Tutorial: A framework for dense vRAN deployments with OpenAirInterface

Beurdouche Romain and Raymond Knopp
Communication Systems dept.
EURECOM
Biot, France

August 18, 2026

Abstract

The advent of 5G virtualized Radio Access Networks (vRANs) brings a new challenge with regards to computer architectures. It requires to design computing technologies that provide appropriate performance while maximizing the efficiency and flexibility of networks. Several solutions were proposed relying on general-purpose processors as well as hardware accelerators. This document describes how to create dense vRAN deployments on top of some of these computer architectures using the 5G software stack OpenAirInterface.

Introduction

This document is intended to provide the necessary instructions and pointers to materials for reproducing the dense vRAN deployments using the OpenAirInterface 5G stack on a variety of computing systems that were introduced in the conference article *A framework for experimental dense vRAN deployments* [1] by Romain Beurdouche and Raymond Knopp.

All the materials pointed out in this document are publicly available online at:

`https://gitlab.eurecom.fr/beurdouc/materials_wintech_2026_2`

Or from OpenAirInterface public repository:

`https://github.com/duranta-project/openairinterface5g.git`

1 Common Prerequisites

1.1 System setup

The three systems on which this framework can be deployed are introduced and described in the conference article [1] including their hardware configuration and operating systems.

The systems shall be configured to operate with an O-RAN 7.2 fronthaul following the instructions from the documentation of OpenAirInterface:

`https://github.com/duranta-project/openairinterface5g/blob/2026.w32/doc/ORAN_FHI7.2_Tutorial.md`

You can follow this tutorial down to installing DPDK 22.11.11 on the HP-GPP and vRAN-SoC systems. We assume that DPDK 22.11.11 was uncompressed and built in directory `~/dpdk-stable-22.11.11`. On the E-GPP & RFSoC system, stop before DPDK installation as it requires a special DPDK version.

On the HP-GPP and E-GPP & RFSoC systems, isolate processor cores 7 to 63. On the vRAN-SoC system, isolate processor cores 5 to 32.

The framework also requires docker and docker compose to deploy the network instances.

1.2 OpenAirInterface

The three setups presented below all rely on the OpenAirInterface 5G stack with different sets of features depending on the system. Then, the three setups require a copy of the repository at tag *2026.w35* or latter that is assumed to be located at `~/openairinterface5g`.

```
git clone https://github.com/duranta-project/openairinterface5g.git ~/openairinterface5g
```

2 HP-GPP system

2.1 Create the Network VFs

You can retrieve all the files necessary to operate the framework on the E-GPP & RFSoC system from the materials repository in the `E-GPP_and_RFSoC` directory.

```
https://gitlab.eurecom.fr/beurdouc/materials_wintech_2026_2/-/tree/main/HP-GPP
```

Copy the content of this directory to the home directory on the system. The framework files are designed assuming they are located in the system home. Otherwise, the files should be adapted.

The VFs of the Network Interface Card (NIC) for the O-RAN 7.2 fronthaul interface can then be created and configured by using the scripts in the `~/setup_scripts` directory. First use the `setup_sriov_vfs.sh` script to create the VFs:

```
~/setup_scripts/setup_sriov_vfs.sh
```

It may be necessary to change the interface name and the NIC VF addresses in all the `setup_sriov_<n>.sh` scripts. The addresses of all the VFs is displayed by `~/dpdk-stable-22.11.11/usertools/dpdk-devbind.py -s`.

The first VF configured with `setup_sriov_1.sh` corresponds to the network instance connecting to a real O-RU. The VLAN tag and O-DU MAC addresses should be changed to match with the actual fronthaul network and O-RU. Then you can run the `setup_sriov_<n>.sh` scripts to configure each VF:

```
~/setup_scripts/setup_sriov_1.sh
~/setup_scripts/setup_sriov_2.sh
~/setup_scripts/setup_sriov_3.sh
~/setup_scripts/setup_sriov_4.sh
~/setup_scripts/setup_sriov_5.sh
~/setup_scripts/setup_sriov_6.sh
~/setup_scripts/setup_sriov_7.sh
```

2.2 Build the docker image

Use the Dockerfiles from the OpenAirInterface repository to build the OpenAirInterface softmodem docker image:

```
cd
docker build -f openairinterface5g/docker/Dockerfile.base.ubuntu -t ran-base:latest openairinterface5g
docker build -f openairinterface5g/docker/Dockerfile.build.fhi72.ubuntu -t ran-build-fhi72:latest openairinterface5g
docker build -f openairinterface5g/docker/Dockerfile.gNB.fhi72.ubuntu -t oai-gnb-fhi72:latest openairinterface5g
```

2.3 Configure the network

The configuration files of the network instances are located in the directory `~/conf_files`. The configuration file `gnb.sa.band77.273prb.fhi72.4x4-fhi72-1o7.conf` corresponds to the network instance with a real O-RU. The MAC address of the O-RU should go in the `ru_addr` field.

