Document Number**:** H2020-EUK-815323/5G-ALLSTAR/D5.2

Project Name:
5G AgiLe and fLexible integration of SaTellite And cellulaR (5G-ALLSTAR)

Deliverable D5.2

Integration and system level testing for European testbed of 5G cellular and satellite access networks

Date of delivery:	30/06/2021	Version:	1.0
Start date of Project:	01/07/2018	Duration:	36+4 months

Deliverable D5.2

Integration and system level testing for European testbed of 5G cellular and satellite access networks

Project Number:	H2020-EUK-815323
Project Name:	5G AgiLe and fLexible integration of SaTellite And cellulaR

Document Number:	H2020-EUK-815323/5G-ALLSTAR/D5.2
Document Title:	Integration and system level testing for European testbed of 5G cellular and satellite access networks
Editor(s):	Guido Casati (FhG IIS)
Authors:	Guido Casati (FhG IIS), Marjorie Thary (TAS), Nicolas Cassiau (CEA), David Miras (CEA), Alessandro Giuseppi (CRAT), Federico Lisi (CRAT), Antonio Ornatelli (CRAT), Vito Andrea Racanelli (CRAT), Emanuele De Santis (CRAT), Andrea Tortorelli (CRAT)
Dissemination Level:	Public
Contractual Date of Delivery:	30/06/2021
Security:	Public
Status:	Reviewed
Version:	1
File Name:	5G-ALLSTAR_D5.2.docx

Abstract

This deliverable was created as part of the project Work Package 5 “Prototyping, Validation and Demonstration” activities, and details implementation, integration and testing of the European Testbed.

Keywords

5G ; terrestrial testbed ; satellite testbed; multi-connectivity; hardware; software

Executive Summary

This deliverable compiles the work that has been done to implement the EU testbed. All the software and hardware components and the architecture of the testbed are described. The incremental method to get to the final testbed is provided. Tests realized to validate the subsystems can also be found.

Contents

List of Figures

List of Tables

List of Abbreviations

3GPP	3rd Generation Partnership Project
5G	Fifth Generation
5G-ALL-STAR	5G AgiLe and fLexible integration of SaTellite And cellulaR
ASO	Aerospace Option
BF-OFDM	Block-Filtered OFDM
C&M	Command & Monitoring
CN	Core Network
cRAN	Centralised RAN
dRAN	Distributed RAN
cRRM	Centralized RRM
DAC	Digital to Analog Converter
DL	Downlink
dRRM	Distributed RRM (dRRM)
DSP	Digital Signal Processing
FD	Frequency Domain
FDD	Frequency Division Duplexing
FPGA	Field Programmable Gate Array
FR1	(5G) Frequency Range 1: from 450 MHz to 6 GHz
FR2	(5G) Frequency Range 2: from 24.25 GHz to 52.6 GHz
FhG IIS	Fraunhofer IIS
GEO	Geostationary Orbit
gNB-CU	gNB Centralized Unit
gNB-DU	gNB Distributed Unit
HW	Hardware
I&V	Integration and Validation
IDFT	Inverse Discrete Fourier Transform

KPI	Key Performance Indicator
LOS	Line of Sight
MAC	Medium Access Control
MCS	Modulation and Coding Scheme
MODCOD	Modulation and Coding scheme
NR	New Radio
NTN	Non-Terrestrial Network
OAI	OpenAirInterface
PoC	Proof of Concept
PDCCH	Physical Downlink Control Channel
QoE	Quality of Experience
RA	Random Access
RAN	Radio Access Network
RAT	Radio Access Technology
RF	Radio Frequency
RRM	Radio Resource Management
SAT	Satellite
SINR	Signal-to-Interference-plus-Noise Ratio
SDR	Software Defined Radio
SoC	System on Chip
SW	Software
TERR	Terrestrial
TN	Terrestrial Network
UE	User Equipment
UL	Uplink
VM	Virtual Machine

1 Introduction

This deliverable, D5.2, reports the integration and testing of the European (EU) testbed within the framework of the 5G-ALLSTAR project.

The unitary testbed software (SW) and hardware (HW) components were described in detail in Deliverable 5.1. Nevertheless, since this latter deliverable was provided two years prior to D5.2, huge progress has been realized since then. We therefore first report hereafter the final status of the components and then their integration into (i) the core testbed, (ii) the satellite (SAT) testbed, (iii) the terrestrial (TERR) testbed and finally (iv) the multi-connectivity testbed.

Important note concerning the multi-connectivity testbed

Originally, the testbed described in this document was planned to be integrated in a unique place (Fraunhofer premises in Erlangen). Nevertheless, the Covid-19 crisis made it impossible to travel. EU partners decided to try to integrate the testbed anyway, connecting the different sites where the HW was, i.e. Erlangen (SAT testbed) and CEA Grenoble (TERR and core testbed). It resulted the architecture described on Figure 2. Great effort was done by all partners to adapt to this new situation: software was compiled into Virtual Machines and provided to CEA (see Figure 9), support was done remotely by teleconferences and a VPN was initiated between Grenoble and Erlangen. Nevertheless, concerning this latter point, the informatics security rules in CEA prevented us from circulating data as expected. The IT department in CEA tried to get around the rules, without success. We therefore were not able to run multi-connectivity scenarios as planned. However, several validation tests, described in this document, proved the validity of our concepts and realizations and allowed to reach, fully or partially, some objectives of the project.

In section 2 the architecture of the testbed is described. First, the final status of the hardware components is provided; we also show how they assembly in the testbed. The software components and the software architecture are then depicted in this section, together with configuration parameters. Finally, the description of the tool used for the automation of the tests, OpenBACH, can be found.

Before the integration of the testbed, great care has been taken by each partner to validate independently their testbed HW and SW sub-parts. These validation steps are detailed in section 3: first the components are individually validated (§3.1) and then their correct integration into low layers are checked (§3.2). Finally the plan for the multi-connectivity testbed integration is described in 3.3.

Section 4 is dedicated to the description of the test scenarios, KPIs and results. Due to the VPN connection problem (see above), measurements do not include scenarios where multi-connectivity is involved.

Conclusions are draw in section 5.

2 Testbed architecture

The main objective of the 5G-ALLSTAR project is to implement advanced functionalities at both RAN and CN side for future hybrid 5G systems involving TNs as well as NTNs.

The Technical Concepts that will be developed, integrated and then demonstrated with the Testbed are:

- Advanced Radio Resource Management (RRM) functionalities for the multi-Radio Access Technologies (RAT) spectrum sharing scenario, where algorithms for interference analysis and mitigation will be developed to share the same spectral resource between a TN and a NTN. RRM functionalities are conceived to be in the Centralized RRM (cRRM) and Distributed RRM (dRRM) blocks.
- Traffic switching, steering and splitting. In fact, in the 5G-ALLSTAR project, the traffic management will be performed taking into account the network status and the Quality of Experience (QoE) requirements for each user.
- Optimized use of the spectral resource by implementation of Block-Filtered OFDM (BF-OFDM) transceivers on the terrestrial links.
- Adaptation of the 5G NR waveform to transparent GEO satellites specificities such as much longer transit times than within TNs.

To prove the relevancy and efficiency of these technological concepts for future 5G Networks the Testbed will emulate the reference system architecture depicted in Figure 1.

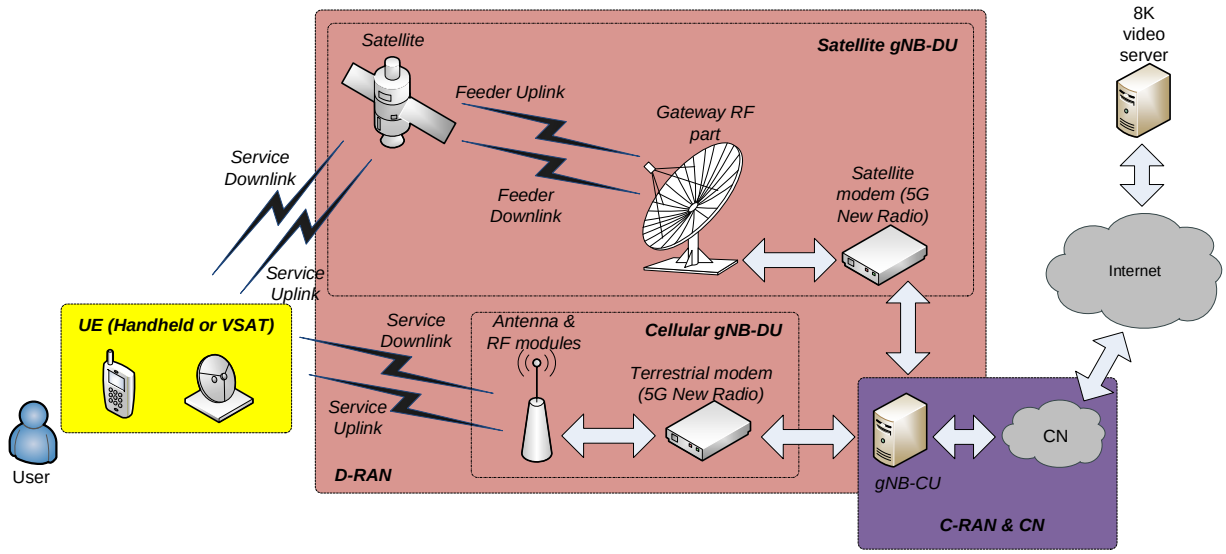


Figure 1. Reference system architecture

2.1 HW Architecture

The EU Testbed HW architecture is built up according to the reference architecture designed in WP2 and reported in D2.2 [1] and the specification delivered in D5.1 [2].

The architecture is made up by 4 main blocks (depicted in Figure 2):

- UE, consisting of
  - a machine running the terrestrial UE (TERR UE)
  - a machine running the satellite UE (SAT UE)

- a machine running the UE Network Flow Manager and the Video Apps
- the distributed RAN (dRAN), consisting of:
    - a machine running the Terrestrial gNB Distributed Unit (TERR gNB-DU) with integrated dRRM
    - a machine running the Satellite gNB Distributed Unit (SAT gNB-DU) with integrated dRRM
- the centralised RAN (cRAN), consisting of a machine running the gNB Centralised Unit (gNB-CU) and CN, including the following modules:
    - cRRM and Network Traffic Flow Control
    - Video Server
    - QoE controller
    - OpenBACH controller

The EU Testbed is physically distributed in split in two segments:

- the Satellite Platform (SAT Platform), located at FhG IIS premises (Erlangen, Germany)
- the core EU Testbed, located at CEA premises (Grenoble, France)

The pairs of TERR and SAT components making up the UE and the dRAN may have been virtually connected to the same testbed network by means of a VPN, see §2.2.1.1.7.

Figure 3 displays the HW components of the Testbeds in Erlangen and Grenoble, reflecting the architecture in Figure 2.

