

The redirection is initiated with a CS fallback indication in the UE context modification request S1 message. The Location Area Information (PLMN and LAC) is used to select the redirection information sent in the RRC connection release. If the Location Area Information is not present, the first redirection is used.

`scell_list`

Optional array of objects. List the cells of the same eNodeB which can be used for carrier aggregation. Each object contains the following fields:

`cell_id` Range: 0 to 1023. Low 7 bit (long macro eNB), 8 bit (macro eNB) or 10 bit (short macro eNB) of the cell identifier.

`cross_carrier_scheduling`
Boolean. True if cross carrier scheduling is enabled for this cell.

`scheduling_cell_id`
Range: 0 to 1023. If cross carrier scheduling is enabled, gives the cell id in which the corresponding PDCCH is sent.

`ul_allowed`
Optional boolean (default = false). If true, enable uplink for this serving cell.

`ue_cap_rat_type`
Optional array of strings. List the RAT types (`eutra`, `utra`, `geran-cs`, `geran-ps`, `cdma2000-1XRTT`) for the RRC UE capability enquiry message. `eutra` is always included.

`gbr_ul_ratio`
Optional float (default = 0.8). Maximum ratio of the uplink resources that can be reserved for GBR ERABs.

`gbr_dl_ratio`
Optional float (default = 0.8). Maximum ratio of the downlink resources that can be reserved for GBR ERABs.

`gbr_init_ul_bits_per_re`
Optional float (default = 2.0). The GBR ERAB resources are measured in terms of resource elements (RE) per second. Each RE can be assigned a given number of bits depending on the exact radio conditions. This parameter gives the initial number of bits per uplink RE when the UE is connecting (in this case no reliable radio quality measurement is available).

`gbr_init_dl_bits_per_re`
Optional float (default = 2.0). Same as `gbr_init_ul_bits_per_re` for downlink.

`ue_count_max`
Optional integer (default = 500). Maximum number of UEs (for this cell).

`erab_count_max`
Optional integer (default = 1500). Maximum number of ERABs (for this cell).

`rrc_cnx_reject_waitTime`
Optional integer (default = 10). RRC connection reject wait time in seconds.

`rrc_cnx_reject_extWaitTime`
Optional integer (default = 0). RRC connection reject extended wait time in seconds.

`rrc_cnx_reject_deprioritisation`
Optional object. If present, the deprioritisationReq-r11 field is added to the RRC Connection Reject message.
The object must contain the following fields:

`type` Enumeration ("none", "frequency" or "e-utra").

`timer` Optional enumeration (5, 10, 15 or 30). Timer in minutes. Required if `type` is not none.

`rrc_cnx_release_extWaitTime`
: Optional integer (default = 0). RRC connection release extended wait time in seconds.

`ims_emergency_support`
: Optional boolean (default = false). If true, IMS emergency support is advertised in SIB1.

8.5.3 Test parameters

The following cell parameters are only useful when the eNodeB is connected to a specific measurement equipment. They cannot normally be used with normal UEs.

`sib_enable`
: Optional boolean (default = true). If false, disable the transmission of the SIBs.

`pdcch_fill`
: Optional boolean (default = false). If true, add dummy PDCCHs filling the available PDCCH resources. For 1.4 and 3 bandwidths, PDCCHs of 1 CCE are added. For the other bandwidths, PDCCHs of 2 CCEs are added.

`phich_fill`
: Optional boolean (default = false). If true, add dummy PHICH filling the available PHICH resources. 2 PHICH are added per group with HI=0 with sequence numbers 0 and 4 for normal cyclic and sequence numbers 0 and 2 for extended cyclic prefix.

`forced_ri`
: Optional integer. Range 0 to 8 (default = 0). If >= 1, use it as Rank Indicator (RI) returned by the UE.

`forced_cqi`
: Optional integer. Range -1 to 15. (default = -1). If >= 0, use it as Channel Quality Indicator (CQI) returned by the UE.

`test_mode`
: Optional object. Enable specific test modes where UE contexts are automatically created when starting the eNodeB. The `type` property selects the test mode:

  `pusch`
  : Enables continuous reception of PUSCH by the eNodeB. DCI 0 and PHICH are transmitted. The following additional properties are available:

    `rnti` Integer. Range 0 to 65535. Select the PUSCH RNTI.

    `pusch_retx`
    : Boolean. If false, don't force the UE to retransmit in case of error.

    `pusch_external_harq_ack`
    : Optional boolean (default = false). If true, transmit the PUSCH HARQ ACK/NACK and timing advance information to the TRX driver so that it can be transmitted to an external signal generator. The HARQ ACK/NACK signal is transmitted at the same time as PHICH (hence at PUSCH TTI + 4 in FDD mode). The timing advance information is transmitted at the same time as the corresponding PDSCH. The timing advance transmission period is set with the `time_alignment_tx_timer` parameter.

The cell properties `pdcch_format`, `pusch_fixed_rb_alloc`, `pusch_mcs` can be used to force specific PUSCH parameters.

`pdsch`

Enables continuous transmission of PDSCH. The PDSCH payload contains valid data with PDCP packets of constant length. DCI are transmitted according to the selected transmission mode. PUCCH are received. The following additional properties are available:

`rnti` Integer. Range 0 to 65535. Select the PDSCH RNTI.

`pdsch_retx`
Boolean. If false, don't retransmit the unacknowledged PDSCH (hence PUCCH ACK/NACK are ignored).

The cell properties `pdcch_format`, `pdsch_mcs`, `forced_ri`, `forced_cqi`, `transmission_mode`, `dl_256qam` can be used to force specific PDSCH parameters.

`load`

CPU load test. Several UEs are instanciated and all are transmitting and receiving at the same time. The following additional properties are available:

`ue_count` Integer. Set the number of UE contexts.

`pusch_fer`
Float. Range 0 to 1. Set the simulated PUSCH Frame Error Rate.

`pdsch_fer`
Float. Range 0 to 1. Set the simulated PDSCH Frame Error Rate.

The cell properties `pusch_mcs`, `forced_ri`, `forced_cqi` can be used to set the simulated radio conditions.

8.5.4 Bandwidth Reduced parameters (Category M1)

The following parameters configure the cell to allow the connection of Bandwidth-Reduced UEs (category M1). All the parameters are in the `br_ue` object. Bandwidth-reduced specific SIB configuration files must be used except for SIB1.

`br_only` Optional boolean (default = false). If true, only category M1 UEs are allowed in this cell. The legacy LTE SIBs are disabled and no legacy LTE resources are allocated. 1.4 and 3 MHz category M1 cells must use `br_only=true`.

`br_root_sequence_index`
Range: 0 to 837. Set the BR PRACH root sequence index. It must be different for each neighbour cell operating on the same frequency and sharing the same PRACH configuration.

`br_r_sib1`
Enumeration: 1, 2 or 4. Number of SIB1 BR repetitions per 20 ms (1, 2 or 4).

`br_tbs_sib1`
Optional enumeration: 26, 32, 41, 63, 89, 117. SIB1 BR size in bytes. If not set, the size is automatically computed by the eNB based on the SIBs defined in the configuration file. It can be useful to set it manually in case new SIBs are scheduled during runtime (like SIB10, 11, 12 or 14).

`br_si_window_length`
: Enumeration: 20, 40, 60, 80, 120, 160, 200. BR SI window length in ms.

`br_si_repetition_pattern`
: Enumeration: 1, 2, 4, 8. SI repetition pattern (one every n Radio Frames)

`br_sib_sched_list`
: Array of object. Each object contains the content of one SI scheduling slot (the first slot must contain the SIB2):

  `si_periodicity`
  : Enumeration: 8, 16, 32, 64, 128, 256, 512. SI periodicity in Radio Frames.

  `filename` Filename containing the SIBs. The content is in textual ASN.1 (GSER syntax).

`br_distributed_mpdcch_precoding_matrix`
: Optional complex matrix. Set the distributed MPDCCH precoding matrix. It has `n_antenna_dl` rows and 2 columns.

`br_coverage_levels`
: Array of objects. Configuration of each coverage level. There must be the same number of coverage levels as PRACH configurations in the SIB2. Since only CE mode A is currently supported, at most 2 coverage levels can be specified. For each coverage level, the following parameters are available:

  `br_rar_coderate`
  : Float. Maximum code rate for the Random Access Response (RAR).

  `br_mpdcch_css_ra_al`
  : Enumeration: 8, 16, 24. MPDCCH aggregation level for the Common Search Space for the RAR message.

  `br_mpdcch_css_n_rep`
  : Integer. Range: 1 to 255. Number of repetitions for the Common Search Space MPDCCH.

  `br_pusch_msg3_mcs`
  : Range: 0 to 7. MCS for Msg3 (=CCCH RRC Connection Request).

  `br_mpdcch_n_rb`
  : Enumeration: 2, 4, 6. Number of PRBs for the UE specific MPDCCH.

  `br_mpdcch_tm_type`
  : Enumeration: distributed or localized. Set the UE specific MPDCCH transmission mode. The localized transmission mode relies on the PMI reports from the UE, so it is normally used only with transmission modes 6 or 9.

  `br_mpdcch_al`
  : Enumeration: 2, 4, 8, 16, 24. Aggregation level for the UE specific MPDCCH (it is currently statically configured).

  `br_mpdcch_n_rep_max`
  : Integer. Range: 1 to 256. Maximum number of repetitions for the UE specific MPDCCH.

  `br_mpdcch_n_rep`
  : Integer. Range: 1 to br_mpdcch_n_rep_max. Number of repetitions for the UE specific MPDCCH.

`br_mpdcch_start_sf`
: Float. Range: 1 to 10. Starting subframe value for the UE specific MPDCCH.

`br_initial_cqi`
: Range: 4 to 10. Initial CQI for BR UEs (used until the first CQI is received). It cannot currently be lower than 4 because no repetition is possible for CCCH.

`br_pdsch_n_rep`
: Integer. Range: 1 to 32. Number of repetitions for PDSCH (for normal UE data and RAR).