It may be necessary to change in all the configuration files the addresses of the NIC VFs in the field `dpdk_devices`. The addresses of all these VFs is displayed by `~/dpdk-stable-22.11.11/usertools/dpdk-devbi`

2.4 Mount the L1 stats logs

In order to access the L1 stats log of the OpenAirInterface softmodem from outside the docker containers, the framework includes the automatic mounting of the L1 stats log in the `~/logs` directory. The log files should be created and owned by `root` before deploying the containers:

```
cd
mkdir logs
mkdir logs/openair-gnb-1
sudo touch logs/openair-gnb-1/nrL1_stats.log
mkdir logs/openair-gnb-2
sudo touch logs/openair-gnb-2/nrL1_stats.log
mkdir logs/openair-gnb-3
sudo touch logs/openair-gnb-3/nrL1_stats.log
mkdir logs/openair-gnb-4
sudo touch logs/openair-gnb-4/nrL1_stats.log
mkdir logs/openair-gnb-5
sudo touch logs/openair-gnb-5/nrL1_stats.log
mkdir logs/openair-gnb-6
sudo touch logs/openair-gnb-6/nrL1_stats.log
mkdir logs/openair-gnb-7
sudo touch logs/openair-gnb-7/nrL1_stats.log
```

Then, the logs can be read or copied at any time while the networks are executing.

2.5 Deploy the framework

The compose files to deploy the framework using docker compose are located in `~/compose`. The framework includes a local core network. You may want to add the UEs you wish to connect to the first gNB in the core network database `~/compose/database/oai_db.sql`. You can also

change the core network configuration — for example the PLMN or available networks — in `~/compose/conf`. If you modify the core network configuration, adapt the configuration of the gNBs accordingly.
Then, you can deploy the network instances:

```
docker compose -f ~/compose/docker-compose-int-cn-5o7-12345.yaml up -d
```

To undeploy:

```
docker compose -f ~/compose/docker-compose-int-cn-5o7-12345.yaml down
```

3 E-GPP & RFSoC system

3.1 DPDK for the AMD T2 RFSoC

Using the AMD T2 RFSoC of this system with Virtual Functions (VFs) requires to use DPDK 20.11.9 with the patch `ACCL_BBDEV_DPDK20.11.3_ldpc_3.2.patch` provided by AMD. Instructions to build and install this version of DPDK are provided in the documentation of OpenAirInterface:

https://github.com/duranta-project/openairinterface5g/blob/2026.w32/doc/LDPC_OFFLOAD_SETUP.md

DPDK should be installed in location `/opt/dpdk` by configuring DPDK build with meson:

```
meson setup --prefix=/opt/dpdk build
```

3.2 Creating 7 AMD T2 VFs

Follow the instructions from OpenAirInterface documentation to create the VFs of the AMD T2 RFSoC:

https://github.com/duranta-project/openairinterface5g/blob/2026.w32/doc/LDPC_OFFLOAD_SETUP.md#binding-the-accelerator-with-vfio-pci

Create 7 VFs instead of the 2 of the tutorial.

3.3 Create the Network VFs

You can retrieve all the files necessary to operate the framework on the E-GPP & RFSoC system from the materials repository in the `E-GPP_and_RFSoC` directory.

https://gitlab.eurecom.fr/beurdouc/materials_wintech_2026_2/-/tree/main/E-GPP_and_RFSoC

Copy the content of this directory to the home directory on the system. The framework files are designed assuming they are located in the system home. Otherwise, the files should be adapted.

The VFs of the Network Interface Card (NIC) for the O-RAN 7.2 fronthaul interface can then be created and configured by using the scripts in the `~/setup_scripts` directory. First use the `setup_sriov_vfs.sh` script to create the VFs:

```
~/setup_scripts/setup_sriov_vfs.sh
```

It may be necessary to change the interface name and the NIC VF addresses in all the `setup_sriov_<n>.sh` scripts. The addresses of all the VFs is displayed by `/opt/dpdk/bin/dpdk-devbind.py -s`.