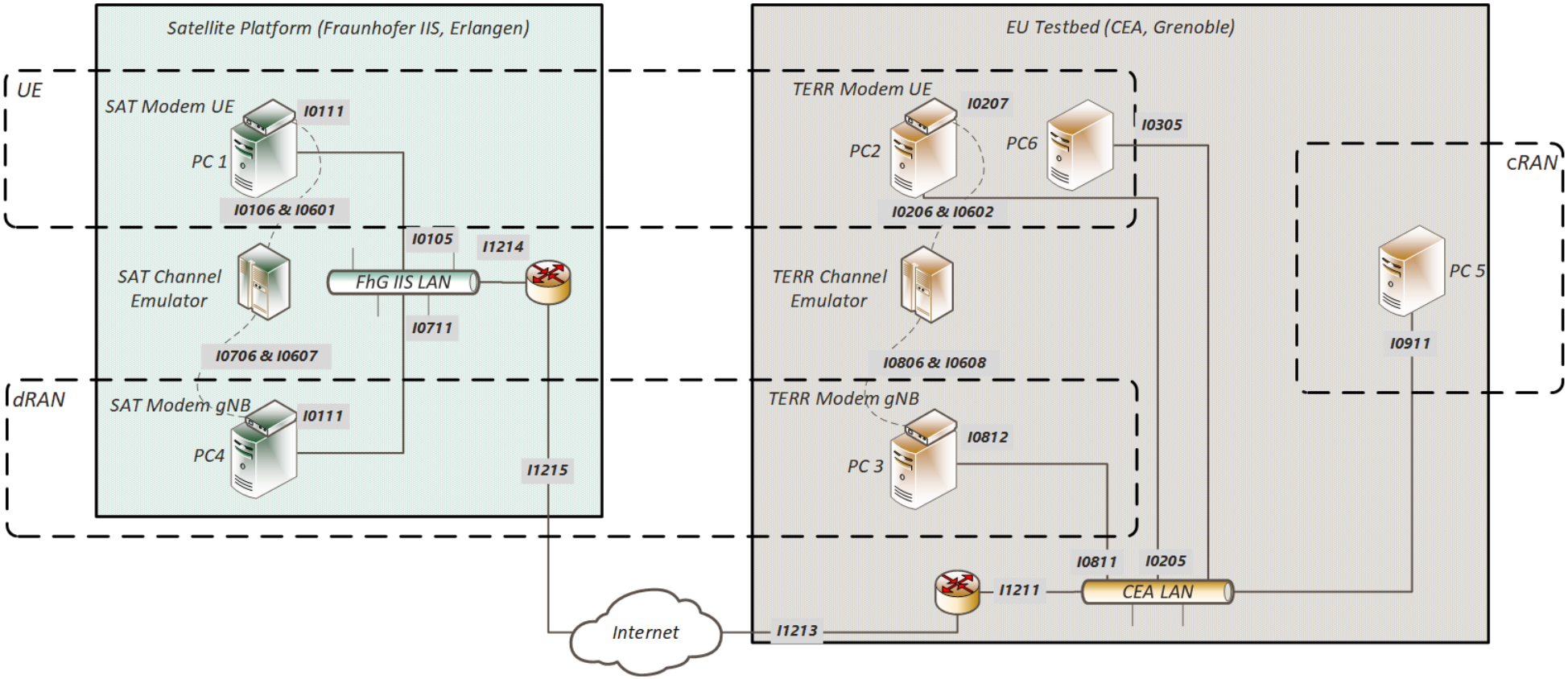


Figure 2. UE Testbed HW architecture with relevant interfaces

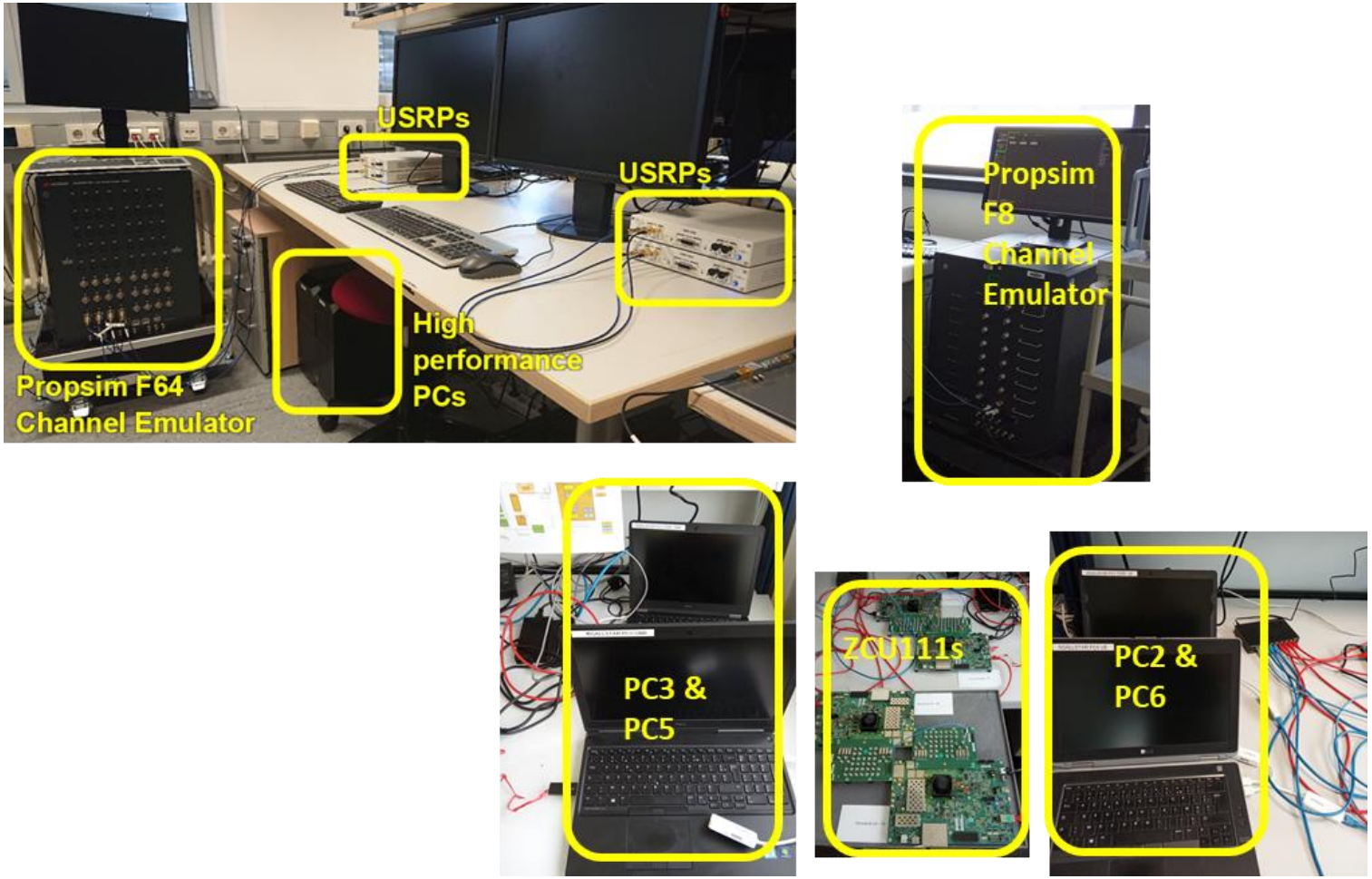


Figure 3. Pictures of the testbed implementation in both sites

2.1.1 Testbed HW Equipment

This section reports a detailed description of the HW deployed in the EU Testbed. The list of EU testbed components is reported in Table 1. The components distribution and location in the Testbed are reported in Figure 2.

Table 1. EU Testbed components list

Component name	Description	Providing partner
UE SAT modem	High-performance PC and Software Defined Radio (SDR) platform to implement satellite (SAT) friendly UE Tx/Rx L1 & MAC layers able to operate NR radio protocol in FDD mode via satellite channel (GEO) with implemented NR adaptations compliant to 3GPP TR 38.811 “NR support Non terrestrial networks”	FhG IIS
UE terrestrial modem	PC with Flex board to implement UE Tx/Rx L1 & MAC layers able to operate in FDD mode via terrestrial channel	CEA
UE core	PC hosting the VM emulating a 5G User Equipment (UE) with satellite/cellular multi-access capabilities. It interfaces with both satellite and terrestrial modems and hosts all the software components needed to emulate the UE in multi-access.	CEA
UE hub	Standard 10 Gbps Ethernet hub interconnecting all the PCs composing the emulated UE	CEA
SAT Channel Emulator	Satellite channel emulator able to emulate the long propagation delay of a GEO satellite channel.	FhG IIS
Terrestrial Channel Emulator	Terrestrial channel emulator.	CEA
SAT gNB-DU	High-performance PC and SDR platform to implement satellite-friendly gNB-DU Tx/Rx L1 & MAC layers able to operate NR radio protocol in FDD mode via satellite channel (GEO) with implemented NR adaptations compliant to 3GPP TR 38.811 “NR support Non terrestrial networks”.	FhG IIS
Terrestrial gNB-DU	PC with Flex board to implement terrestrial gNB-DU Tx/Rx L1 & MAC layers able to operate in FDD mode via terrestrial channel	CEA
cRAN: • gNB-CU & CN • Testbed C&M • 8K video server	PC (with screen, mouse and keyboard) emulating in a VM all the required functionalities of: • a 5G C-RAN and CN • Testbed C&M • 8K video server	CEA
RAN & CN hub	Standard 10 Gbps Ethernet hub interconnecting all the equipment composing the emulated C-RAN and CN	CEA
CN router	IP router (with firewall function) providing the Testbed with an access to the public data network (the Internet)	CEA

The specifics of the PCs mentioned in the HW list are detailed in Table 2.

Table 2. PCs deployed in the EU Testbed and most relevant specifics

Component ID	Component name	Description	Providing partner
PC1	Scope: UE • UE SAT modem	GPP: Intel Core i7-7820X 3.6 GHz, 8 cores • OS: Linux Ubuntu 4.15.0-70 low latency kernel • 32 GB RAM • NIC: 1000 Mb/s Ethernet card	FhG IIS
PC2	Scope: UE • UE terrestrial modem	i5 8 Go RAM	CEA
PC3	Scope: dRAN • gNB terrestrial modem	i5 8 Go RAM	CEA
PC4	Scope: dRAN • gNB SAT modem	GPP: Intel Core i7-7820X 3.6 GHz, 8 cores • OS: Linux Ubuntu 4.15.0-70 low latency kernel • 32 GB RAM • NIC: 1000 Mb/s Ethernet card	FhG IIS
PC5	Scope: cRAN • cRRM, Network Traffic Flow Controller, OpenBACH Controller (VM2) • Traffic flow control, video QoE control& video encoder/streamer (VM3) • Other application servers (critical app…)	Intel Core i7 2,7GHz, 16 Go RAM • Minimum requirements for VM: • GB RAM • CPU 1.8 GHz dual core • 30 GB available on Hard Drive	CEA
PC6	Scope: UE • UE traffic flow router • Network Traffic Flow Controller	VM Specifications: • vCPUs @ 3.2GHz (minimum, due to 8K video decoding) • 4GB RAM	CEA

2.1.1.1 Channel Emulators

The EU Testbed is equipped with 2 channel emulators, one for the TERR channel, located in France, and the other for the SAT channel, located in Germany.

2.1.1.1.1 SAT Channel Emulator

The 5G NR emulation platform at Fraunhofer IIS (FhG IIS) is capable to achieve a combined satellite/cellular file-based emulation of the test scenarios.

The channel emulator used in the emulation platform is the PROPSIM F64, engineered by Keysight Technologies (see [3]). It enables recreating the wireless channel propagation effects in a controlled laboratory environment and features:

- 5G RF channel modelling as defined in [4] through the Geometric Channel Modelling tool (GCM tool)
- Emulation of impairments produced by complex radio channel propagation effects, such as:

  - Dynamic multipath propagation
  - Pathloss, shadowing and fast fading
  - Doppler effect from mobility
  - Noise and synchronous programmable interference
- Antenna pattern embedding
- Scalable channel capacity (e.g. up to 64 MIMO channels)
- Wide bandwidth (e.g. up to 100 MHz)
- Bi-directional emulation
- Delay spread for terrestrial channel emulation up to 3 ms
- Multiple fading channels (fading, Doppler, pat amplitude and phase offset are independently configurable through GUI)
- Multiple fading paths per interface channel

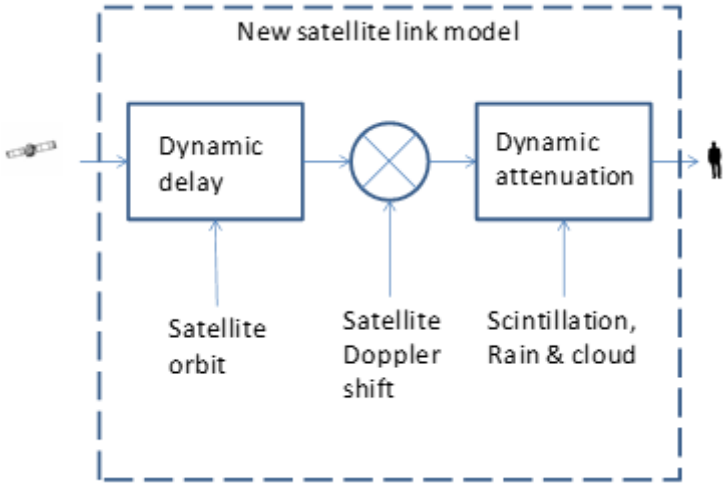


Figure 4. Satellite channel model (conceptual drawing derived from [5])

The F64 PROPSIM is equipped also with an Aerospace Option (ASO) and Aerospace and Satellite Modelling tool for testing airborne, aerospace and satellite radio communication devices and systems in order to emulate SISO topologies with high Doppler shift (up to +/- 1.5 MHz), long propagation delay spread (up to 1.3 s) and high range rates in a laboratory environment. The reference satellite link model for the emulation is illustrated in Figure 4. The most relevant specifics are detailed in Table 3.