`br_pusch_n_rep`
: Integer. Range: 1 to 32. Number of repetitions for PUSCH (for MSG3 and normal UE data).

`br_paging_mcs`
: Integer. Range: 0 to 7. MCS used for paging messages.

`br_mpdcch_paging_n_rep`
: Integer. Range: 1 to 256. Number of repetition for the paging MPDCCH.

`br_paging_n_rep`
: Integer. Range: 1 to 32. Number of repetitions for the paging message (PDSCH).

`br_n1_pucch_sr_count`
: Range: 1 to 1000. Number of Scheduling Request PUCCH resources reserved for BR UE.

`br_cqi_pucch_n_rb`
: Range: 2 to `n_rb_ul`. Number of resources blocks reserved for CQI reporting thru PUCCH for BR UE. Must be even.

`br_mpdcch_ue_count`
: Integer >= 1. Maximum number of UEs assigned to a single MPDCCH resource.

`br_t304` Optional enumeration: 50, 100, 150, 200, 500, 1000, 2000, 10000 (default = `t304` value). T304 timer for handover.

`br_srs_enabled`
: Optional boolean (default = false). Enable SRS for the BR UEs.

`br_hdfdd_pattern`
: Optional integer (default = 0). Range: 0 to 1. Select the HD-FDD uplink/downlink pattern used by the eNodeB for half duplex BR UEs. The pattern 0 is `UUU-DDDDD-` (10 ms period). The pattern 1 is `UUU-DDD-` (8 ms period).

`br_forced_mpdcch_nb_idx`
: Optional integer (default = -1). Forces the narrow band index used for MPDCCH. The value -1 means that the eNB selects the narrow band automatically.

`br_forced_pdsch_nb_idx`
: Optional integer (default = -1). Forces the narrow band index used for PDSCH. The value -1 means that the eNB selects the narrow band automatically.

`br_forced_pusch_nb_idx`
: Optional integer (default = -1). Forces the narrow band index used for PUSCH. The value -1 means that the eNB selects the narrow band automatically.

8.6 DRB configuration

Array of objects giving the Data Radio Bearer configuration for each QCI (QoS Class Identifier). There must be at least one definition for QCI = 9 which is the default QCI.

Each object contains the following properties:

`qci` Range: 1 to 255. The following parameters apply to DRBs of this QCI.

`rlc_config`
Object. Gives the RLC configuration. If UM (Unacknowledged Mode) is used, the `ul_um` and `dl_um` objects must be present. If AM (Acknowledged Mode) is used, the `ul_am` and `dl_am` objects must be present.

`ul_um` Object. Uplink RLC UM configuration.

`sn_FieldLength`
Enumeration: 5, 10. Sequence number field length in bits.

`dl_um` Object. Downlink RLC UM configuration.

`sn_FieldLength`
Enumeration: 5, 10. Sequence number field length in bits.

`t_Reordering`
Enumeration: 0, 5, 10, 15, 20, 25, 30, 35, 40, 45, 50, 55, 60, 65, 70, 75, 80, 85, 90, 95, 100, 110, 120, 130, 140, 150, 160, 170, 180, 190, 200, 1600. t_Reordering timer value in ms.

`ul_am` Object. Uplink RLC AM configuration.

`t_PollRetransmit`
Enumeration: 5, 10, 15, 20, 25, 30, 35, 40, 45, 50, 55, 60, 65, 70, 75, 80, 85, 90, 95, 100, 105, 110, 115, 120, 125, 130, 135, 140, 145, 150, 155, 160, 165, 170, 175, 180, 185, 190, 195, 200, 205, 210, 215, 220, 225, 230, 235, 240, 245, 250, 300, 350, 400, 450, 500, 800, 1000, 2000, 4000. t_PollRetransmit timer value in ms.

`pollPDU` Enumeration: 4, 8, 16, 32, 64, 128, 256, 0. pollPDU value.

`pollByte` Enumeration: 25, 50, 75, 100, 125, 250, 375, 500, 750, 1000, 1250, 1500, 2000, 3000, 0. pollByte value in kBytes. 0 means infinity.

`maxRetxThreshold`
Enumeration: 1, 2, 3, 4, 6, 8, 16, 32. maxRetxThreshold value.

`ul_extended_RLC_LI_Field_r12`
Optional boolean. If set to true and supported by the UE, a 15 bits LI will be used.

`ul_extended_RLC_AM_SN_r13`
Optional boolean. If set to true and supported by the UE, a 16 bits SN and SO will be used.

`pollPDU_v1310`
Optional enumeration: 512, 1024, 2048, 4096, 6144, 8192, 12288, 16384. pollPDU-v1310 value.

`pollByte_r14`
: Optional enumeration: 1, 2, 5, 8, 10, 15, 3500, 4000, 4500, 5000, 5500, 6000, 6500, 7000, 7500, 8000, 9000, 10000, 11000, 12000, 13000, 14000, 15000, 16000, 17000, 18000, 19000, 20000, 25000, 30000, 35000, 40000. pollByte-r14 value in kBytes. Sent if supported by the UE.

`dl_am`
: Object. Downlink RLC AM configuration.

`t_Reordering`
: Enumeration: 0, 5, 10, 15, 20, 25, 30, 35, 40, 45, 50, 55, 60, 65, 70, 75, 80, 85, 90, 95, 100, 110, 120, 130, 140, 150, 160, 170, 180, 190, 200, 1600. t_Reordering timer value in ms.

`t_StatusProhibit`
: Enumeration: 0, 5, 10, 15, 20, 25, 30, 35, 40, 45, 50, 55, 60, 65, 70, 75, 80, 85, 90, 95, 100, 105, 110, 115, 120, 125, 130, 135, 140, 145, 150, 155, 160, 165, 170, 175, 180, 185, 190, 195, 200, 205, 210, 215, 220, 225, 230, 235, 240, 245, 250, 300, 350, 400, 450, 500, 800, 1000, 1200, 1600, 2000, 2400. t_StatusProhibit timer value in ms.

`dl_extended_RLC_LI_Field_r12`
: Optional boolean. If set to true and supported by the UE, a 15 bits LI will be used.

`dl_extended_RLC_AM_SN_r13`
: Optional boolean. If set to true and supported by the UE, a 16 bits SN and SO will be used.

`pdcp_config`
: Object. Gives the PDCP configuration.

`discardTimer`
: Integer. PDCP discardTimer variable (in ms). 0 means infinity.

`pdcp_SN_Size`
: (UM only) Enumeration: 7, 12. pdcp sequence number size in bits.

`pdcp_SN_Size_v1130`
: (AM only) Optional boolean. If set to true and supported by the UE, a 15 bits SN will be used.

`pdcp_SN_Size_v1310`
: (AM only) Optional boolean. If set to true and supported by the UE, a 18 bits SN will be used.

`statusReportRequired`
: (AM only) Boolean. PDCP statusReportRequired variable.

`headerCompression`
: Optional object. If not present or `null`, header compression is disabled.

`maxCID` Range: 1 to 16383.

`profile0x0001`
: Boolean. If true, enable RTP v1 ROHC profile.

`profile0x0002`
: Boolean. If true, enable UDP v1 ROHC profile.

`profile0x0004`
Boolean. If true, enable IP v1 ROHC profile.

`logical_channel_config`
Object. MAC Logical channel configuration. The downlink values are assumed to be the same as the uplink values.

`priority` Range: 1 to 16. logical channel priority. Lower value has more priority.

`prioritisedBitRate`
Enumeration: 0, 8, 16, 32, 64, 128, 256, -1, 512, 1024, 2048. Prioritised bit rate. -1 means infinity.

`bucketSizeDuration`
Enumeration: 50, 100, 150, 300, 500, 1000. Bucket size duration in ms.

`logicalChannelGroup`
Range: 0 to 3. Logical channel group to which this logical channel belongs.

`logicalChannelSR_Mask`
Optional boolean. Indicates whether this DRB should use SR masking or not.

`logicalChannelSR_Prohibit`
Optional boolean. Indicates whether this DRB will use the R12 logical channel SR prohibit timer or not. Note that it requires setting `logical_channel_sr_prohibit_timer` parameter in `mac_config` object.

8.7 MBMS configuration

The object `mbms` gives the eNB MBMS configuration. Other parameters previously present in this object are now configured in MBMS Gateway. Here are the properties of the object:

`sib13_periodicity`
Integer. Range: 8 to 512, power of two. Periodicity (in 10 ms frames) of the SIB13 retransmission. The SIB13 contain the parameters to find the MCCH for each MBSFN area.

`synchronization_area_id`
Integer. Range: 0 to 65535. MBSFN synchronization area identifier sent in the M2 Setup Request message.

`use_precise_timestamp`
Optional boolean (default=false). If set to true, eNB and MBMSGW internal time (as seen by the `time` monitor command) are assumed to be synchronized. SYNC packets will be dropped in their time stamp is not ahead of 1 to `msp_fifo_size` number of times the MCH Scheduling Period compared to the current eNB internal time. If set to false, the eNB will not consider the SYNC packet timestamp.

`mbms_time_offset`
Optional integer. Offset in ms applied to the eNB internal time (as retrieved by the `time` monitor command) so as to synchronize the eNB with the MBMSGW and ensure that they have a common time base for SYNC protocol. It is ignored if `use_precise_timestamp` option is set to false.

`msp_fifo_size`
Optional integer. Range: 2 to 512, default set to 8. Size of the FIFO used by eNB to store SYNC packets based on their timestamp, ahead of the current MCH

Scheduling Period (one entry per MCH Scheduling Period). Should be set to a relevant value according to the time_offset parameter set in MBMS Gateway.

`service_area_id_list`
Array of integers. Range: 0 to 65535 per item. List of MBMS service areas for which the cell is subscribed. This list is sent in the M2 Setup Request message.

`notification_config`
Object. Definition of the MCCH change notification parameters. Note: the MCCH parameters are currently static so that eNodeB never signals MCCH change.

`notification_repetition_coeff`
Enumeration: 2, 4.

`notification_offset`
Range: 0 to 10.

`notification_sf_index`
Range: 1 to 6.