The first VF configured with `setup_sriov_1.sh` corresponds to the network instance connecting to a real O-RU. The VLAN tag and O-DU MAC addresses should be changed to match with the actual fronthaul network and O-RU. Then you can run the `setup_sriov_<n>.sh` scripts to configure each VF:

```
~/setup_scripts/setup_sriov_1.sh
~/setup_scripts/setup_sriov_2.sh
~/setup_scripts/setup_sriov_3.sh
~/setup_scripts/setup_sriov_4.sh
~/setup_scripts/setup_sriov_5.sh
~/setup_scripts/setup_sriov_6.sh
~/setup_scripts/setup_sriov_7.sh
```

3.4 Build the docker image

The docker builder can only read document from the path provided as argument. Since OpenAirInterface requires the special DPDK that we installed in `/opt/dpdk` to build, we should make a copy of it in the home directory:

```
cp -r /opt/dpdk ~/
```

Then, use the files of the `~/Dockerfiles` directory to build the OpenAirInterface softmodem docker image:

```
cd
docker build -f openairinterface5g/docker/Dockerfile.base.ubuntu -t ran-base:latest openairinterface5g
docker build -f Dockerfiles/Dockerfile.build.fhi72.ubuntu -t ran-build-fhi72:latest ./
docker build -f Dockerfiles/Dockerfile.gNB.fhi72.ubuntu -t oai-gnb-fhi72:latest openairinterface5g
```

3.5 Configure the network

The configuration files of the network instances are located in the directory `~/conf_files`. The configuration file `gnb.sa.band77.273prb.fhi72.4x4-fhi72-1o7.conf` corresponds to the network instance with a real O-RU. The MAC address of the O-RU should go in the `ru_addr` field.

It may be necessary to change in all the configuration files the addresses of the NIC VFs in the field `dpdk_devices` and of the AMD T2 RFSoC VFs in the field `dpdk_dev`. The addresses of all these VFs is displayed by `/opt/dpdk/bin/dpdk-devbind.py -s`.

3.6 Mount the L1 stats logs

In order to access the L1 stats log of the OpenAirInterface softmodem from outside the docker containers, the framework includes the automatic mounting of the L1 stats log in the `~/logs`

directory. The log files should be created and owned by `root` before deploying the containers:

```
cd
mkdir logs
mkdir logs/openair-gnb-1
sudo touch logs/openair-gnb-1/nrL1_stats.log
mkdir logs/openair-gnb-2
sudo touch logs/openair-gnb-2/nrL1_stats.log
mkdir logs/openair-gnb-3
sudo touch logs/openair-gnb-3/nrL1_stats.log
mkdir logs/openair-gnb-4
sudo touch logs/openair-gnb-4/nrL1_stats.log
mkdir logs/openair-gnb-5
sudo touch logs/openair-gnb-5/nrL1_stats.log
mkdir logs/openair-gnb-6
sudo touch logs/openair-gnb-6/nrL1_stats.log
mkdir logs/openair-gnb-7
sudo touch logs/openair-gnb-7/nrL1_stats.log
```

Then, the logs can be read or copied at any time while the networks are executing.

3.7 Deploy the framework

The compose files to deploy the framework using docker compose are located in `~/compose`.
The framework includes a local core network. You may want to add the UEs you wish to connect to the first gNB in the core network database `~/compose/database/oai_db.sql`. You can also change the core network configuration — for example the PLMN or available networks — in `~/compose/conf`. If you modify the core network configuration, adapt the configuration of the gNBs accordingly.
Then, you can deploy the network instances:

```
docker compose -f ~/compose/docker-compose-int-cn-7o7-1234567.yaml up -d
```

To undeploy:

```
docker compose -f ~/compose/docker-compose-int-cn-7o7-1234567.yaml down
```

4 vRAN-SoC system

4.1 vRAN Boost setup

Intel vRAN Boost a.k.a. VRB1 a.k.a. ACC200 is the accelerator embedded in the processor of the vRAN-SoC system. Follow the instructions from OpenAirInterface documentation to setup the VRB1 / ACC200:

`https://github.com/duranta-project/openairinterface5g/blob/2026.w32/doc/LDPC_OFFLOAD_SETUP.md`

Create 3 VFs instead of the only one of the tutorial.

4.2 Create the Network VFs

You can retrieve all the files necessary to operate the framework on the vRAN-SoC system from the materials repository in the `vRAN-SoC` directory.

```
https://gitlab.eurecom.fr/beurdouc/materials_wintech_2026_2/-/tree/main/vRAN-SoC
```

Copy the content of this directory to the home directory on the system. The framework files are designed assuming they are located in the system home. Otherwise, the files should be adapted.

The VFs of the Network Interface Card (NIC) for the O-RAN 7.2 fronthaul interface can be created and configured by using the scripts in the `~/setup_scripts` directory. First use the `setup_sriov_vfs.sh` script to create the VFs:

```
~/setup_scripts/setup_sriov_vfs.sh
```

It may be necessary to change the interface name and the NIC VF addresses in all the `setup_sriov_<n>.sh` scripts. The addresses of all the VFs is displayed by `~/dpdk-stable-22.11.11/usertools/dpdk-devbind.py -s`.