The PROPSIM emulator scope is limited to the radio channel, thus excluding transmitters and receivers. Each channel unit is equipped with RF input/output duplex ports and output only RF connectors that allow interconnection with third-party devices to be tested (e.g. RF transceiver), irrespective of the system technology or modulation (as the PROPSIM supports all major wireless standards and waveforms).

The emulator is also equipped with external local oscillators and interfaces to laboratory hardware (i.e. DVI display port, USB for external I/O) that is used to monitor and control purposes.

Table 3 Specification of PROPSIM F64 channel emulator

Parameter	Specification
Delay	Propagation delay up to 1 s in ASO Min. HW delay is 2.5 µs
Max Instantaneous RF signal bandwidth	40 - 600 MHz

Input signal frequency range	450 MHz – 6GHz
Fading direction	Bi-directional and uni-directional fading supported
UL & DL separation	Integrated uplink (UL) and downlink (DL) separation
Dual mode emulation	Achievable. PROPSIM F64 supports the co-existence of terrestrial and ASO channels in single unit.
Fast fading profiles/channel models	ASO channel modelling is based on 3GPP 5G NR models of the PROPSIM Standard Tools software: Constant, Rayleigh, Rice, Nakagami, Lognormal, Suzuki, Pure Doppler, flat, rounded, Gaussian, Jakes, Butterworth, user-defined, and CIR data from 3rd party simulation tools Each digital channel can be set for independent fading profile (delay, Doppler, amplitude, correlation)

2.1.1.1.1.1 Aerospace Model Editor

The ASO is an additional layer in the PROPSIM that allows to model a satellite channel through the Aerospace Model Editor. The generated model file is eventually used in the definition of the channel emulation.

The satellite channel is configured according to the arbitrary model data type. The arbitrary model contains a point-to-point link between moving transmitter and receiver, where the link is assumed to be Line of Sight (LOS) and the user can use it to directly control the time series with delay, Doppler and gain values for transmitter, receiver and reflectors. The configuration windows of the Aerospace Model Editor are illustrated in Figure 5.

The model editor generates an ASO model file as an output. The following example shows the output file for an arbitrary model designed for a GEO satellite channel:

```
; Propsim Aerospace Model file, version 1.0


[Model]
SimulationCenterFrequency = 36e+008 Hz
RFCenterFrequency = 36e+008 Hz


[LOS]
A = 0, 0.2, 0, 0
A = 60, 0.2, 0, 0
```

Custom models are available and can be directly open and edited in a text editor. Further details about setup and configuration in §3.2.1.

Figure 5. Arbitrary model configuration window

For user applications with frequency band outside PROPSIM frequency range, the ASO offers the possibility to set different frequencies for the model and for the emulation. Doppler is always calculated by using emulation centre frequency value. RF centre frequency defines the actual frequency used in connections to and from the PROPSIM emulator.

2.1.1.1.2 TERR Channel Emulator

The channel emulator used for the emulation of terrestrial DL and UL channels is the EB Propsim F8, engineered by Keysight Technologies. It is the same kind of channel emulator as the satellite component, even though the product model is different. The main features of interest for the EU Testbed are reported in Table 4.

Table 4. Specification of PROPSIM F8 channel emulator

Feature	Working range
RF interface channel frequency range	220 to 6000 MHz
RF interface channel signal bandwidth	up to 160 MHz
Number of fading paths per RF interface channel (in terrestrial channel emulation mode)	up to 48
Number of fading channels. All independently controllable via GUI for fading, Doppler, path amplitude and path phase offset	up to 64
Internal interference generators	AWGN, CW
Excess delay range for terrestrial channel emulation	up to 3000 µs
Bi-directional emulation	Flexible and reliable duplex separation provided with interfacing unit
Number of integrated RF local oscillators	up to 4 internal and 4 external carrier frequencies (in total up to 8)
Input power measurement	Automatic input level setting
Standard channel models	3GPP LTE, WCDMA, GSM, 3GPP2 (IS-54, IS 95), TETRA, ITU 3G, WLAN, DVB-T/H

Fading profiles	Constant, Rayleigh, Rice, Nakagami, Lognormal, Suzuki, Pure Doppler, flat, rounded, Gaussian, Jakes, Butterworth, user-defined profiles, models from 3rd party simulation tools and ray-tracing applications
Delay profiles	Constant, sinusoidal sliding delay, linear sliding delay, 3GPP birth-death, 3GPP sliding delay group, user-defined, delay profiles from 3rd party simulation tools and ray-tracing applications
Run-time fading engine	Amplitude, delay, Doppler and environment separately controlled for each fading channel

2.1.1.2 USRP X300

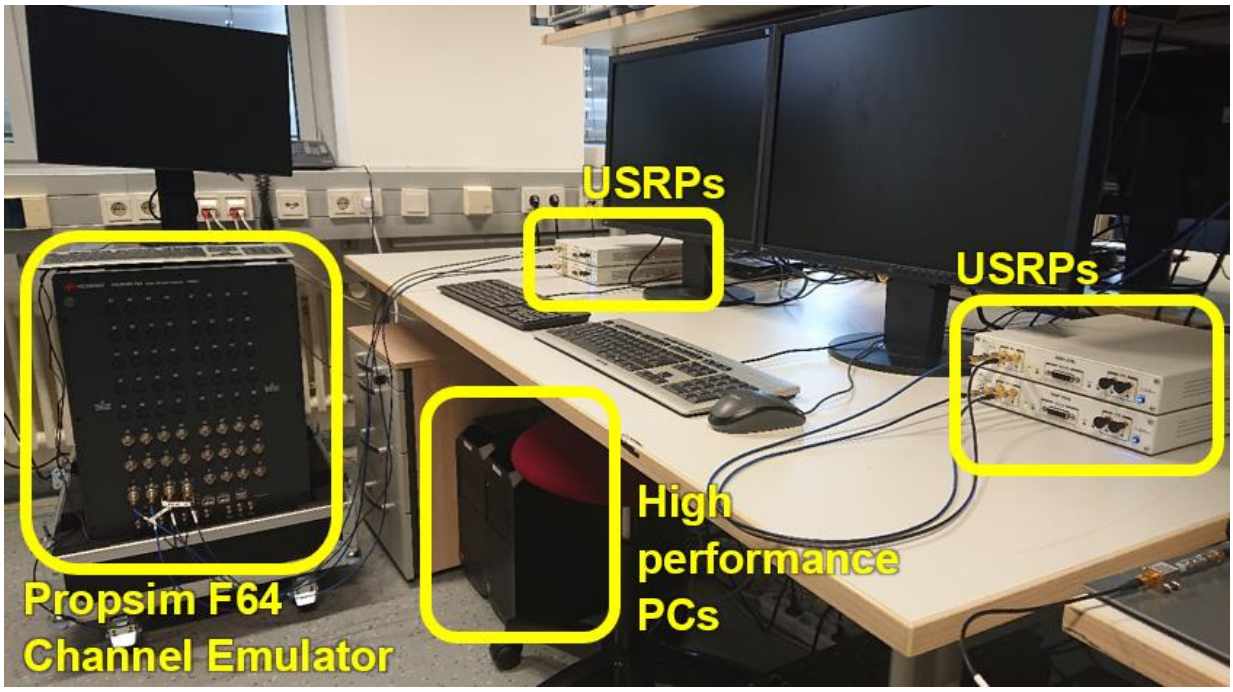


Figure 6. Lab setup of the IIS Emulation Platform

The SAT modems are running on same HW for both EU and gNB, based on the SDR platform Ettus USRP X300, equipped with:

- 1 CBX-120 daughterboard
- x 50 dB attenuators

And providing the following features:

- RF I/O ports („baseband”):
  - Frequency range: tuneable 1200 MHz - 6000 MHz
  - Channels bandwidth available: up to 120 MHz (OpenAirInterface currently supports up to 40 MHz)
  - TDD and FDD operation modes

The interface with the PC is a PCI express (PCIe). The lab setup of the SAT emulation platform at FhG IIS is shown in Figure 6.

2.1.1.3 ZCU111 boards

The PHY of the terrestrial modem (transmitter and receiver) is implemented on Xilinx ZCU111 boards, see §3.1.1.1. The architecture of the board is provided in Figure 7. It schematically consists of:

- an RFSoC (ZU28DR)
- a Programmable Logic Memory (SDRAM)
- connectors

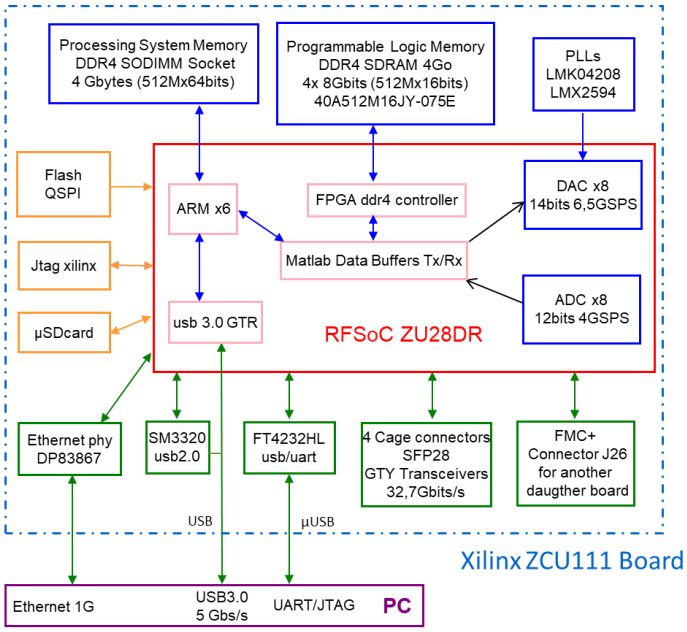


Figure 7. Xilinx ZCU11 architecture

The features of the RFSoc ZUCU28DR are reported in Table 5.