8.8 NB-IoT cell configuration

8.8.1 Basic NB-IoT cell parameters

`plmn_list`
Array of objects or strings. List of PLMNs broadcasted by the eNodeB. At most 6 PLMNs are supported. Each element of the array is either a PLMN (5 or 6 digit string) or an object containing the following properties:

`plmn` String. PLMN (5 or 6 digits).

`reserved` Boolean. True if the cell is reserved for operator use.

`attach_without_pdn`
Optional boolean (default = false). Indicates if PLMN supports attach without PDN connectivity.

When `reserved` is not provided, its default value is `false`.

`operation_mode`
Enumeration: same_pci, diff_pci, guardband, standalone. Set the cell operation mode. `same_pci` and `diff_pci` are for in-band operation. `diff_pci` must be used in case of a LTE base cell with 4 PBCH antenna ports.

For in-band operation, the eNodeB checks that the specified DL and UL EARFCN are consistent.

`dl_prb` Optional Integer. If provided, set the DL PRB number in the base LTE cell for in-band operation. If not present, dl_earfcn must be present.

`ul_prb` Optional Integer. If provided, set the UL PRB number in the base LTE cell for in-band operation. If not present, ul_earfcn must be present.

`dl_earfcn`
Range: 0 to 262143. Set the DL EARFCN. For in-band operation `dl_prb` can be used instead. For in-band operation, the DL EARFCN must be such as it is at less than 7.5 kHz of the center of a resource block of the base LTE cell. Such DL EARFCN are spaced by 900 kHz (5 resource blocks).

`ul_earfcn`
: Optional. Range: 0 to 262143. Set the UL EARFCN. If not provided, the default DL/UL gap is used. For in-band operation `ul_prb` can be used instead. For in-band operation, the UL EARFCN should be such as the corresponding resource block in the base LTE cell is not used by PRACH or PUCCH. It is more efficient to set it at the edge of the PUSCH spectrum to have larger contiguous PUSCH allocations.

`ul_carrier_freq_offset`
: Optional. Range -10 to 9 (default = 0). Offset in 5 kHz units of the center of the UL bandwidth from the UL EARFCN. It is useful to align the UL bandwidth with a LTE uplink resource block in case of in-band operation.

`n_antenna_dl`
: Enumeration: 1, 2, 4 or 8. Number of DL antennas. It must be the same for all NB-IoT and LTE cells sharing the same RF port.

`n_antenna_ul`
: Enumeration: 1, 2, 4 or 8. Number of UL antennas. It must be the same for all NB-IoT and LTE cells sharing the same RF port.

`n_antenna_pbch`
: Optional enumation: 1, 2. Number of NPBCH antenna ports. It is automatically set to `min`(2, number of PBCH antenna ports of the base cell) for in-band operation. Otherwise its default value is `min(2, n_antenna_dl)`.

`multi_band_list`
: Optional array of integers. List the additional bands supported by the cell, in decreasing priority order (MFBI feature). The downlink and uplink frequency of the cell must exist in all these bands.

`cell_id` Range: 0 to 1023. 7 bit (long macro eNB), 8 bit (macro eNB) or 10 bit (short macro eNB) cell identifier. For a macro eNodeB, the 28 bit cell identifier is the concatenation of `enb_id` and `cell_id`.

`tac` Range: 0 to 65535. Tracking Area Code of the cell. Note: the NB-IoT and LTE tracking areas must be different.

`base_cell_id`
: Integer. Only needed for in-band operation. 7, 8 or 10 bit cell identifier of the base cell in which the NB-IoT cell is mapped.

`cell_gain`
: Optional float (default = 0). Downlink cell gain in dB. Must be between -200 and 0 (included).

`nrs_crs_power_offset`
: Float. Range: -6 to 9. Power offset in dB of the Narrow band Reference Signal with respect to the LTE Cell Reference Signal. This field is only necessary when the operation mode is `same_pci`.

`n_id_ncell`
: Range: 0 to 503. Physical cell identifier. It is not necessary if the operation mode is `same_pci`.

`cipher_algo_pref`
: Array of integers. Set the preferred algorithms for RRC and User Plane encryption in decreasing order of preference (see corresponding LTE cell parameter).

`integ_algo_pref`
: Array of integers. Set the preferred algorithms for RRC integrity check in decreasing order of preference (see corresponding LTE cell parameter).

`inactivity_timer`
: Integer. Send RRC connection release after this time (in ms) of network inactivity.

`rel13_5` Optional boolean (default = true). If true, enable incompatible physical layer changes for NPBCH/BCCH introduced in release 13.5.

8.8.2 System Information parameters

`si_value_tag`
: Range: 0 to 31. Increment modulo 32 if SI is modified.

`r_sib1` Enumeration: 4, 8, 16. Number of SIB1 repetitions for 256 radio frames.

`tbs_sib1` Optional enumeration: 26, 41, 55, 85. SIB1 size in bytes. If not set, the size is automatically computed by the eNB based on the SIBs defined in the configuration file. It can be useful to set it manually in case new SIBs are scheduled during runtime (like SIB14).

`cell_barred`
: Boolean. Value of SIB1.cellBarred-r13

`intra_freq_reselection`
: Boolean. Value of SIB1.intraFreqReselection-r13

`q_rx_lev_min`
: Integer. Value of SIB1.q-RxLevMin.

`delta_rx_lev_min`
: Optional integer (-8 to 0). Value of SIB1.nonCriticalExtension.cellSelectionInfo-v1350.delta-RxLevMin-v1350. If set to 0, the field is not transmitted.

`q_qual_min`
: Integer. Value of SIB1.q-QualMin

`p_max_enable`
: Boolean. If true SIB1.p-Max is present.

`p_max` Integer. Value of SIB1.p-Max.

`dl_bitmap`
: Optional bit string. Set the Downlink Subframe bitmap. It must contain 10 or 40 bits.

`si_window_length`
: Integer. SI window length in ms.

`si_radio_frame_offset`
: Integer. SI radio frame offset (in radio frames).

`si_value_tag_list_enable`
: Boolean. If true, enables per SIB si_value_tag.

`sib_sched_list`
: Array of object. Each object contains the content of one SI scheduling slot (the first slot must contain the SIB2):

  `si_periodicity`
  : Integer. SI periodicity in Radio Frames.

`si_repetition_pattern`
: Integer. The SI is present every `si_repetition_pattern` radio frames.

`si_value_tag`
: Optional integer. Range: 0 to 3. Must be present if `si_value_tag_list_enable` is true. Increment modulo 4 if the corresponding SIB is modified.

`filename`
: Filename containing the SIBs. The content is in textual ASN.1 (GSER syntax).

`sib16`
: Optional object. If present, the SIB16 message will be scheduled. It must contain the `si_periodicity`, `si_repetition_pattern` and `si_value_tag` objects described in `sib_sched_list`. See [sib_sched_list], page 61.

8.8.3 MAC configuration

`mac_config`
: Object. MAC configuration. Currently the same for all UEs. The following properties are defined:

`msg3_max_harq_tx`
: Integer. Maximum number of HARQ transmissions for MSG3.

`ul_max_harq_tx`
: Integer. Maximum number of HARQ transmissions for uplink.

`dl_max_harq_tx`
: Integer. Maximum number of HARQ transmissions for downlink.

`ul_max_consecutive_retx`
: Integer. Maximum number of UL retransmissions after which the UE is disconnected.

`dl_max_consecutive_retx`
: Integer. Maximum number of DL retransmissions after which the UE is disconnected.

`time_alignment_timer_dedicated`
: Integer. Time alignment timer dedicated in ms. 0 means infinity.

`periodic_bsr_timer`
: Integer. Periodic BSR timer value in NPDCCH periods.

`retx_bsr_timer`
: Integer. Retransmission BSR timer value in NPDCCH periods.

`logical_channel_sr_prohibit_timer`
: Integer. Logical Channel SR prohibit timer value in NPDCCH periods. 0 means that the timer is released.

`data_inactivity_timer`
: Integer. Value in seconds of the data inactivity monitoring timer. 0 means that the timer is deactivated.

`time_alignment_tx_timer`
: Optional integer from 0 to 10240 (default = 0). Transmit the UL time alignment information every `time_alignment_tx_timer` ms. The value 0 means infinity. No actual UL time alignment measurement is done and a zero time alignment MAC control element is always sent. Hence this option is only useful for UE testing.

8.8.4 PHY and L1 configuration

`npusch_max_its`
: Integer. Set the maximum number of turbo decoder iterations

`coverage_levels`
: Array of objects. Configuration of each coverage level. There must be the same number of coverage levels as NPRACH configurations in the SIB2.

  NPRACH Parameters:

  `nprach_detect_threshold`
  : Optional float. Set the NPRACH SNR detection threshold in dB.

  RAR Parameters:

  `npdcch_ra_n_rep`
  : Integer. Number of RAR (Random Access Response) NPDCCH repetitions. It must be <= SIB2.npdcch-NumRepetitions-RA-r13.

  `npdsch_ra_n_rep`
  : Integer. Number of repetitions for RAR NPDSCH..

  `npdsch_ra_i_tbs`
  : Integer. Range 0 to 12. I_TBS for the RAR NPDSCH. For in-band cells, the maximum value is 10.

  `npdsch_ra_i_delay_min`
  : Optional Integer (default = 0). Range: 0 to 7. Minimum value for the RAR DCI N1 scheduling delay field.

  `ul_sc_spacing`
  : Enumeration: 0, 1. Select the subcarrier spacing used by the UE. 0 = 3.75 KHz subcarriers, 1 = 15 KHz subcarriers.

  MSG3 parameters:

  `msg3_n_sc`
  : Enumeration: 1, 3, 6, 12. Maximum number of subcarriers for MSG3. The eNodeB uses more than one subcarrier only if the UE supports it.

  `msg3_single_tone_mcs: 2`
  : Integer. Range: 0 to 2. MCS for single-tone MSG3.

  `msg3_multi_tone_mcs`
  : Integer. Range 0 to 2. MCS for multi-one MSG3. Only needed if `msg3_n_sc > 1`.