The first VF configured with `setup_sriov_1.sh` corresponds to the network instance connecting to a real O-RU. The VLAN tag and O-DU MAC addresses should be changed to match with the actual fronthaul network and O-RU. Then you can run the `setup_sriov_<n>.sh` scripts to configure each VF:

```
~/setup_scripts/setup_sriov_1.sh
~/setup_scripts/setup_sriov_2.sh
~/setup_scripts/setup_sriov_3.sh
```

4.3 Build the docker image

Use the files of the `~/Dockerfiles` directory to build the OpenAirInterface softmodem docker image:

```
cd
docker build -f openairinterface5g/docker/Dockerfile.base.ubuntu -t ran-base:latest openairinterface5g
docker build -f Dockerfiles/Dockerfile.build.fhi72.ubuntu -t ran-build-fhi72:latest openairinterface5g
docker build -f Dockerfiles/Dockerfile.gNB.fhi72.ubuntu -t oai-gnb-fhi72:latest openairinterface5g
```

4.4 Configure the network

The configuration files of the network instances are located in the directory `~/conf_files`. The configuration file `gnb.sa.band77.273prb.fhi72.4x4-fhi72-1o3.conf` corresponds to the network instance with a real O-RU. The MAC address of the O-RU should go in the `ru_addr` field. The `vfio_vf_token` also need to be changed to the value configured through `pf_bb_config`.

It may be necessary to change in all the configuration files the addresses of the NIC VFs in the field `dpdk_devices` and of the accelerator VFs in the field `dpdk_dev`. The addresses of all these VFs is displayed by
`~/dpdk-stable-22.11.11/usertools/dpdk-devbind.py -s`.

4.5 Mount the L1 stats logs

In order to access the L1 stats log of the OpenAirInterface softmodem from outside the docker containers, the framework includes the automatic mounting of the L1 stats log in the `~/logs` directory. The log files should be created and owned by `root` before deploying the containers:

```
cd
mkdir logs
mkdir logs/openair-gnb-1
sudo touch logs/openair-gnb-1/nrL1_stats.log
mkdir logs/openair-gnb-2
sudo touch logs/openair-gnb-2/nrL1_stats.log
mkdir logs/openair-gnb-3
sudo touch logs/openair-gnb-3/nrL1_stats.log
```

Then, the logs can be read or copied at any time while the networks are executing.

4.6 Deploy the framework

The compose files to deploy the framework using docker compose are located in `~/compose`.
The framework includes a local core network. You may want to add the UEs you wish to connect to the first gNB in the core network database `~/compose/database/oai_db.sql`. You can also change the core network configuration — for example the PLMN or available networks — in `~/compose/conf`. If you modify the core network configuration, adapt the configuration of the gNBs accordingly.
Then, you can deploy the network instances:

```
docker compose -f ~/compose/docker-compose-int-cn-3o3-123.yaml up -d
```

To undeploy:

```
docker compose -f ~/compose/docker-compose-int-cn-3o3-123.yaml down
```

5 Data collection

5.1 High-PHY processing times

Processing times recorded in the physical layer of the first gNB `openair-gnb-1` can be recorded at any time from the L1 stats log. You can use the following command to display the content of the l1 stats log in real time and press `Ctrl + C` to record the file when the times are consistent:

```
watch -n0.1 'cat ~/logs/openair-gnb-1/nrL1_stats.log | tee dst.log'
```

The plots in [1] are then obtained with the PGFPlots LaTeXpackage [3].

5.2 Power consumption

The power consumed by a server can be measured through a standard Redfish [2] management interface. Redfish enables querying system status information with HTTP through the management interface of a server. Querying the system status information requires to have access to the

management interface and to have the login information to connect to the management interface. Then, the status can be queried and processed, for example with the following command:

```
watch -n 1 "curl -k -u <login>:<password> https://<server management interface address>/redfish/v1/Chassis/1/Power
| jq -r | grep -A13 PowerControl | grep Watts | tee dst.log"
```

This command displays the server consumption in real time until pressing `Ctrl + C` to record a snapshot to a file. The command may need to be adapted depending on the server and the behavior of Redfish on that server.

References

[1] Romain Beurdouche and Raymond Knopp. A framework for experimental dense vRAN deployments. In *Proceedings of the 20th ACM Workshop on Wireless Network Testbeds, Experimental Evaluation & Characterization*, WiNTECH '26, Austin, TX, USA, 2026. Association for Computing Machinery.

[2] DMTF. Redfish®. Redfish website, [Accessed: 17-April-2026].

[3] Christian Feuersänger and Henri Menke. PGFPlots. CTAN, [Accessed: 26-August-2026].