Table 5. Zynq UltraScale+ RFSoc feature summary

	XCZU21DR	XCZU25DR	XCZU27DR	XCZU28DR	XCZU29DR
12-bit, 4.096GSPS RF-ADC w/ DDC	0	8	8	8	0
12-bit, 2.058GSPS RF-ADC w/ DDC	0	0	0	0	16
14-bit, 6.554GSPS RF-DAC w/ DUC	0	8	8	8	16
SD-FEC	8	0	0	8	0
Application Processing Unit	Quad-core ARM Cortex-A53 MPCore with CoreSight™; NEON and Single/Double Precision Floating Point; 32KB/32KB L1 Cache, 1MB L2 Cache				
Real-Time Processing Unit	Dual-core ARM Cortex-R5 with CoreSight; Single/Double Precision Floating Point; 32KB/32KB L1 Cache, and TCM				
Embedded and External Memory	256KB On-Chip Memory w/ECC; External DDR4; DDR3; DDR3L; LPDDR4; LPDDR3; External Quad-SPI; NAND; eMMC				
General Connectivity	214 PS I/O; UART; CAN; USB 2.0; I2C; SPI; 32b GPIO; Real Time Clock; Watchdog Timers; Triple Timer Counters				
High-Speed Connectivity	4 PS-GTR; PCIe® Gen1/2; Serial ATA 3.1; DisplayPort 1.2a; USB 3.0; SGMII				
System Logic Cells	930,300	678,318	930,300	930,300	930,300
CLB Flip-Flops	850,560	620,176	850,560	850,560	850,560
CLB LUTs	425,280	310,088	425,280	425,280	425,280
Distributed RAM (Mb)	13.0	9.6	13.0	13.0	13.0
Block RAM Blocks	1,080	792	1,080	1,080	1,080
Block RAM (Mb)	38.0	27.8	38.0	38.0	38.0
UltraRAM Blocks	80	48	80	80	80
UltraRAM (Mb)	22.5	13.5	22.5	22.5	22.5
DSP Slices	4,272	3,145	4,272	4,272	4,272
CMTs	8	6	8	8	8
Maximum HP I/O	208	299	299	299	312
Maximum HD I/O	72	48	48	48	96
System Monitor	1	1	1	1	1
GTY Transceivers	16	8	16	16	16
Transceivers Fractional PLLs	8	4	8	8	8
PCIe Gen3 x16 and Gen4 x8	2	1	2	2	2
150G Interlaken	1	1	1	1	1
100G Ethernet w/ RS-FEC	2	1	2	2	2

ZCU111

Figure 8 shows the gNB and UE boards plugged to the TERR Testbed.

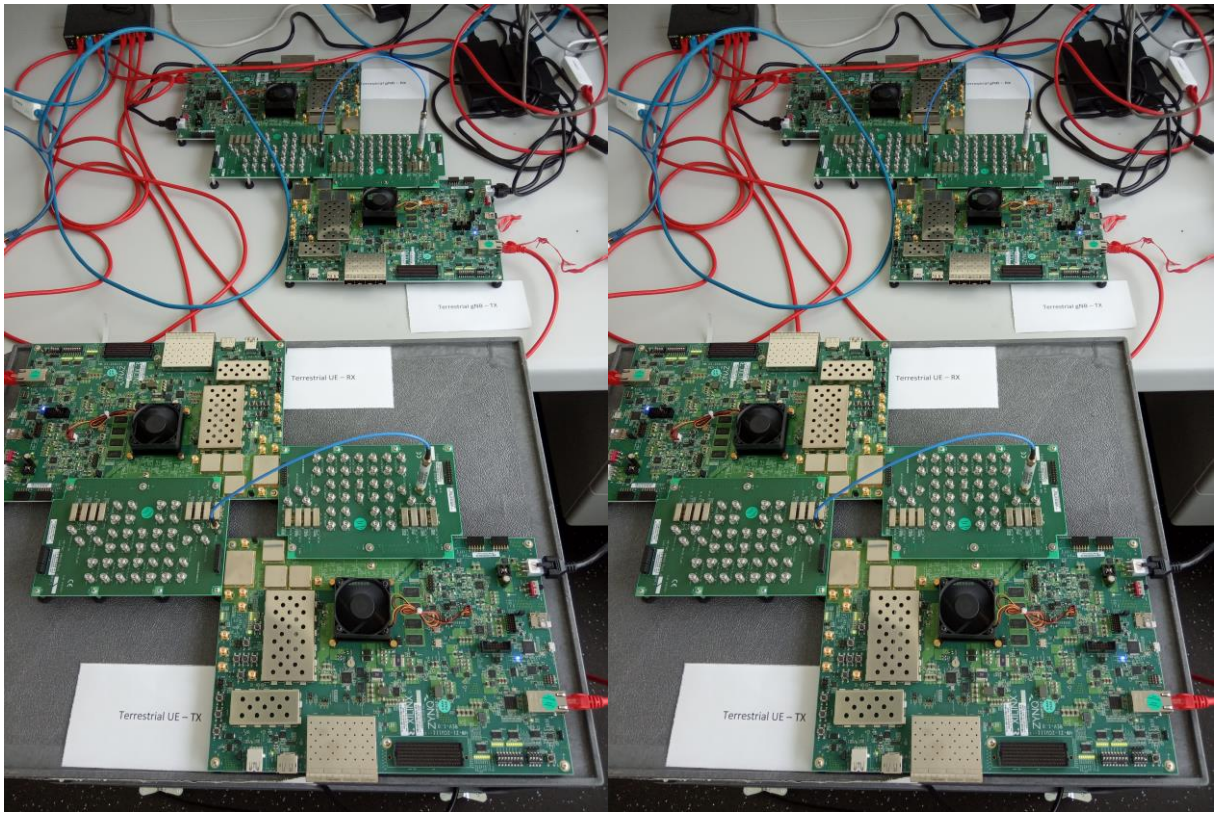


Figure 8. ZCU111 TX and RX for UL and DL.

2.1.1.4 8K TV

In the implementation of the final demonstration the 8K screen has been removed, since there is no added value to actually see the video that has been played. Indeed, the quality level of the video played at UE side is constantly measured (with a value ranging from 1 to 6, where 6 corresponds to 8K video stream) and reported through OpenBACH APIs by the video server at CN side even if it the video stream is not actually showed on a real screen by the video client. In order to let the video stream reach 8K resolution also without an actual screen, at UE side the video client simulates a screen size of 7680×4320.

The choice to remove the 8K screen has been made since the hardware requirements needed to play an 8K video are very high (e.g., in terms of GPU, but also CPU cores to process the 8K video frames), so a high-performance PC would be needed. Instead, removing the 8K screen it is possible to still validate the proposed multi-connectivity and video streaming components, together with all the other functional components of the demonstrator, using fewer hardware resources, using measures of quantitative KPIs from the video server and the video client.

2.1.2 HW interfaces

This section describes the interfaces between the testbed components introduced in the previous chapters. Table 6, previously introduced in [1], has been updated and changes are here reported with reference to Figure 2.

Table 6. Updated EU Testbed physical interfaces list

Interface label	From	To	Type	Description
I0607	SAT Channel emulator	SAT gNB-DU	RF coaxial connector	Interface between the Gateway RF reception chain (RF-to-IF down-converter output) and the gNB-DU SAT modem Rx input
I0706	SAT gNB-DU	SAT Channel emulator	RF coaxial connector	Interface between the Gateway RF transmission chain (IF-to-RF up-converter input) and the gNB-DU SAT modem Tx output
I0608	TERR Channel emulator	TERR gNB-DU	RF coaxial connector	Interface between the Terrestrial gNB RF reception chain (RF-to-IF down-converter output) and the gNB-DU TERR modem Rx input
I0806	TERR gNB-DU	TERR Channel emulator	RF coaxial connector	Interface between the terrestrial gNB RF transmission chain (IF-to-RF up-converter input) and the gNB-DU TERR modem Tx output
I0601	SAT Channel emulator	UE SAT modem	SMA RF Connector	Interface between the UE satellite RF reception chain (RF-to-IF down-converter output) and the EU SAT modem Rx input
I0106	UE SAT modem	SAT Channel emulator	SMA RF Connector	Interface between the EU satellite RF transmission chain (IF-to-RF up-converter input) and the EU SAT modem Tx output
I0602	TERR channel emulator	UE Terrestrial modem	RF coaxial connector	Interface between the EU terrestrial RF reception chain (RF-to-IF down-converter output) and the EU TERR modem Rx input
I0206	UE TERR modem	TERR channel emulator	RF coaxial connector	Interface between the EU terrestrial RF transmission chain (IF-to-RF up-converter input) and the EU TERR modem Tx output
I0105	PC1 (UE SAT)	IIS LAN	Internal Duplex Ethernet	Emulates UE sub-components internal interface
I0106	UE SAT Modem	PC1 (UE SAT)	PCIe	Interface between the PC and the SAT UE modem
I0205	PC2 (UE TERR modem)	CEA LAN	Internal Duplex Ethernet	UE sub-components internal interface
I0207	UE TERR modem	PC2 (UE TERR modem)	Internal Duplex Ethernet	Interface between the PC and the TERR UE modem
I0305	PC6 (UE core)	CEA LAN	Internal Duplex Ethernet	UE sub-components internal interface
I0711	PC4 (SAT gNB-DU)	RAN & CN hub	Internal Duplex Ethernet	Interface between satellite gNB-DU and gNB-CU
I0811	PC3 (TERR gNB-DU)	RAN & CN hub	Internal Duplex Ethernet	Interface between terrestrial gNB-DU and gNB-CU

Interface label	From	To	Type	Description
I0812	TERR gNB-DU Modem	PC3 (TERR gNB-DU)	Internal Duplex Ethernet	Interface between the PC and the TERR gNB modem
I0911	PC5 (gNB-CU & CN)	CEA LAN	Internal Duplex Ethernet	gNB-CU interfaces with gNB-DUs and CN/ Testbed Command & Monitoring
I1011	SAT gNB-DU Modem	PC4 (SAT gNB-DU)	PCIe	Interface between the PC and the SAT gNB-DU modem
I1211	CN router	CEA LAN	Internal Duplex Ethernet	Internal (user data) interface of the EU Testbed LAN with its Web/Internet router (for 8K video server interfacing)
I1213	CN router	Web/Inter-net	External Duplex	Web/Internet access point of the EU Core Testbed and interface with the IIS SAT Platform (for 8K video server interfacing)
I1214	IIS LAN router	SAT Plat-form	Internal Duplex Ethernet	Interface between the IIS SAT Platform LAN and its Web/Internet router
I1215	IIS CN router	Web/Inter-net	External Duplex	Internet access point and interface between the IIS SAT Platform and the Core EU Testbed

RF interfaces are detailed in Annex 7.3.

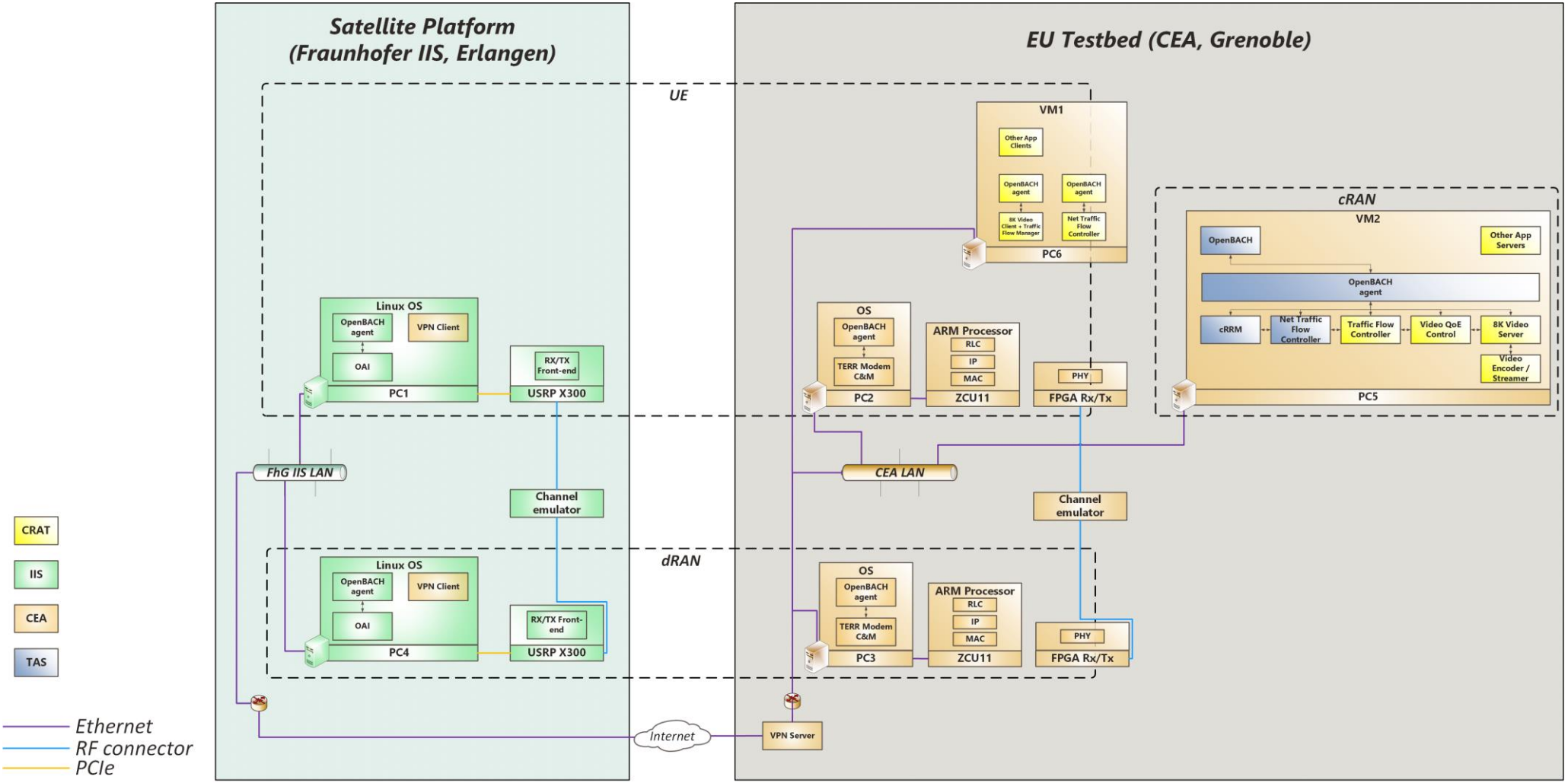


Figure 9. SW architecture of the Testbed

2.2 SW Architecture

The SW Architecture of the EU Testbed is illustrated in Figure 9.