  `msg3_n_rep`
  : Integer. Range 1 to 128. Number of repetitions for MSG3.

  `msg3_i_delay_min`
  : Optional Integer (default = 0). Range: 0 to 3. Minimum value for the RAR UL grant scheduling delay field.

  Paging parameters:

  `npdcch_paging_n_rep`
  : Integer. Range: 1 to 2048. Number of repetitions for the paging NPDCCH. It must be <= SIB2.npdcch-NumRepetitionPaging-r13.

  `npdsch_paging_i_tbs`
  : Integer. Range: 0 to 12. I_TBS for the paging NPDSCH. For in-band cells, the maximum value is 10.

`npdsch_paging_n_rep`
: Integer. Range: 1 to 2048. Number of repetitions for the paging NPDSCH.

UE dedicated parameters:

`npdcch_uss_n_rep_max`
: Integer. Range: 1 to 2048. Max number of NPDCCH repetitions for the User Search Space (USS).

`npdcch_uss_n_rep`
: Integer. Actual number of repetitions for the USS NPDCCH. The special value 0 means to use a single CCE (instead of 2) with a single transmission.

`npdcch_uss_start_sf`
: Enumeration: 1.5, 2, 4, 8, 16, 32, 48, 64. Used to compute of the period of the USS NPDCCH by multiplying it to `npdcch_uss_n_rep_max`.

`npdcch_uss_offset`
: Integer. Range: 0 to 3. USS NPDCCH start offset in 8th of the USS NPDCCH period.

`npdsch_i_tbs`
: Integer. Range: 0 to 13. I_TBS for NPDSCH. For in-band cells, the maximum value is 10. For category NB1 UEs, the value is limited to 12.

`npdsch_i_sf`
: Optional Integer (default = -1). Range: -1 to 7. I_SF value for NPDSCH. -1 means that the eNodeB scheduler automatically chooses it.

`npdsch_n_rep`
: Integer. Range: 1 to 2048. Number of NPDSCH repetitions.

`npdsch_i_delay_min`
: Optional Integer (default = 0). Range: 0 to 7. Minimum value for the DCI N1 scheduling delay field.

`npusch_n_sc`
: Enumeration: 1, 3, 6, 12. Maximum number of subcarriers for NPUSCH. The eNodeB uses more than one subcarrier only if the UE supports it.

`npusch_single_tone_i_tbs`
: Integer. Range: 0 to 10. I_TBS for single-tone NPUSCH.

`npusch_multi_tone_i_tbs`
: Integer. Range: 0 to 13. I_TBS for multi-tone NPUSCH. For category NB1 UEs, the value is limited to 12.

`npusch_i_ru`
: Optional Integer (default = -1). Range: -1 to 7. I_RU value for NPUSCH. -1 means that the eNodeB scheduler automatically chooses it.

`npusch_n_rep`
: Integer. Range: 1 to 128. Number of NPUSCH repetitions.

`npusch_i_delay_min`
: Optional Integer (default = 0). Range: 0 to 3. Minimum value for the DCI N0 scheduling delay field.

`npusch_all_symbols`
: Optional boolean. If true, NPUSCH symbols are transmitted in the SRS symbols. The field must be present if SRS is enabled on the base cell for in-band operation.

`group_hopping_disabled`
: Optional boolean (default = false). If true, disable group hopping in the UE RRC dedicated signaling.

`dedicated_ack_nack_num_rep_enabled`
: Optional boolean (default = false). If true, set the `ack-NACK-NumRepetitions-r13 parameter` in the RRC connection setup message. It is set to the same value as `ack-NACK-NumRepetitions-Msg4-r13` from the SIB2. Note: this parameter is only useful for UE testing.

8.8.5 Advanced parameters

`rx_epre_in_dbfs`
: Optional boolean (default = false). In the logs, the EPRE (Energy Per Resource Element) is displayed in dBm if the RF interface provides its reference receive power and if rx_epre_in_dbfs = false. Otherwise it is displayed in dBFS (Decibels relative to Full Scale).

`manual_ref_signal_power`
: Optional boolean (default = false). If the RF interface provides its transmit power, then `SIB2.nrs-Power-r13` is automatically set. If manual_ref_signal_power is true, then `SIB2.nrs-Power-r13` is never automatically set by the eNodeB.

`rrc_cnx_reject_extWaitTime`
: Optional integer. Range: 1 to 1800 (default = 10). Set the wait time in seconds in the RRC connection reject message.

`rrc_cnx_release_extWaitTime`
: Optional integer. Range: 0 to 1800 (default = 0). RRC connection release extended wait time in seconds.

`rrc_cnx_release_extWaitTime_CPdata`
: Optional integer. Range: 0 to 1800 (default = 0). RRC connection release extended wait time for Control Plane CIoT EPS optimisation in seconds.

`power_npss`
: Option float (default = 0). Set the NPSS power level (in dB) relative to the NRS power level.

`power_nsss`
: Option float (default = 0). Set the NSSS power level (in dB) relative to the NRS power level.

`force_full_bsr`
: Optional boolean (default = false). If true, the eNodeB considers the UE always indicates a full buffer size. Hence the UE is scheduled as often as possible for NPUSCH transmission.

`force_dl_schedule`
: Optional boolean (default = false). If true, the eNodeB considers there is always DL data waiting for transmission. Hence the UE is scheduled as often as possible for NPDSCH transmission.

`rrc_procedure_filter`
: Optional object. Allows to define the eNB behavior for a list of RRC procedures. Each property name represents a RRC procedure. The ones currently supported are `rrc_connection_request` and `rrc_connection_reestablishment_request`. Each property value is an enum: `treat` (UE message is processed), `ignore` (UE message is ignored) or `reject` (UE message is rejected).
By default all procedures are treated.

Example:

```
rrc_procedure_filter: {
    rrc_connection_request: "treat",
    rrc_connection_reestablishment_request: "reject"
}
```

`rach_ignore_count`
: Optional integer. Indicates how many consecutive RACH attempts are ignored by the eNB.

`srb_config`
: Optional object. Allows to override some parameters of the default configuration specified in 3GPP 36.331 chapter 9.2.1. If unset, the eNB will configure maxRetxThreshold value to 32 and t-PollRetransmit value to 25 s. The object contains the following fields:

`maxRetxThreshold`
: Enumeration: 1, 2, 3, 4, 6, 8, 16, 32 (default 32). maxRetxThreshold value on UE side.

`enb_maxRetxThreshold`
: Enumeration: 1, 2, 3, 4, 6, 8, 16, 32 (default 32). maxRetxThreshold value on eNB side.

`t_PollRetransmit`
: Enumeration: 250, 500, 1000, 2000, 3000, 4000, 6000, 10000, 15000, 25000, 40000, 60000, 90000, 120000, 180000 (default 25000). t-PollRetransmit timer value in ms on UE side.

`enb_t_PollRetransmit`
: Enumeration: 250, 500, 1000, 2000, 3000, 4000, 6000, 10000, 15000, 25000, 40000, 60000, 90000, 120000, 180000 (default 25000). t-PollRetransmit timer value in ms on eNB side.

`drb_config`
: String. Filename for the DRB configuration. See the file `drb_nb.cfg` to have a description of its fields. Note that the DRB configuration is ignored when Control Plane CIoT optimization is used.

`ue_count_max`
: Optional integer (default = 500). Maximum number of UEs (for this cell).

`erab_count_max`
: Optional integer (default = 1500). Maximum number of ERABs (for this cell).

`rar_backoff`
: Optional. Range: -1 to 15. If set to -1, no Backoff Indicator is sent in the Random Access Response message. Values 0 to 15 refer to the index of table 7.2-2 found in 3GPP 36.321.

`npdcch_uss_half_rb_cce`
: Optional Integer (default = 0). Range: 0 to 1. Set the first CCE index used for half RB NPDCCH allocation.

`test_mode`
: Optional object. Enable specific test modes where UE contexts are automatically created when starting the eNodeB. The `type` property selects the test mode:

  `npusch`
  : Enables continuous reception of NPUSCH by the eNodeB. DCI N0 is transmitted. The following additional properties are available:

    `rnti` Integer. Range 0 to 65535. Select the NPUSCH RNTI.

    `npusch_retx`
    : Boolean. If false, don't force the UE to retransmit in case of error.

  `npdsch`
  : Enables continuous transmission of NPDSCH. The NPDSCH payload contains valid data with PDCP packets of constant length. DCI are transmitted. NPUSCH ACK/NACK are received. The following additional properties are available:

    `rnti` Integer. Range 0 to 65535. Select the PDSCH RNTI.

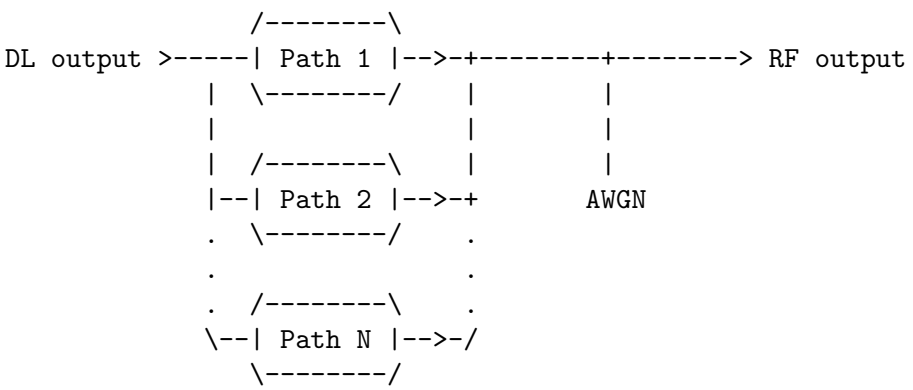
    `npdsch_retx`
    : Boolean. If false, don't retransmit the unacknowledged NPDSCH (hence NPUSCH ACK/NACK are ignored).

`preemptive_ul_grant`
: Optional boolean (default = true). When set to true, the eNB can send a DCIN0 grant before the UE explicitly request an allocation via the random access procedure.