2.2.1.1 Architecture of SW modules

2.2.1.1.1 cRRM

2.2.1.1.1.1 Overall presentation

cRRM (Central Radio Resource Management) is a central component in the 5G-ALLSTAR approach that ensures a safe coexistence of both heterogeneous RATs cellular (TERR) and satellite (SAT) in terms of **Dynamic spectrum sharing** and interference prevention.

It implements advanced strategies to dynamically select the appropriate radio configuration depending on the actual conditions. Radio measurements made by the UEs and gNodes (gNBs) are provided by the dRRM (distributed Radio Resource Management) software components serving to connect each gNB-DU to the cRRM located in the gNB-CU and to exchange control messages.

2.2.1.1.1.2 Satellite from/to Terrestrial Handover emulation

Note that complete testbed does not conform to 3GPP / 5G control plane. The Handover (HO) process must somewhat be simplified. For practical implementation reasons, the UE modems always maintain a minimal control link with its two gNB-DUs.

The HO management is supported by means of:

1) traffic management – with support of Network or Video Traffic Flow Controller (for which available throughput are provided). This is supported with Traffic switching
2) coordinated /synchronized application of the Channels, emulating the UE movement. It is up to the testbed *operator* to ensure he provides consistent channel models / channel dynamics and relevant policies. The testbed itself only makes decision based on SNR received from the two links.

2.2.1.1.1.3 cRRM interfaces

cRMM acts in the control plane only and partly in management plane (monitoring and configuration). cRRM has interfaces with :

- dRRMs (hosted in modems) ;
- Network Traffic Flow Controller;
- Applicative Traffic Flow Control
- OpenBACH controller

The purposes of these interfaces are as follows:

1. Interface with dRRM (modems):

- allows to periodic receive SINR measurement reports from modems/dRRMs.
- allows to receive instantaneous notification of the modem/dRRMs in case link status goes completely down or up (link resumed).
- allows to send the applicable sub-band sharing (under the form of RB allocations) within the shared frequency channel, and the MCS (ModCod schemes) for each link, in real-time

2. Interface with (Video application) traffic flow controller

- sends periodically the AVAILABLE throughput capacity for each satellite and terrestrial links.

3. Interface with Network traffic flow controller

- sends periodically the AVAILABLE throughput capacity for each satellite and terrestrial links.

NB: see a further section to see how the two traffic flow controller can work together.

4. Further, for analysis and monitoring purposes, cRRM can send some predefined KPIs (Key Performance Indicators) to OpenBACH Collector which saves them in a time-series database and also allows to visualize them in real-time/offline from a user-friendly dashboard. Therefore the machine hosting the cRRM shall run an OpenBACH Agent for enabling OpenBACH communication.

2.2.1.1.1.4 Constraints

The cRRM plays a central role in the 5GALLSTAR approach. Given the difference with the cRRM described for the full system in (see [6]), we choose to implement it in an adapted form.

We select some strategy policies with respect to demonstration and other testbed prototyping constraints, as follows:

- Firstly the demonstrator and its hardware does not support the notion of 3GPP cells.
- Secondly, and mainly due to the difference in delays between terrestrial links and GEO, a fast-loop PRB allocation process is not achievable.
- Thirdly, the demonstrator is limited to a single terminal (UE)

When both links are used together, cRRM and each modems are configured to work consistently over a shared channel **with pre-configured parameters** (see §2.2.1.2.1).

In the following we assume **a model** where resources are (statically of dynamically) shared in fractions of the channel:

- one is allocated to satellite
- one is allocated to terrestrial

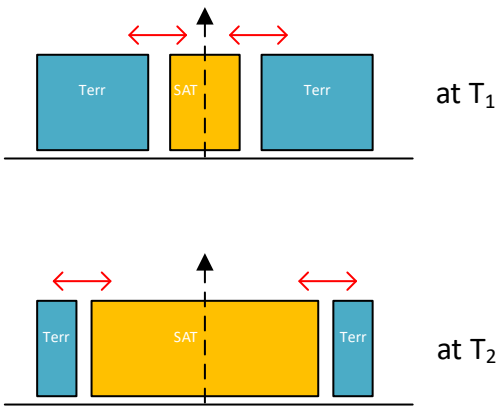
Hence once allocation to sub-band is known, each modem can run independently its local short-term MAC scheduling.

The splitting of this total band constitutes one output of the emulator cRRM. Ideally, this sharing would have resulted in 2 contiguous sub-band, with a moving frontier.

This scheme was not possible for the satellite modem because it requires a fixed central frequency.

We therefore implemented the dynamic sharing in **3 equivalent sub-bands:**

- 1 central sub-band at centre of the channel (→ satellite resources)
- 1 lower-side and 1 upper-side sub-bands (→ terrestrial resources).



The sub-band are expressed in groups of RBs (number of RBs). Traffic allocation to the specific subcarriers is then done by dRRM/modems, transparently to the cRRM.

For simplification, the band sharing and MCS decision are done identical for each forward or return direction of the satellite channel (resp. downlink/uplink for the terrestrial channel).

Note that the extremity sub-bands (lower and upper borders of the channel) must be configured as guard sub-bands and will not allow to allocate user traffic.

Finally, in order to maintain its link operations (radio signalling reception), the satellite modem shall never receive an allocation with RB=0. This constrains the maximal (equivalent) sub-band provided to the terrestrial modem that can never fully be allocated the full number of RB within the channel.

Note that the terrestrial modem supports this band modification but it requires to restart the modems and therefore to stop the traffic.

2.2.1.1.1.5 cRRM Algorithms

MCS Management

cRRM supports both modes:

- ➔ Constant Coding & Modulation mode (each MCS or MODCOD or each link being configured statically, whatever the SINR reports are, from the modems)
- ➔ Adaptive Mode (ACM), so that higher throughput can be delivered when conditions are favourable. A margin may be configured so that additional SNR margin, if desired, can protect signal quality and QoS from quick variations of the channel. For example imagine a MCS requires 4.5 dB (under assumption of a given target BLER). If the margin is 1.0 dB the MCS will be available for allocation only when SNR>5.5 dB.

The cRRM has a configuration table parameter indicating for each MCS what is the achievable throughput, for a given amount of sub-band. Table 7 and Table 8 can be theoretically derived from 3GPP tables of MCS/Coding rate vs Spectral Efficiency, however, in the scope of this project, they have been actually calibrated and measured for safer operations, for each prototype link modem.

Table 7 shows the performance of the terrestrial modem and the minimum CNR required for each MCS to reach a Transport Block Error Rate of 10^{-2}. Note that depending on the bandwidth configuration, some MCS cannot be used, see §3.1.1.2.

Table 7. MCS for Terrestrial modem link

Modulation	MCS #	Coding rate (x1024)	Spectral efficiency (bps/Hz)	Min. required CNR (dB)
QPSK	2	308	0.457	1.1
QPSK	3	449	0.666	2.8
QPSK	4	602	0.892	5.1
16-QAM	5	378	1.121	7.4
16-QAM	7	490	1.453	9.5
16-QAM	9	616	1.826	11.4
64-QAM	11	466	2.072	13.3
64-QAM	13	567	2.521	15.3

64-QAM	15	666	2.962	17.2
64-QAM	17	772	3.433	19.3
64-QAM	19	873	3.882	21.2
256-QAM	21	711	4.216	22.8
256-QAM	23	797	4.726	24.9
256-QAM	25	885	5.248	27.4
256-QAM	27	948	5.621	29.3

Table 8 shows the performance of the satellite-friendly modems at the L1 with a target BLER of 0.01 w.r.t the MCS table 5.1.3.1-1 of [7]. The spectral efficiency is computed taking into account a 20% overhead.

Table 8. MCS for satellite-friendly NR modems

Modulation	Coding rate (x1024)	Spectral efficiency (bps/Hz)	Min. required CNR (dB)
QPSK	120	0.18	1.18
QPSK	157	0.24	1.82
QPSK	193	0.30	2.66
QPSK	251	0.39	3.50
QPSK	308	0.48	3.98
QPSK	379	0.58	4.89
QPSK	449	0.70	5.71
QPSK	526	0.82	6.51
QPSK	602	0.93	7.45
QPSK	679	1.05	8.08
16-QAM	340	1.05	9.20
16-QAM	378	1.17	9.83
16-QAM	434	1.34	10.57
16-QAM	490	1.51	11.75
16-QAM	553	1.71	12.68
16-QAM	616	1.90	13.36
16-QAM	658	2.03	14.19

Link Throughput reports

Given the known MCS, the corresponding spectral efficiency can be derived. The throughput report done to Video and/or Generic traffic flow router just has to take the fraction of band from this throughput, with MCS allocated to the satellite or terrestrial link. A default period of 1 second is configured for the throughput report.

Bandwidth sharing profile

The bandwidth sharing is a profile to be configured and/or added in CRRM. As previously explained this would be the result of a full radio environment (multiple UEs, SINR reports not modelled in cRRM).

A single UE case cannot by itself be representative of that. Instead we have chosen to demonstrate the sharing concept was working for any dynamic profiles; these are inputs for cRRM

We propose some test profiles for the functional validation of the scheme. Any other could be added.