8.9 Channel simulator

The channel simulator applies after the DL modulator(s) on each RF port using the configured sample rate. It takes `cell.n_antenna_dl` channels as inputs (*n_input*) and may output a different number of channels (*n_output*) (see the global `n_antenna_dl` parameter).

It applies a number of delay paths with a configurable gain and phase for each *n_input* x *n_output* antenna combination. Each path can also appply a Rayleigh fading (Jakes or Gaussian fading model). In case of MIMO channels, a configurable MIMO correlation matrix is applied for each path. Then a white Gaussian noise is added (AWGN).



Preconfigured path configurations are available for common 3GPP channel models. Preconfigured 3GPP MIMO correlation matrixes are also available.

When the channel simulator is enabled, the default digital gain (`tx_gain_offset`) is lowered from -11 dB to -20 dB to allow a larger dynamic range. Use the `t spl` monitor command to check that no overflow is present at the RF output.

The CPU usage of the channel simulator increases with the sample rate, number of MIMO channels and the number of paths. If the CPU load is too high (see the `t cpu` monitor command to estimate it), you can reduce the RF bandwidth (i.e. `n_rb_dl`), reduce the number of MIMO channels or use a simpler channel model with a smaller number of paths.

The `channel_dl` object contains the downlink channel simulator parameters:

`type` Optional. Must be present if no `paths` property. Set the predefined channel type:

- `"awgn"` Additive White Gaussian Noise channel. It is equivalent to specifying a single zero delay unit gain constant path:

```
paths: [{
    type: "constant",
    gain: 1.0,
    delay: 0,
    channel_matrix: [[ 1 ]],
}]
```

  When there are more than one input or output antennas, the channel matrix is set to a truncated identity matrix. In this case, it is usually better to manually specify a `paths` configuration to select an explicit channel matrix.

- `"epa"` Extended Pedestrian A model from TS 36.101.
- `"eva"` Extended Vehicular A model from TS 36.101.
- `"etu"` Extended Typical Urban model from TS 36.101.
- `"mbsfn"` MBSFN channel from TS 36.101.

`freq_doppler`
For `"epa"`, `"eva"`, `"etu"` or `"mbsfn"` channels, sets the doppler frequency, in Hz.

`mimo_correlation`
Optional enumeration or matrix. The default value is `low`.

MIMO correlation matrix. Allowed values:

- `low` Low correlation matrix from TS 36.101 (identity matrix).
- `medium` Medium correlation matrix from TS 36.101
- `high` High correlation matrix from TS 36.101

Alternatively, an explicit complex matrix of `n` rows and `n` columns can be provided where `n` is the product of the number *n_input* and *n_output* antennas. The matrix must be Hermitian positive.

`paths` Optional array of objects. Set user defined paths. If present the `type` parameter is ignored. Each path has the following fields:

`type`

Enumeration. Type of path. `constant` for constant path, `rayleigh` for Rayleigh fading with the Jakes model, `rayleigh_gauss` for Rayleigh fading with the Gaussian model.

`gain` Relative path gain, in dB.

`delay` Path delay, in ns. Note: the delay is internally rounded to an integer number of samples.

`channel_matrix`
Only necessary for `"constant"` path. Complex matrix of *n_output* rows and *n_input* columns giving the channel coefficients.

`freq_doppler`
Only needed for Rayleigh paths. Doppler frequency in Hz.

`mimo_correlation`
Only needed for Rayleigh paths. Path specific correlation matrix (same definition as the global `channel.mimo_correlation` property). If not present, the global `channel.mimo_correlation` matrix is used for this path.

`snr` Float. Set the initial Signal to Noise Ratio in dB. The Gaussian noise is generated with a constant power density over the whole generated bandwidth. Over a 15 kHz bandwidth, the signal power is defined as the Cell Reference Signal power. So a SNR of 0 dB gives a noise level having the same EPRE as the Cell Reference Signal.

The snr can be interactively modified with the `snr` monitor command.

Warning: the reference signal level is not modified when the `cell_gain` monitor command is used. So you can monitor the noise level on a spectrum analyzer by suppressing the DL signal with a near zero cell gain (e.g. `cell_gain 1 -200`).

`dump_paths`
Optional boolean (default = false). Print on the standard output the path delays and relative powers.

It only applies when the `paths` property is not set.

`max_paths`
Optional integer. Set the maximum number of paths. The paths with the smallest power are removed. It can be used to reduce the CPU load at the expense of the precision of the simulated impulse response. The default value depends on the sample rate.

It only applies when the `paths` property is not set.

Note: the channel simulator only supports a sample rate which is an even multiple of 1.92 MHz (more precisely, `sample_rate_num` must be even). So the sample rate should be manually set with the `sample_rate` option for the following bandwidths:

Bandwidth (MHz)	**Sample rate (MHz)**
1.4	3.84
5	7.68
15	23.04

9 Remote API

You can access LTEENB via a remote API.
Protocol used is WebSocket as defined in RFC 6455 (`https://tools.ietf.org/html/rfc6455`).

9.1 Messages

Messages exchanged between client and LTEENB server are in strict JSON format.

Each message is represented by an object. Multiple message can be sent to server using an array of message objects.

Time and delay values are floating number in seconds.

All messages have at least following definition:

`message` String. Represent type of message. This parameter is mandatory and depending on its value, other parameters will apply.
If message is a response from server, response message will have same message member.

`message_id`
Optional any type. If set response sent by the server to this message will have same message_id. This is used to identify response as WebSocket does not provide such a concept.

`start_time`
Optional double. Represent the delay before executing the message.
If not set, the message is executed when received.
Note that some command (*log_get*, *log_reset*, *config_get*, *config_set*, *stats*) can't be executed in future.

`absolute_time`
Optional boolean (default = false). If set, `start_time` is interpreted as absolute.
You can get current clock of system using `time` member of `config_get` command.

9.2 Errors

If a message produces an error, response will have an error string field representing the error.

9.3 Sample nodejs program

You will find in this documentation a sample program: `ws.js`.
This is a nodejs program that allow to send message to `PROG`.
It requires nodejs to be installed:

```
yum install nodejs npm
npm install nodejs-websocket
```

Then simply start it with server name and message you want to send:

```
./ws.js 127.0.0.1:9000 '{"message": "config_get"}'
```

9.4 Common messages

`config_get`
: Retrieve current config.

  Response definition:

  `type` Always `"ENB"`

  `name` String representing server name.

  `time` Number representing time in seconds.
  Usefull to send command with absolute time.

  `logs` Object representing log configuration.
  With following elements:

  `layers` Object. Each member of the object represent a log layer configuration:

  `layer name`
  : Object. The member name represent log layer name and parameters are:

    `level` See [log_options], page 25,

    `max_size` See [log_options], page 25,

  `count` Number. Number of bufferizer logs.

`global_enb_id`
: Object containing the following members:

  `plmn` String. PLMN identity part of the global eNB ID.

  `enb_id_type`
  : String. eNB type (short_macro, macro, long_macro, home).

  `enb_id` Integer. eNB identity part of the global eNB ID.

`cells` Object. Each member name/value represents the LTE cell ID/cell definition:

`n_id_cell`
: Integer. Physical cell ID.

`n_rb_dl` Integer. Number of downlink resource blocks.

`n_rb_ul` Integer. Number of uplink resource blocks.

`dl_earfcn`
: Integer. Downlink EARFCN.

`ul_earfcn`
: Integer. Uplink EARFCN.

`gain` Float. Cell gain in dB.

`mode` Enumeration: FDD, TDD. Operation mode.

`uldl_config`
: Optional integer. TDD subframe assignment. Only present if `mode` is `"TDD"`.

`sp_config`
: Optional integer. TDD special subframe pattern. Only present if `mode` is "TDD".

`dl_cyclic_prefix`
: Enumeration: normal, extended. Downlink cyclic prefix.

`ul_cyclic_prefix`
: Enumeration: normal, extended. Uplink cyclic prefix.

`prach_config_index`
: Integer. PRACH configuration index.

`prach_freq_offset`
: Integer. PRACH frequency offset.

`delta_pucch_shift`
: Integer. deltaPUCCH-Shift.

`nrb_cqi` Integer. nRB-CQI.

`n_cs_an` Integer. nCS-AN.

`pucch_allocation`
: Array of objects. Each object contains:

  `type` Enumeration: 2/2a/2b, 3.

  `rbs` Integer. Number of resource blocks for this type.

  `n` Integer. Number of PUCCH for this type.

`pucch_ack_nack_start`
: Integer. n1PUCCH-AN.

`pucch_reserved_rbs`
: Array of 10 integers. Each entry gives the number of resource blocks reserved for PUCCH in the corresponding subframe.

`sr_resource_count`
: Integer. Number of Scheduling Request resources.

`cqi_resource_count`
: Integer. Number of Channel Quality Indicator resources.

`br_sr_resource_count`
: Optional integer. Number of Bandwidth Reduced Scheduling Request resources.

`br_cqi_resource_count`
: Optional integer. Number of Bandwidth Reduced Channel Quality Indicator resources.

`srs_resources`
: Array containing the SRS related information:

  `offsets` Integer. Number of possible offsets.

  `freqs` Integer. Number of possible frequencies.

  `total` Integer. Total number of resources.

`gbr` Object containing the GBR related information:

`dl_limit` Integer. Downlink limit in number of resource elements per second.

`ul_limit` Integer. Uplink limit in number of resource elements per second.

`nb_cells` Object. Each member name/value represents the NB-IoT cell ID/cell definition:

`n_id_ncell`
Integer. Physical cell ID.

`dl_earfcn`
Integer. Downlink EARFCN.

`ul_earfcn`
Integer. Uplink EARFCN.

`gain` Float. Cell gain in dB.

`operation_mode`
Enumeration: same_pci, diff_pci, guardband, standalone.

`config_set`
Change current config.
Each member is optional.
Message definition:

`logs` Object. Represent logs configuration. Same structure as config_get (See [config_get logs member], page 71).
All elements are optional.

`cells` Optional object used to configure cells individually. Each cell configured must be a new object inside `cells` object, named with the cell_id value and containing the following fields:

`pusch_mcs`
Integer or array of 10 integers (range: -1 to 28). Force the PUSCH MCS (test feature). If an array is provided, it provides the PUSCH MCS for each subframe. Use -1 not to force the MCS in a given subframe.