For reference we have worked with:

- Static profile: the allocation is set to X% of the channel band for satellite, 100-X% for the terrestrial
- Periodic "on/off" time series describing the output of the cRRM allocation (see next figure, left part).
    - This demonstrate maximal amplitude variation change of the band.
    - Triangular" test profile (right part of the figure) where the variation is progressive. The period and amplitude steps are parameters.
- SNR-based profile: the allocated bandwidth is depending on the received SNR

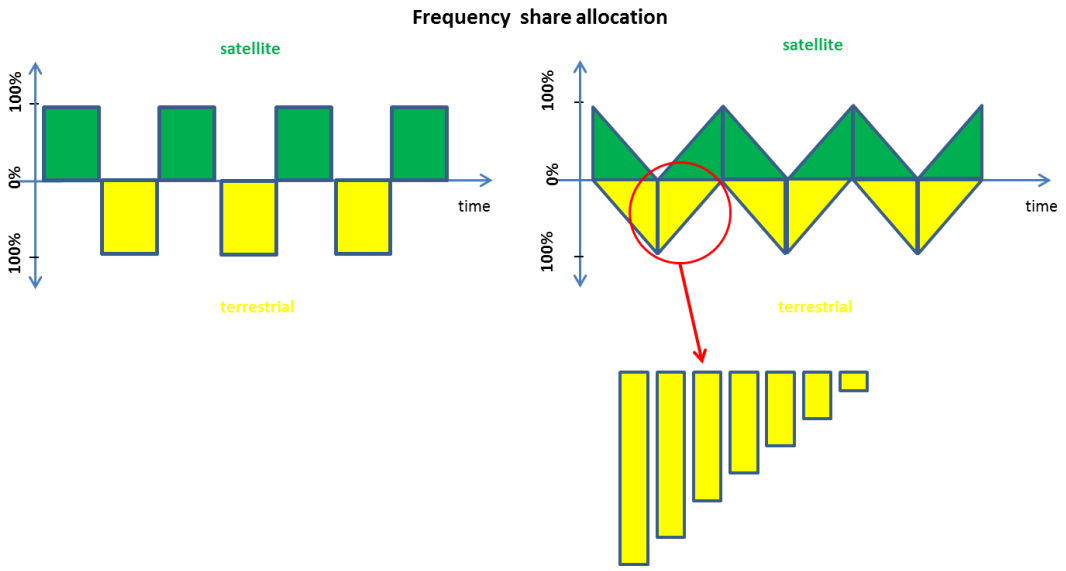


Figure 10. Example of output of the cRRM to be tested

Those profile could pave the way for enhancement models that would also use SNR measurements reports:

- "on-off" model based on best link signal: instead of being periodic as in on/off profile), the cRRM allocates the full band to the link with best SNR (or : best MCS)
- proportional model: fraction of the band is allocated to reflect the fraction of total available SNR for UE such that:
    - Band Fraction $_i$ = SNR$_i$ / (SNR $_i$ + SNR $_{j \neq i}$)
    - with i={terrestrial; satellite}, and SNR the measured SNR (number expressed in natural representation)

Many other models could be further defined.

2.2.1.1.2 Network Traffic Flow Controller

2.2.1.1.2.1 Concepts and algorithms

The Network Traffic Flow Controller is a software component able to route, and according to custom policies /rules, the traffic data (IP packets) it receives of TCP or UDP - IP ports, to its two out ports, one to be connected to the terrestrial link and the other to the satellite link. By configuring relevant routing policies, it can support traffic flow steering, splitting, switching.

The same component is used in 2 instances for UE and gNB sides.

The traffic policy followed by the Network Traffic Flow Controller is defined by series of rules contained in dedicated configuration file.

The Network Traffic Flow Controller can be run in several modes:

- Static: the configuration fully defines by configuration which traffic are forwarded to which link regardless of cRRM indication. All traffic may or may not use the same output interface upon operator preference. This mode supports steering**.**
- Static with load balancing: still static, the input traffic balancing to the output ports are fixed (X% of the receive traffic to “Satellite” port and (100 – X%) to the “Terrestrial” port. This is a form of steering+splitting.
- Dynamic polices:
    - These policies will take into account the indication of the offered throughput as provided by cRRM. Either rule-based or more complex control laws, as described in document D4.3, can be deployed. The traffic rules are preconfigured so as to define some preferences (for example if a traffic prefers low delay (terrestrial) or satellite (for any reason). The allocation works in strict (ordered) priority as defined by configuration. According to available resources, the Router tries to firstly satisfy preference or Application Flow (AF) #1, which has the top-level priority, then #2, and so on. When no more resource is available the AF #x is forwarded to its non- preferred interface. An indication of expected AF input data rate could be provided in configuration to ease the process. Otherwise the SW router could be extended so as to measure incoming throughput of each AF.

NB: A side mode can enable **a policy overrule** when one link has been reported as completely failing. This triggers an immediate reaction (**switching**) in order to minimize packet losses at each unpredictable link loss event so that all traffic is routed to the sole working radio link.

Policy illustration examples are provided below:

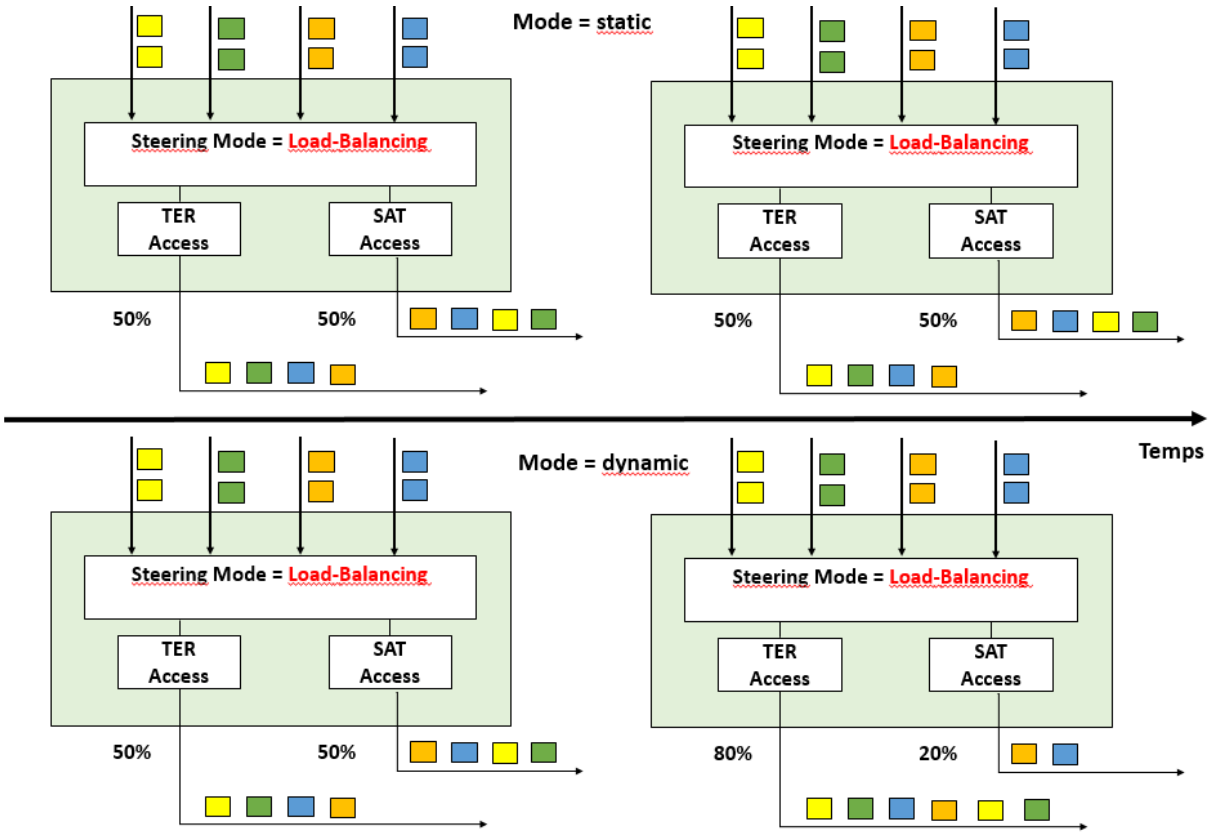


Figure 11. Policy illustration examples

This only shows few possible example of uses of the controller. Many more advanced policies could be extended from what has been developed.

The applicable policy are defined by means of rules, corresponding to traffic profiles. A given profile is identified as:

- IP addresses/subnets (source and/or destination)
- Port number or ranges

Other tags (e.g. IP DSCP…) are currently not used in traffic profiles.

Note:

When (Video Applicative) Traffic Flow Controller is used, it will take the responsibility of the flow sharing (and based on real-time throughout information received from cRRM). In that case a default (or “transparent”) mode shall be configured at network Flow controller for that application/rule, so that the decision done at applicative level is not overruled by the network flow router.

The policies are distributed at start-up of cRRM.

2.2.1.1.2.2 Software Design

The controller handles traffic by means of configuring Linux routing rules via a python script implementing iptables interface commands.

Note that it is very sensitive to the network interface environment.

2.2.1.1.3 OAI SAT-Friendly modems

OpenAirInterface (OAI) is an implementation of the 3GPP mobile communication standards. It currently supports LTE UE, eNB and EPC. Development on the 5G NR standards (beginning with 3GPP release 15) is ongoing.

The companies contributing to the OAI software are organized within the OpenAirInterface Software Alliance (OSA, [8]).

The baseline feature set of the OAI modems is reported in Table 9.

Table 9. OAI features set

Scope	Description
General Parameters	The following features are valid for the gNB and the 5G-NR UE. • TDD/FDD • Normal CP • 30 kHz subcarrier spacing • Bandwidth: 40, 80, 100 MHz • All FR1 5G NR bands are supported • Single antenna port (single beam) • Slot format: 14 OFDM symbols in UL or DL • Highly efficient 3GPP compliant LDPC encoder and decoder (BG1 and BG2 supported) • Highly efficient 3GPP compliant polar encoder and decoder • Encoder and decoder for short blocks • Non-Standalone (NSA) running mode: NR configuration coming through a virtual LTE cell coded in the configuration file. Upper layers (i.e. PDCP) are still LTE standard compliant.
gNB PHY Layer	• 30KHz SCS for FR1 and 120 KHz SCS for FR2 • Generation of NR-PSS/NR-SSS • NR-PBCH supports multiple SSBs and flexible periodicity • Generation of NR-PDCCH for SIB1 (including generation of DCI, polar encoding, scrambling, modulation, RB mapping, etc) • common search space configured by MIB • user-specific search space configured by RRC • DCI formats: 00, 10 • Generation of NR-PDSCH (including Segmentation, LDPC encoding, rate matching, scrambling, modulation, RB mapping, etc). • Single symbol DMRS, DMRS-TypeA-Position Pos2, DMRS configuration type 1 • PDSCH mapping type A • NR-PUSCH (including Segmentation, LDPC encoding, rate matching, scrambling, modulation, RB mapping, etc). • NR-PUCCH • Format 0 (2 bits, mainly for ACK/NACK) • Format 2 (up to 64 bits, mainly for CSI feedback) • NR-PRACH • Formats 0,1,2,3, A1-A3, B1-B3 • Highly efficient 3GPP compliant LDPC encoder and decoder (BG1 and BG2 are supported) • Highly efficient 3GPP compliant polar encoder and decoder • Encoder and decoder for short block
gNB PDCP Layer	In the noS1 mode a TUN interface to PDCP is created to inject and receive user-place traffic.

	The current PDCP layer is header compliant with 3GPP 36.323 Rel 10.1.0 and implements the following functions: • User and control data transfer • Sequence number management • RB association with PDCP entity • PDCP entity association with one or two RLC entities • Integrity check and encryption using the AES and Snow3G algorithms
gNB RRC	• NR RRC (38.331) Rel 15 messages using new asn1c • LTE RRC (36.331) also updated to Rel 15 • Generation of CellGroupConfig (for eNB) and MIB • Application to read configuration file and program gNB RRC • RRC can configure PDCP, RLC, MAC
gNB X2AP	• X2 setup with eNB • Handling of SgNB Addition Request / Addition Request Acknowledge / Reconfiguration Complete
gNB MAC	• MAC -> PHY configuration using NR FAPI P5 interface • MAC <-> PHY data interface using FAPI P7 interface for BCH PDU, DCI PDU, PDSCH PDU • Scheduler for RA procedure gNB • MAC downlink scheduler (fixed allocations) • MAC header generation (including timing advance) • ACK / NACK handling and HARQ procedures for downlink • As of May 2020 only DL was validated with COTS phone ; UL in progress, validated with OAI UE in noS1 mode

UE PHY Layer	• Initial synchronization
	• Time tracking based on PBCH DMRS
	• Time tracking based on PBCH DMRS
	• Frequency offset estimation
	• PBCH RX
	• PDCCH RX
	• PDSCH RX, including a first version of dual stream receiver for PDSCH
	• 30KHz SCS for FR1 and 120 KHz SCS for FR2
	• Generation of NR-PSS/NR-SSS
	• NR-PBCH supports multiple SSBs and flexible periodicity
	• Generation of NR-PDCCH for SIB1 (including generation of DCI, polar encoding, scrambling, modulation, RB mapping, etc)
	  ○ common search space configured by MIB
	  ○ user-specific search space configured by RRC
	  ○ DCI formats: 00, 10
	• Generation of NR-PDSCH (including Segmentation, LDPC encoding, rate matching, scrambling, modulation, RB mapping, etc).
	  ○ Single symbol DMRS, DMRS-TypeA-Position Pos2, DMRS configuration type 1
	  ○ PDSCH mapping type A
	• NR-PUSCH (including Segmentation, LDPC encoding, rate matching, scrambling, modulation, RB mapping, etc).
	• NR-PUCCH
	  ○ Format 0 (2 bits, mainly for ACK/NACK)
	  ○ Format 2 (up to 64 bits, mainly for CSI feedback)
	• NR-PRACH
	  ○ Formats 0,1,2,3, A1-A3, B1-B3

	• Highly efficient 3GPP compliant LDPC encoder and decoder (BG1 and BG2 are supported) • Highly efficient 3GPP compliant polar encoder and decoder • Encoder and decoder for short block
UE MAC	• Initial sync and MIB detection • MAC -> PHY configuration of PHY via UE FAPI P5 interface • Basic MAC to control PHY via UE FAPI P7 interface • Random access (RA) procedure

Please refer to Annex 7.2 for operational instructions of the SAT Modems.

The following features have been implemented to adapt the NR procedures of OAI to the long delay scenarios, such as GEO satellite channels. These features are reflecting the NTN solutions described in [9].

Table 10. Adaptations implemented to the OAI modems to support NTN

Scope	Description
Extension of k2	Additional *k2* should be configured via the command-line parameter followed by the slot offset (expressed in number of slots). This slot offset should correspond to at least twice the one-way delay between gNB and UE. Command-line parameter at gNB side: `--gnb_k2`
PUSCH scheduled by RAR UL grant (clause 6.2.1.1 of [9])	With reference to slots for a PUSCH transmission scheduled by a RAR UL grant, if a UE receives a PDSCH with a RAR message ending in slot n for a corresponding PRACH transmission from the UE, the UE transmits the PUSCH in slot $n + k_2 + \Delta$ (clause 8.3 of [10]). The enhancement is to introduce an offset k_{offset} so that UE transmits the PUSCH in slot $n + k_2 + \Delta + k_{offset}$. Correspondingly, the frame number where UE transmits the PUSCH is calculated taking k_{offset} into account. The enhancement is implemented at the gNB scheduler.
DCI scheduled PUSCH (clause 6.2.1.1 of [9])	If *n* is the slot with the scheduling DCI, the slot where the UE transmits the PUSCH is $n \cdot \frac{2^{\mu_{PUSCH}}}{2^{\mu_{PDCCH}}} + k_2$ (clause 6.1.2.1 of [11]). The enhancement is to introduce an offset k_{offset}. The transmission slot of DCI scheduled PUSCH is then $n \cdot \frac{2^{\mu_{PUSCH}}}{2^{\mu_{PDCCH}}} + k_2 + k_{offset}$. The corresponding frame number is also calculated taking k_{offset} into account. The enhancement is implemented at the gNB scheduler.
Extension of DURATION_RX_TO_TX	Additional slot number depending on UE capability between UE Rx and Tx. This is configured via the command-line parameter followed by the additional slot number (expressed in number of slots). This additional slot number should correspond to at least twice of the one-way delay between gNB and UE. Command-line parameter at UE side: `--ue_slot_Rx_Tx`
Extension of random access response (RAR) window (clause 7.2.1.1.1.2 of [9])	Use k2 value to configure additional window size for the reception of RAR at UE side via command-line parameter (expressed in number of slots). The original RAR window size is too small for NTN case. UE periodically complains that no successful RAR reception is done and send periodically

	a new preamble. This can be avoided by introducing the offset for the start of the RAR window. Command-line parameter at UE side: `--ue_k2`
Extension of UL future TTI request and UL virtual RB mapping	Maximum number of frames buffering UL scheduling information at gNB for UL_tti_req_ahead and vrb_map_UL can be configured via command-line parameter (expressed in number of frames). This value must be an exponent of 2. UL_tti_req_ahead contains the information about the MAC layer PDU, such as system frame number, slot index, number of PDUs that are included in this message, information about PRACH/PUSCH/PUCCH/SRS PDUs, etc. vrb_map_UL contains the virtual resource block mapping for common channels and PUSCH. Originally these two structs buffer 1 frame only. Because of the short propagation delay, 1 frame is enough for cyclic usage. For NTN, the scheduling information buffered is dramatically increased proportional to the propagation delay. The frame number should correspond to at least twice of the one-way delay between gNB and UE. Command-line parameter at gNB side: `--ul_sched_f`
TA update period	To guarantee that a new TA update is only calculated after the old TA update is applied, the TA update period is increased from 10 frames to 50 frames. The TA update period must be at least twice of the one-way delay between gNB and UE.
Deactivation of HARQ processes	DL/UL preprocessor functions for NTN to disable HARQ retransmission are added: nr_preprocessor_ntn() and nr_ul_preprocessor_ntn(). For NTN make ACK=0, which means no ACK/NACK for received DLSCH from UE. “pucch_ack_payload” & “n_HARQ_ACK” are set to 0. Originally, the number of the HARQ processes must be increased to support long delay in scheduling despite that ACK/NACK is disabled. The increment of the number of processes causes the problem that the “ping” command after establishing the connection between gNB and UE shows unstable latency and sometimes the latency measured by ping is much higher than the expected one. The solution is to always set the HARQ process ID to a fixed number and always set the “is_waiting” flag to false for both DL and UL HARQ for NTN case.
Random data in absence of IP traffic	Additionally, random data is now sent for DL in case of no IP traffic and no TA update by removing the check of the DL buffer and TA update frame. By this means the signal constellation in the DL scope can be better observed.

2.2.1.1.4 Terrestrial MAC

The terrestrial MAC is responsible for the configuration of the terrestrial modems. It also interfaces the terrestrial modems with the cRRM.

The configuration parameters of the transmitter are reported in Table 11.

Table 11. Configuration parameters of the TERR MAC

Configuration Parameter	Value	Type
Maximum bandwidth	100 MHz	Static

Number of sub-bands	12, each 7.68 MHz width sub-band can be used for transmitting data	Static
Numerology	1 (i.e. inter-carrier spacing=30 kHz)	Static
Number of RBs per sub-band	8	Static
Number of carriers per RB	32	Static
Allocated sub-carriers inside the sub-bands used.	0..31	Real-time configuration by the MAC
Number of sub-bands used for transmitting data (the change of this parameter requires to start again the scenario).	0…14	configuration by cRRM
MCS (i.e. modulation and coding rate).	0…27	configuration by cRRM

The receiver is configured when the scenario is launched with the fixed parameters and with a number of used sub-bands by default. When the scenario is running, the received decodes from the control channel (PDCCH) from the transmitter the allocated sub-carriers and the MCS.

The indicators sent from the receiver MAC to the cRRM is the SNR.

2.2.1.1.5 Video server, encoder and streamer

The Video Server proposed for the 5G-ALLSTAR testbed is a web service module able to stream video contents up to 8K resolution (7680x4320 pixels) using a Video-on-Demand (VoD) distribution.

The Video Server module is composed by a set of components (shown in Figure 12) in order to provide the requested service. These components are interconnected using REST APIs and are internally orchestrated by the *API service* component, that is implemented as a NodeJS microservice. Among the components of the Video Server module, that are explained in detail in document D4.3 (Implementation of the multi-RAT load balancing algorithms and technical specifications of the relevant interfaces) §7.1, the most important ones are here described:

- **Web interface**: implementing the web GUI at UE side and that contains the *Adaptive Video Player*, that is the component that actually implements the adaptive streaming algorithm based on the measured KPIs at UE side, choosing the most suitable resolution to be played at any moment and requesting video packets for the selected resolution to the Video Server modules
- **VOD encoder**: it performs the video encoding of the uploaded contents (i.e., the videos that are loaded into the Video Server), generating multiple synchronized file versions at increasing resolution and bandwidth; the video encoder outputs are encoded using H.264 video codec and AAC video codec, packaged into MP4 format;
- **VOD module**: it implements the DASH and HLS stream packagers used for delivering the video contents playable by the Adaptive Video player component.

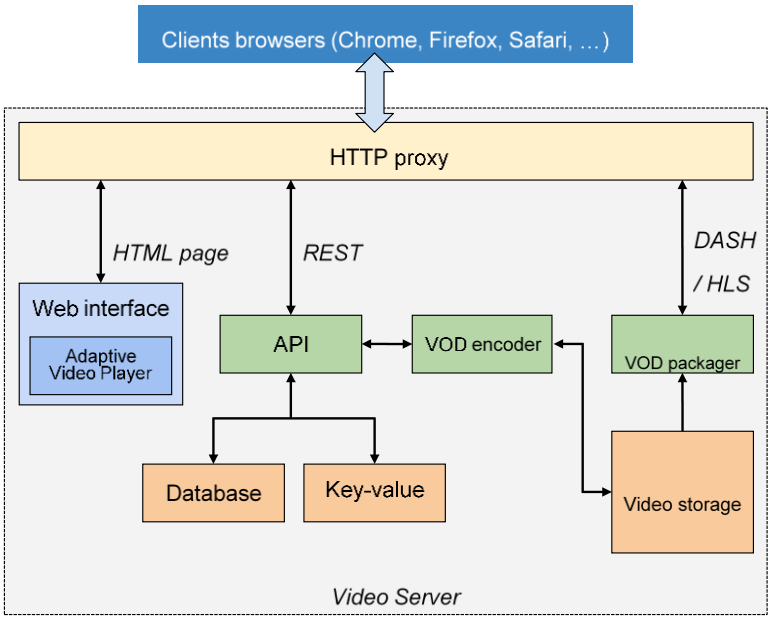


Figure 12. Video Server architecture

2.2.1.1.5.1 The Adaptive Video Player algorithm

The video player module, and in particular the Adaptive Video Player component, implements a control algorithm aiming at maximizing the QoE at user side. This is done maintaining the highest bitrate possible (and so the highest resolution) while avoiding rebuffering events (happening if the queue for the current video quality is empty, e.g., due to the fact that the resolution currently chosen is not supported by the network capacity) and minimizing the video level switches. Moreover, the algorithm obtains a fair resource share among all the long-lived TCP connections.

The full control algorithm specification for dynamic video resolution selection can be found in document D4.3 §7.1.2

It is worth to mention that the control algorithm used for selecting the video level to be downloaded is placed at the client side (i.e., UE side), running directly in the browser web page as a JavaScript executable that parses, decodes and displays the video contents. Indeed, a video is composed by a sequence of chunks that are time-aligned between the different file versions for the different resolutions available at Video Server side, with a chunk duration ranging from 2 to 10 seconds; each chunk contains one or more Group of Pictures (GoP), that are video frames that can be independently decoded, thus allowing the client decoder to switch from different versions in real-time.
The control algorithm has the goal of selecting in real-time the video level **l(t)** in order to maximize QoE perceived by user, based on the network measurements (i.e., estimated bandwidth **b(t)** and round-trip-time **T**) and on local measurements (i.e., current level played **l(t-T)**, received bitrate **r(t)** and playout buffer level **q(t)**).