`pusch_fixed_rb_alloc`
Optional boolean or array of 10 booleans. Force fixed PUSCH RB allocation in all or a selected set of subframes. If an array is provided, a value true at the index value i of the array indicates that a fixed PUSCH RB allocation is used in subframe number i.

The parameters `pusch_fixed_rb_start` and `pusch_fixed_l_crb` are used for the fixed allocation. `pusch_fixed_l_crb` must be of the form `2^n1*3^n2*5^n3`. PUSCH are allocated only if they don't overlap with PUCCH or PRACH, so care must be taken when defining the range. In some cases, PUSCH retranmissions may use other RBs.

`pusch_fixed_rb_start`
: Optional integer or array of 10 integers. First RB for fixed PUSCH allocation. If an array is provided, it gives the first RB for each subframe (see `pusch_fixed_rb_alloc`).

  For a cell configured for category M1 UEs, `pusch_fixed_rb_start` and `pusch_fixed_l_crb` give the allocation inside a narrow band (hence `pusch_fixed_rb_start + pusch_fixed_l_crb` <= 6).

`pusch_fixed_l_crb`
: Optional integer or array of 10 integers. Number of consecutive RBs for fixed PUSCH allocation. If an array is provided, it gives the number of consecutive RBs for each subframe (see `pusch_fixed_rb_alloc`).

`pusch_fixed_rb_forced`
: Optional boolean (default = false). If true, the eNodeB schedules the PUSCH with fixed RB allocation even if it collides with PUCCH/PRACH or another PUSCH.

`pusch_multi_cluster`
: Optional boolean (default = false). If true, enable multi-cluster PUSCH resource allocation for the UEs supporting it (release 10). Note: this is a UE test feature, so the multi cluster allocation is not optimized by the scheduler.

`pusch_max_mcs`
: Optional. Range: 0 to 28 (default = 28). CPU load limitation: maximum MCS allocated by the eNodeB for PUSCH. Smaller MCS give a smaller bitrate and a smaller CPU load.

`pusch_max_its`
: Optional. Range 1 to 20 (default = 10). CPU load limitation: set the maximum number of iterations of the turbo decoder. A higher value gives a lower frame error rate but a higher CPU load.

`force_full_bsr`
: Optional boolean (default = false). If true, the eNodeB considers the UE always indicates a full buffer size. Hence the UE is scheduled as often as possible for PUSCH transmission.

`force_dl_schedule`
: Optional boolean (default = false). If true, the eNodeB considers there is always DL data waiting for transmission. Hence the UE is scheduled as often as possible for PDSCH transmission.

`pdsch_mcs`
: Integer or array of 10 integers (range: -1 to 28). Force the PDSCH MCS (test feature). If an array is set, it provides the PDSCH MCS for each subframe. Use -1 not to force the MCS in a given subframe.

`pdsch_mcs_from_cqi`
: Integer or array of 16 integers (range: -1 to 28). Force the PDSCH MCS (test feature).

If an array is set, it provides the PDSCH MCS according to the CQI reported by UE. Use -1 not to force the MCS for a given CQI.

`pdsch_fixed_rb_alloc`
: Optional boolean or array of 10 booleans. Force fixed PDSCH RB allocation using the parameters `pdsch_fixed_rb_start` and `pdsch_fixed_l_crb`. If an array is provided, it selects the fixed PDSCH allocation for each subframe.

  For a cell configured for category M1 UEs, fixed PDSCH RB allocation is only possible in subframes where the PDSCH MCS is fixed (see `pdsch_mcs`).

`pdsch_fixed_rb_start`
: Optional integer or array of 10 integers. First RB for fixed PDSCH allocation (see `pdsch_fixed_rb_alloc`). If an array is provided, it provides the first RB for each subframe.

  For a cell configured for category M1 UEs, `pdsch_fixed_rb_start` and `pdsch_fixed_l_crb` give the allocation inside a narrow band (hence `pdsch_fixed_rb_start + pdsch_fixed_l_crb` <= 6).

`pdsch_fixed_l_crb`
: Optional integer or array of 10 integers. Number of consecutive RBs for fixed PDSCH allocation (see `pdsch_fixed_rb_alloc`). If an array is provided, it provides the consecutive RBs for each subframe.

`rrc_procedure_filter`
: Optional object. Allows to define the eNB behavior for a list of RRC procedures.
  Each property name represents a RRC procedure. The ones currently supported are `rrc_connection_request` and `rrc_connection_reestablishment_request`.
  Each property value is an enum: `treat` (UE message is processed), `ignore` (UE message is ignored) or `reject` (UE message is rejected).
  By default all procedures are treated.

  Example:

```
rrc_procedure_filter: {
    rrc_connection_request: "treat",
    rrc_connection_reestablishment_request: "reject"
}
```

`rach_ignore_count`
: Optional integer. Indicates how many consecutive RACH attempts are ignored by the eNB.

`cell_barred`
: Optional boolean. Indicates if the cell should be barred or not. SIB1 content is updated accordingly.

`rrc_cnx_reject_waitTime`
: Optional integer (range: 1 to 16). RRC connection reject wait time in seconds. Only applicable to LTE cells.

`rrc_cnx_reject_extWaitTime`
: Optional integer (range: 0 to 1800 for LTE cells, 1 to 1800 for NB-IoT cells). RRC connection reject extended wait time in seconds.

`rrc_cnx_reject_deprioritisation`
: Optional object. If present, the deprioritisationReq-r11 field is added to the RRC Connection Reject message. Only applicable to LTE cells.
The object must contain the following fields:

  `type` Enumeration ("none", "frequency" or "e-utra").

  `timer` Optional enumeration (5, 10, 15 or 30). Timer in minutes. Required if `type` is not none.

`rrc_cnx_release_extWaitTime`
: Optional integer (range: 0 to 1800). RRC connection release extended wait time in seconds.

`rrc_cnx_release_extWaitTime_CPdata`
: Optional integer (range: 0 to 1800). RRC connection release extended wait time for Control Plane CIoT EPS optimisation in seconds. Only applicable to NB-IoT cells.

`br_ue` Optional object. It can contain the following objects:

  `br_forced_mpdcch_nb_idx`
  : Optional integer (default = -1). Forces the narrow band index used for MPDCCH. The value -1 means that the eNB selects the narrow band automatically.

  `br_forced_pdsch_nb_idx`
  : Optional integer (default = -1). Forces the narrow band index used for PDSCH. The value -1 means that the eNB selects the narrow band automatically.

  `br_forced_pusch_nb_idx`
  : Optional integer (default = -1). Forces the narrow band index used for PUSCH. The value -1 means that the eNB selects the narrow band automatically.

`log_get` Get logs.
Message definition:

`min` Optional number (default = 1). Minimum amount of logs to retrieve. Response won't be sent until this limit is reached (Unless timeout occurs).

`max` Optionnal number (default = 4096). Maximum logs sent in a response.

`timeout` Optional number (default = 1). If at least 1 log is available and no more logs have been geenrated for this time, response will be sent.

`rnti` Optional number. If set, send only logs matching rnti.

`ue_id` Optional number. If set, send only logs with matching ue_id.

`layers` Optional Object. Each member name represents a log layer and values must be string representing maximum level. See [log_options], page 25. If *layers* is not set, all layers level will be set to *debug*, else it will be set to *none*.
Note also the logs is also limited by general log level. See [log_options], page 25.

Response definition:

`logs` Array. List of logs. Each item is a an object with following members:

- `data` Array. Each item is a string representing a line of log.
- `timestamp` Number. Number of seconds since start of session or start of day.
- `layer` String. Log layer.
- `level` String. Log level: *error*, *warn*, *info* or *debug*.
- `dir` Optional string. Log direction: *UL*, *DL*, *FROM* or *TO*.
- `ue_id` Optional number. UE_ID.
- `cell` Optional number (only for PHY layer logs). Cell ID.
- `rnti` Optional number (only for PHY layer logs). RNTI.
- `frame` Optional number (only for PHY layer logs). Frame number (Subframe is decimal part).
- `channel` Optional string (only for PHY layer logs). Channel name.
- `src` String. Server name.
- `idx` Integer. Log index.

`discontinuity` Optional number. If set, this means some logs have been discarded due to log buffer overflow.

Note that only one request can be sent by client.
If a request is sent before previous one has returned, previous one will be sent without matchine min/max/timeout conditions.

`log_reset`
Resets logs buffer.

`quit` Terminates lteenb.

`help` Provides list of available messages in *messages* array of strings and events to register in *events* array of strings.

`stats` Provides statistics.
Every time this message is received by server, statistics are reset.

Response definition:

`time` Time in seconds since LTEENB starting.

`cpu` Object. Each member name defines a type and its value cpu load in % of one core.

`instance_id`
Number. Constant over process lifetime. Changes on process restart.

`counters` Object. List of counters, with following sub members:

- `messages` Object. Each member name is the message name and its value is its occurence.
  To get list of message, type *cevent help msg* in LTEENB monitor.
- `errors` Object. Each member name is the error name and its value is its occurence.
  To get list of message, type *cevent help msg* in LTEENB monitor.

`cells` Object. Each member name is the cell ID and each value is an object representing statistics as follow:

- `dl_bitrate`
  Number. Downlink bitrate in bits per seconds.
- `ul_bitrate`
  Number. Uplink bitrate in bits per seconds.
- `ue_count` Number. Current number of connected UE.
- `mbms_bitrate`
  Number. Broadcast downlink bitrate in bits per seconds.
- `dl_tx` Integer. Number of downlink transmitted packets (Without retransmissions).
- `ul_tx` Integer. Number of uplink transmitted packets (Without retransmissions).
- `dl_retx` Integer. Number of downlink retransmitted packets.
- `ul_retx` Integer. Number of uplink retransmitted packets.
- `dl_use_min`
  Number between 0 and 1. Minimum downlink usage ratio.
- `dl_use_avg`
  Number between 0 and 1. Average downlink usage ratio.
- `dl_use_max`
  Number between 0 and 1. Maximum downlink usage ratio.
- `ul_use_min`
  Number between 0 and 1. Minimum uplink usage ratio.
- `ul_use_avg`
  Number between 0 and 1. Average uplink usage ratio.
- `ul_use_max`
  Number between 0 and 1. Maximum uplink usage ratio.
- `ue_count_min`
  Integer. Minimum number of UE contexts.

`ue_count_max`
: Integer. Maximum number of UE contexts.

`ue_count_avg`
: Integer. Average number of UE contexts.

`erab_count_min`
: Integer. Minimum number of established radio bearer.

`erab_count_max`
: Integer. Maximum number of established radio bearer.

`erab_count_avg`
: Integer. Average number of established radio bearer.

`dl_gbr_use_min`
: Number. Minimum downlink GBR usage ratio.

`dl_gbr_use_avg`
: Number. Average downlink GBR usage ratio.

`dl_gbr_use_max`
: Number. Maximum downlink GBR usage ratio.

`ul_gbr_use_min`
: Number. Minimum uplink GBR usage ratio.

`ul_gbr_use_avg`
: Number. Average uplink GBR usage ratio.

`ul_gbr_use_max`
: Number. Maximum uplink GBR usage ratio.

`rf_ports` Object. Each member name is the RF port ID and each value is an object representing the TX-RX latency statistics (average, max and min values).

9.5 LTE messages

`ue_get` Get ue list.
Message definition:

`ue_id` Optional integer. If set, will filter on UE_ID.

`stats` Optional boolean (default is false). If true, will display stats for each cell

Response definition:

`ue_list` Array of object, representing current connected UEs.
Each element has following definition:

`time` Time in seconds since eNB starting.

`enb_ue_id`
: Integer. eNB UE id.

`mme_ue_id`
: Optional integer. MME UE id. It is present when the UE-associated logical S1-connection is setup.

`rnti` Integer. RNTI.

`cells` Array of object. Each object represent a cell.
First entry is primary cell.
Only *cell_id* is displayed for each cell unless *stats* is set to true.

`cell_id` Number. Cell ID.

`cqi` Number. Last reported cqi.

`ri` Number. Last reported rank indicator.

`dl_bitrate`
Number. Downlink bitrate in bits per seconds.

`ul_bitrate`
Number. Uplink bitrate in bits per seconds.

`dl_tx` Integer. Number of downlink transmitted transport blocks (without retransmissions).

`ul_tx` Integer. Number of received uplink transport blocks (without CRC error).

`dl_retx` Integer. Number of downlink retransmitted transport blocks.

`ul_retx` Integer. Number of received uplink transport blocks with CRC errors.

`dl_mcs` Number. Average downlink MCS.

`ul_mcs` Number. Average uplink MCS.

`turbo_decoder_min`
Integer. Minimum turbo decoder pass.

`turbo_decoder_avg`
Number. Average turbo decoder pass.

`turbo_decoder_max`
Integer. Maximum turbo decoder pass.

`pucch1_snr`
Optional number. PUCCH snr.

`pusch_snr`
Optional number. Last received PUSCH snr.

`erab_get` Get radio bearer list.
Response definition:

`timestamp`
Integer. Timestamp in milliseconds.

`erab_list`
Array of object, representing radio bearers.
Each element has following definition:

`enb_ue_id`
Integer. eNB UE id.

`erab_id` Integer. Radio bearer ID.

`qci` Integer. Bearer QCI.

`dl_mbr` Integer. Downlink maximum bitrate (only if erab is GBR).

`dl_gbr` Integer. Downlink guaranteed bitrate (only if erab is GBR).

`dl_gbr_re`
Integer. Downlink GBR per RE per second (only if erab is GBR).

`ul_mbr` Integer. Uplink maximum bitrate (only if erab is GBR).

`ul_gbr` Integer. Uplink guaranteed bitrate (only if erab is GBR).

`ul_gbr_re`
Integer. Uplink GBR per RE per second (only if erab is GBR).

`dl_total_bytes`
Integer. Total downlink PDCP SDU byte count.

`ul_total_bytes`
Integer. Total uplink PDCP SDU byte count.

`cell_gain`
Set cell DF RF signal gain. See [cell_gain], page 86.
Message definition:

`cell_id` Integer. Cell ID.

`gain` Float. Gain in dB. Must be between -200 and 0 (included).

`rf` Set radio frontend channels gain.
Message definition:

`tx_gain` Optional number or array of numbers. Set TX gain.
Same definition as the [tx_gain], page 28, property.

`tx_channel_index`
Optional number. If set, apply gain to specified channel only.

`rx_gain` Optional number or array of numbers. Set RX gain.
Same definition as the [rx_gain], page 28, property.

`rx_channel_index`
Optional number. If set, apply gain to specified channel only.

Response definition:

`tx_gain` Array. List of TX gain per channel.

`rx_gain` Array. List of RX gain per channel.

`rf_info` Optional string. RF driver information (depends on radio frontend).

`rf_gain` Deprecated. Use [rf], page 81, instead.

`cell_ul_disable`
Enable/disable UL on cell?
Message definition:

`cell_id` Integer. Cell ID.

`disabled` Boolean. Set state

`handover` Triggers a handover.
Message definition:

`enb_ue_id`
Integer. eNB UE id.

`pci` Integer. Physical Cell ID.

`dl_earfcn`
Optional integer. If set use look for cell with this earfcn, else use UE current earfcn.

`type` Optional string. Can be `auto` (default), `intra`, `s1` or `x2`.

`rrc_cnx_release`
Forces a RRC Connection release.
Message definition:

`enb_ue_id`
Integer. eNB UE id.

`redirect` Optional integer. If set, defines RRC redirection index (See [rrc_redirect], page 49).

`rrc_ue_info_req`
Sends a UE Information Request message.
Message definition:

`enb_ue_id`
Integer. eNB UE id.

`req_mask` Integer. Bitmap of the information to request (bits: 0:RACH, 1:RLF, 2:LogMeas, 3:ConnEst, 4:MobHist).

`x2` Get X2 peers state.
Response definition:

`peers` Array of object. One for each peer.
Each element has the following definition:

`state` String. Can be `connecting`, `connected` or `setup_done`.

`addr` String. Address of peer

`cells` Array of object. One for each cell. Each element has the following definition:

`cell_id` Integer. Cell ID.

`tac` Integer. TAC.

`dl_earfcn`
Integer. Downlink cell EARFCN.

`pci` Integer. Physical Cell ID

`x2connect`
Forces connection to a X2 peer.
Message definition

`addr` String. X2 peer address.

`s1` Get MME link state.
Response definition:

`s1_list` Array of object. One for each MME connection defined as follow:

`state` Link state: *disconnected*, *connecting*, *connected*, *inactive* or *setup_done*.

`address` MME address.

`PLMN` If connection complete, PLMN.

`s1connect`
Forces connection to a MME.
Message definition

`addr` Optional string. If not set, will try to connect to all registered MME, else will try with the specified address.

`s1disconnect`
Forces disconnection from a MME.
Message definition

`addr` Optional string. If not set, will to disconnect from all registered MME, else will try with the specified address.

`s1add` Adds a new MME to the list of S1AP connections.
Message definition
The message must contain the same parameters as one of the object defined in mme_list array. See [mme_list], page 27.

`s1delete` Removes a MME address from the lsit of S1AP connections.
Message definition

`addr` String. MME address to be removed from the list.

`m2` Get M2AP link state.
Response definition:

`state` Link state: *disconnected*, *waiting*, *connecting*, *connected*.

`address` MBMSGW address.

`m2connect`
Forces connection to a MBMSGW.
Message definition

`addr` Optional string. If not set, the eNB will try to connect to the previously configured address.

`m2disconnect`
Releases connection to a MBMSGW.

`sib14` Starts / stops SIB14 scheduling and defines its content.
Message definition

`cell_id` Integer. Cell ID.

`enabled` Boolean. If set to true, SIB14 is scheduled.

`si_periodicity`
Optional enumeration: 8, 16, 32, 64, 128, 256, 512 (for LTE cells) or 64, 128, 256, 512, 1024, 2048, 4096 (for NB-IoT cells). Sets the periodicity (in frames) of the transmission of SIB 14. Required if `enabled` is set to true.

`si_repetition_pattern`
Optional enumeration: 2, 4, 8, 16. Sets the repetition pattern (in frames) of the transmission of SIB14 in the NB-IoT cell. Required if `enabled` is set to true.

`config` Optional object or array of object. Required if `enabled` is set to true.
If `config` is an object, SIB14 contains a common configuration.
If `config` is an array, SIB14 contains a per PLMN configuration and you must define as many objects as the number of PLMNs defined in SIB1.
Each object contains the following fields (see 3GPP 36.331 for details):

`category` Enumeration: "a", "b", or "c".

`barring bitmap`
String. Bit string of 10 bits.

`barring_for_exception_data`
Optional boolean. Only used for NB-IoT cells.

`barring_for_special_ac`
Optional string. Bit string of 5 bits, mandatory for NB-IoT cells.

`page_ue` Sends a paging message for a UE on a list of cells.
Message definition

`type` Enumeration ("normal", "cat0", "ce" or "nb-iot"). Defines the type of UE to be paged.

`cn_domain`
Optional enumeration ("cs" or "ps"). Not required for NB-IoT UEs.

`imsi` String. IMSI of the UE to be paged.

`"s-tmsi"` Optional object. S-TMSI to be used for the paging identity. If the object is not present, the UE is paged by its IMSI.
The object must contain the following fields:

`mmec` Integer.

`m-tmsi` Integer.

`cell_id` Array. The array contains the cell_id of the cells on which the paging message must be transmitted.

`snr` Sets the SNR (relative to the CRS level) when the channel simulator is enabled.
Message definition

`snr` Float. Defines the SNR value to be set.

`channel` Optional integer. Defines the TX channel number on which the the new SNR value is applied. If not present, the new SNR value is applied on all TX channels.

`ncell_list_add`
Add a new neighbour cell to the `ncell_list` object.
Message definition

`cell_id` Integer. Cell ID.

`ncell` Object. Contains the same parameters as those defined for `ncell_list` object. See [ncell_list], page 33.

`ncell_list_del`
Remove a neighbour cell from the `ncell_list` object.
Message definition

`cell_id` Integer. Cell ID.

`n_id_cell`
: Integer (range 0 to 503). Physical cell identity.

`dl_earfcn`
: Optional integer (range 0 to 262143). DL EARFCN. If not present, it is assumed to be the same as the current cell.

9.6 Examples

1. Config
   1. Client sends

```
{
    "message": "config_get",
    "message_id": "foo"
}
```

   2. Server replies

```
{
    "message_id": "foo",
    "message": "config_get",
    "name": "UE",
    "logs": {
        "phy": {
            "level": "error",
            "max_size": 0
        },
        ...
        "rrc": {
            "level": "debug",
            "max_size": 1
        }
    }
}
```

2. Error
   1. Client sends

```
{
    "message": "bar",
    "message_id": "foo"
}
```

   2. Server replies

```
{
    "message_id": "foo",
    "message": "bar",
    "error": "Unknown message: bar"
}
```

10 Command line monitor reference

The following commands are available:

`help` Display the help. Use `help` *`command`* to have a more detailed help about a command.

`t [ue|g|cpu|spl] [`*`period`*`]`
: Activate various traces on the console. The display is stopped when typing return. The default trace is `ue`. An optional display period (in seconds) is accepted.

  Available traces:

  `ue[:n]` UE MAC and PRACH traces. If `n` is provided, only display the UE ID `n`.

  `g` Show global eNodeB statistics.

  `cpu` Display the CPU usage from the TRX (transceiver) API and the TX-RX latency statistics.

  `spl` Display various statistics about the sent and received complex samples (at the TRX API level). For the TX side, the RMS, PAPR and maximum sample value are displayed. The number of saturation events (`abs(sample) > 1`) are displayed too. For the RX side the RMS and maximum sample value are displayed. The unit is dB FS (dB Full Scale). 0 dB FS is reached with a square signal of amplitude 1.

`log [`*`log_options`*`]`
: Display the current log state. If *log_options* are given, change the log options. The syntax is the same as the `log_options` configuration property.

`cell` List the available cells.

`cell_gain` *`cell_id gain`*
: Set the DL gain of the cell *cell_id*. The gain is in dB and must be <= 0. The gain of the other cells is not modified.

`cell_ul_disable` *`cell_id flag`*
: Disable the uplink of the cell *cell_id* if *flag* = 1.

`snr level [channel]`
: Change the SNR level. If `channel` is not provided, the same SNR is set for all the TX channels. This command only applies if the channel simulator is configured. See [Channel simulator], page 67.

`ue` List connected UEs.

`handover` *`eNB_UE_ID pci`* `[`*`dl_earfcn`*`]`
: Initiate a handover of UE *eNB_UE_ID* to the cell of physical identifier *pci* at EARFCN *dl_earfcn*. If the EARFCN is not given, it is assumed to be the same as the source cell. The target cell must be defined in the source cell neighbour list.

`pcap [-w` *`filename`*`] [-l` *`data_len`*`] [-b] [-d` *`ms`*`] [-p]`
: Record packet data in the pcap format used by Wireshark. By default data are written until a `pcap_stop` request is made. To record for a fixed period of time the *-d* can be used to specify the number of milliseconds to capture data. The remaining cmd line options mimic the control found in the config file: the *-w* option can be use to specify an output file name (default is */tmp/enb.pcap*); the *-l* option specifies the maximum length for packet data written (default is 65535); the *-b* option enables capture of broadcast packets on the BCCH channel. the *-p* option can be set to capture into a pipe instead of a file.

`pcap_stop`
: Stop recording pcap packet data.

`rf_info`
: Get RF driver informations

`tx_gain` *`gain channel`*
: Set the TX gain in dB of the radio driver. If no channel is specified, all cells are affected. Same definition as the [tx_gain], page 28, property.

`rx_gain` *`gain channel`*
: Set the RX gain in dB of the radio driver. If no channel is specified, all cells are affected. Same definition as the [rx_gain], page 28, property.

`s1`
: Dump the S1 connection state. It is useful to see if the eNodeB is connected to the MME.

`s1connect [`*`mme_addr`*`]`
: Force a S1 (re)connection to the MME. The MME IP address and optional port can be given as an optional parameter.

`s1disconnect`
: Force a S1 disconnect from the MME.

`x2`
: Display the state of the X2 connections and the associated cell parameters.

`x2connect` *`peer_addr`*
: Force a X2 connection to eNodeB *peer_addr*.

`x2disconnect` *`peer_addr`*
: Force a X2 disconnection from the eNodeB *peer_addr*.

`m2`
: Display the state of the M2 connection.

`m2connect [`*`server_addr`*`]`
: Force a M2 connection to MBMSGW *server_addr*. If *server_addr* is not present, it uses the previous address.

`m2disconnect`
: Force a M2 disconnection from the MBMSGW.

`hwcaps`
: Show the CPU capabilities. Useful to see if AES acceleration is supported.

`mbms`
: Show the MBMS status. It is useful to see packet losses, the instantaneous bitrate of each session and the maximum bitrate allowed for each PMCH.

`erab [-a]`
: Show the allocated radio bearers (only GBR bearers by default, all the bearers with the `-a` option).

`rrc_ue_info_req` *`UE_ID req_mask`*
: Send a RRC UE Information Request to UE *UE_ID*. 'req_mask' is a bitmask: 0:RACH, 1:RLF, 2:LogMeas, 3:ConnEst, 4:MobHist

`rrc_cnx_release` *`UE_ID`* `[`*`redirect_type`*`]`
: Forces a RRC connection release. See [rrc_cnx_release], page 82, in remote API.

`rlc_drop_rate` *`UE_ID drb_id rate`*
: Define a *rate* percentage of uplink RLC PDUs dropped.

11 Log file format

11.1 PHY layer

When a PHY message is dumped (debug level), the format is:

```
time layer dir ue_id cell rnti frame.subframe channel:short_content
   long_content
```

`time` Time using the selected format.

`layer` Layer (`[PHY]` here).

`dir` UL (uplink) or DL (downlink).

`ue_id` eNodeB UE identifier (hexadecimal, unique among all cells).

`cell` Low 8 bits of the cell identifier (hexadecimal).

`rnti` Associated RNTI (hexadecimal) or `-` if none.

`frame.subframe`
Frame number (0-1023) and subframe number (0-9).

`channel` PHY channel name (PUSCH, PUCCH, PRACH, SRS, PSS, PBCH, PCFICH, PDSCH, PHICH, PDCCH or EPDCCH).

`short_content`
Single line content.

`long_content`
Hexadecimal dump of the message if `phy.max_size > 0`.

11.2 MAC, RLC, PDCP, RRC and NAS layers

When a message is dumped, the format is:

```
time layer - ue_id message
```

When a PDU is dumped (debug level), the format is:

```
time layer dir ue_id short_content
   long_content
```

`time` Time using the selected format

`layer` Layer (`[MAC]`, `[RLC]`, `[PDCP]`, `[RRC]` or `[NAS]` here).

`dir` UL (uplink) or DL (downlink).

`ue_id` eNodeB UE identifier (hexadecimal, unique among all cells).

`short_content`
Single line content.

- RLC, PDCP: preceded by the SRB or DRB identifier.

`long_content`

- MAC, RLC, PDCP: hexadecimal dump of the message if `layer.max_size > 0`.
- RRC: full ASN.1 content of the RRC message if `layer.max_size > 0`.
- NAS: full content of the NAS message if `layer.max_size > 0`.

11.3 S1AP, X2AP, M2AP and GTP-U layers

When a message is dumped, the format is:

```
time layer - message
```

When a PDU is dumped (debug level), the format is:

```
time layer dir ip_address short_content
   long_content
```

`time` Time using the selected format.

`layer` Layer (e.g. `[S1AP]`).

`dir` Direction: `TO` or `FROM`.

`ip_address`
Source or destination IP address, depending on the `dir` field.

`short_content`
Single line content.

`long_content`

- S1AP, X2AP, M2AP: full ASN.1 content of the message if `layer.max_size > 0`.
- GTPU: hexadecimal dump of the message if `layer.max_size > 0`.

12 License

Abbreviations

APN	Access Point Name
CA	Carrier Aggregation
CIoT	Cellular Internet of Things
CQI	Channel Quality Indication
DL	Downlink
DRB	Data Radio Bearer
EARFCN	E-UTRA Absolute Radio Frequency Channel Number
EPRE	Energy Per Resource Element
E-UTRA	Evolved UMTS Terrestrial Radio Access
FDD	Frequency Division Duplexing
HARQ	Hybrid Automatic Repeat reQuest
HSS	Home Subscriber Server
IMEI	International Mobile Equipment Identity
IMSI	International Mobile Subscriber Identity
LTE	Long Term Evolution
MAC	Media Access Control
MBSFN	Multicast-Broadcast Single-Frequency Network
MBMS	Multimedia Broadcast Multicast Service
MCC	Mobile Country Code
MIMO	Multiple-Input Multiple-Output
MME	Mobility Management Entity
MNC	Mobile Network Code
NAS	Non Access Stratum
NB-IoT	Narrow Band Internet of Things
PAPR	Peak to Average Power Ratio
PDCP	Packet Data Convergence Protocol
PDN	Packet Data Network
PLMN	Public Land Mobile Network
PMI	Precoding Matrix Indicator
PRS	Positioning Reference Signals
QCI	QoS Class Identifier
QoS	Quality of Service
RB	Resource Block
RI	Rank Indicator
RLC	Radio Link Control

RMS	Root Mean Square
ROHC	Robust Header Compression
RRC	Radio Resource Control
SIB	System Information Block
SISO	Single-Input Single-Output
TDD	Time Division Duplexing
TMSI	Temporary Mobile Subscriber Identity
UE	User Equipment
UL	Uplink
USIM	Universal Subscriber Identity Module