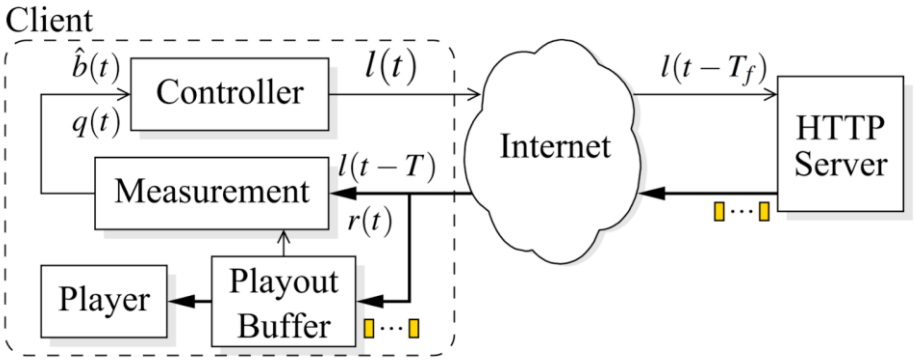


Figure 13. The ELASTIC controller block diagram

The VOD Encoder module has been configured to encode the 8K video using the configuration represented in Table 12. The test video sequence used in the testbed is "*First-8K-Video-from-Space*" from NASA. The DASH/HLS chunk duration is kept fixed to 3.2 seconds and the player buffering target is set to 9.6 seconds. The video codec used is H.264 (x264 software encoder with `preset=medium` and `tune=film`), while the audio codec is AAC (2 audio channels with 128 kbps bitrate).

Table 12. Video Encoder settings

Level	Resolution	Framerate	Bitrate	Profile
L1 (QHD)	640x360	12	500 Kbps	baseline
L2 (QFHD)	960x540	25	1 Mbps	main
L3 (HD)	1280x720	25	2 Mbps	high
L4 (FHD)	1920x1080	25	6 Mbps	high
L5 (4K)	3840x2160	25	20 Mbps	high
L6 (8K)	7680x4320	25	30 Mbps	high

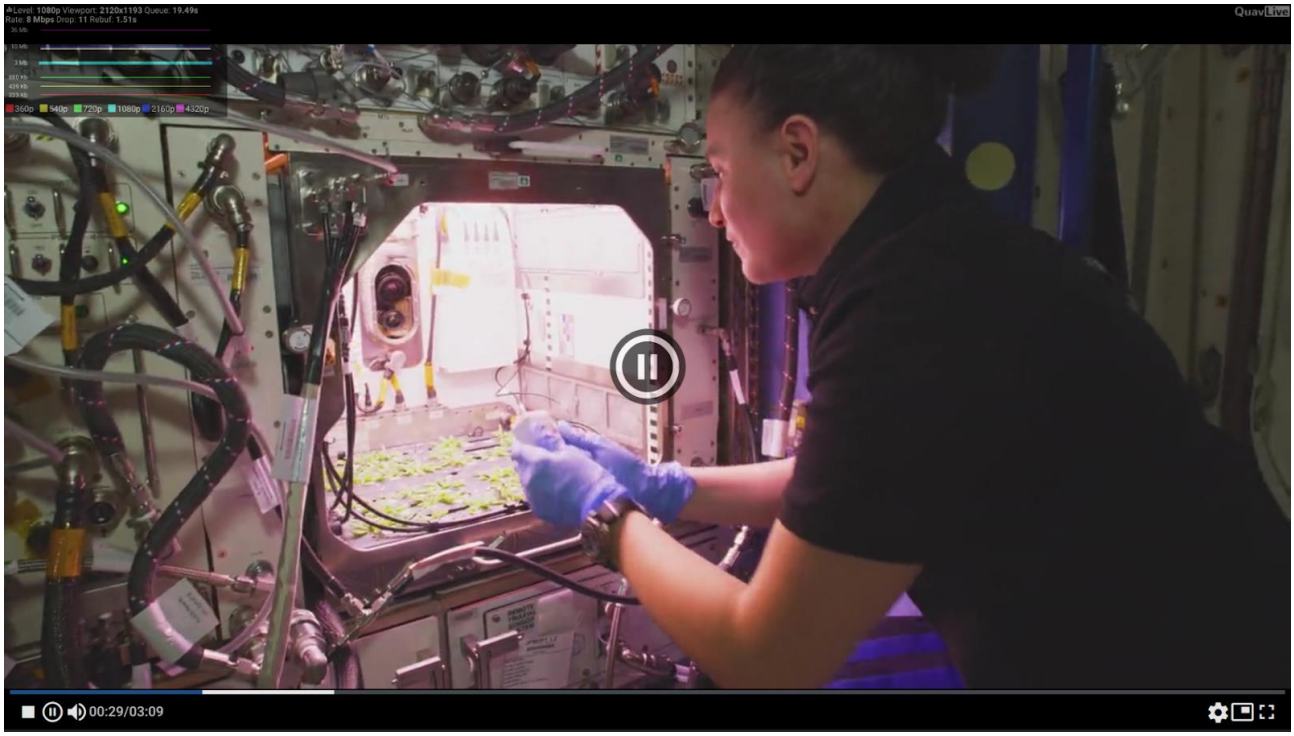


Figure 14. Video player screenshot

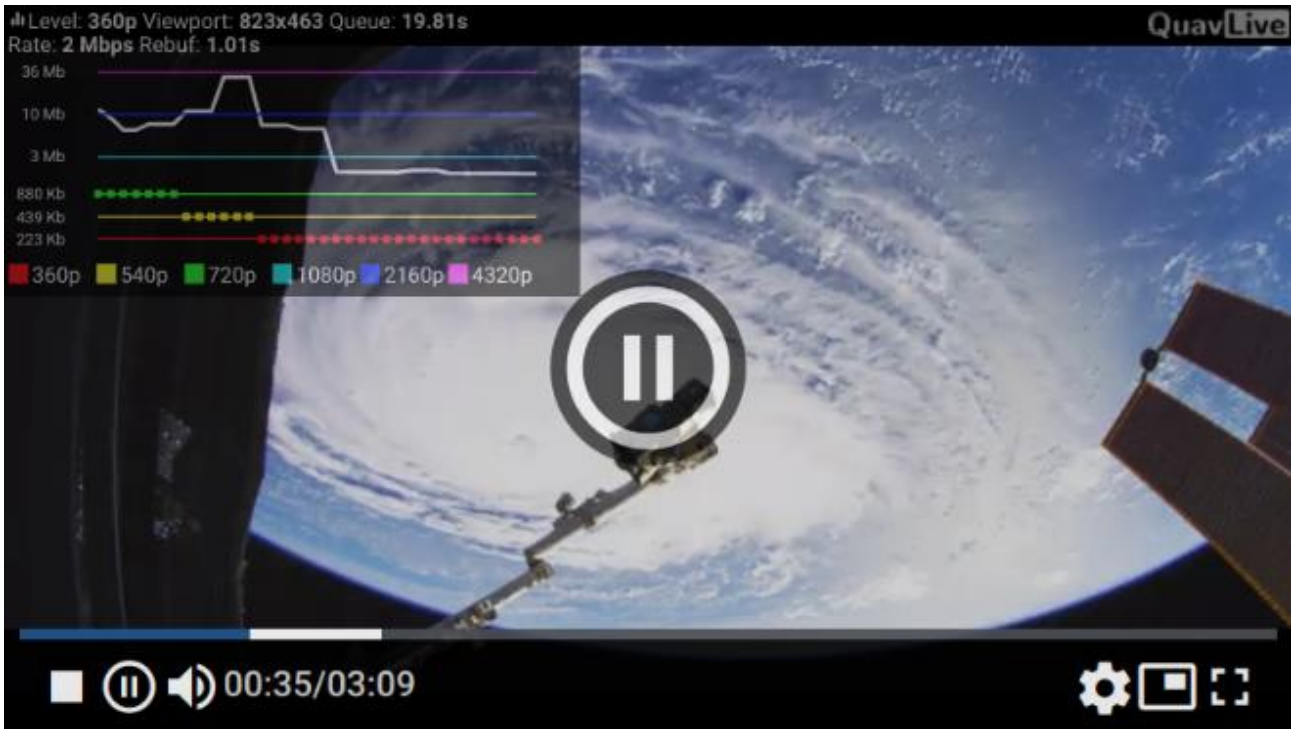


Figure 15. Video Player screenshot with quality levels detail

2.2.1.1.5.2 Testing setup and tools

In order to integrate the Video Streaming components in the testbed platform, we developed a specific Video Player client (shown in Figure 16) based on Selenium library to run a web browser window in headless mode by automated scripts. In particular Chromium web browser has been chosen for the testbed. All the browser functionalities are kept unchanged, except for the browser window resolution, that has been set to 8K (7680x4320 pixels) in order to avoid the video resolution to be capped to lower resolutions.

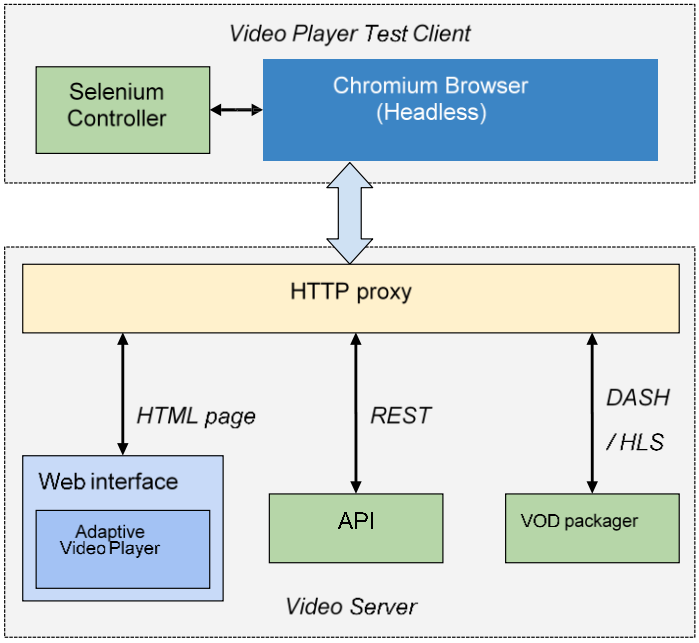


Figure 16. Video Server and Client architecture for testbed

Inside the testbed environment, the tool is able to capture QoE-related metrics about the session from the browser instance. Such metrics are described in Table 13.

Table 13. Video client QoE metrics

Metric name	Description
t	The current playing time in seconds
buffer	The player queued data expressed in seconds
totalVideoFrames	The total number of video frames received
droppedVideoFrames	The total number of video frames dropped by the player if the client CPU is not capable of decoding the video in real time
index	The video chunk fragment index requested
level	The video quality level currently selected in the range L0-L5
rebuf	The number of rebuffering events, including the startup buffering event
rebufTime	The total time elapsed in rebuffering states in seconds, including the startup buffering event
screenWidth, screenHeight	The client browser window size
bandwidth	The average download rate in Mbps

paused, ended	If the player state is “paused” or “ended”
bytesSent	The total bytes downloaded by the client

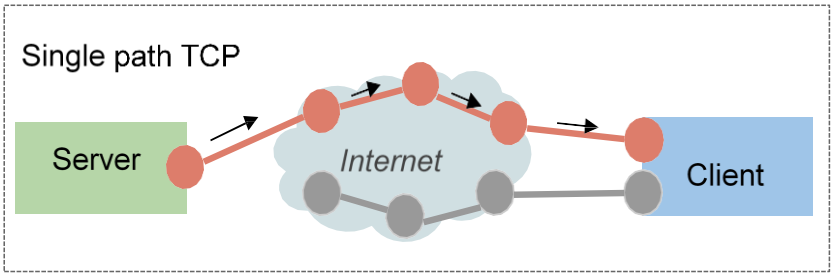
At the same moment, at the server side, it is possible to gather a set of metrics related to the opened TCP connections, as shown in Table 14. Each metric is associated to the destination IP of each TCP connection, in order to show the metrics for all the paths of the testbed. The metrics are obtained using the *iproute2* tools and in particular, using the *ss* command.

Table 14. Video Server metrics

Metric name (for each destination IP)	Description
rtt, rtt_dev, rtt_min	The Round Tip Time current value, the standard deviation and the minimum value
cwnd	The TCP congestion window size
bytes_sent	The total bytes sent
bytes_retrans	The total bytes retransmitted
bytes_received	The total bytes received
send_bitrate	The send bitrate
delivery_bitrate	The delivery bitrate

2.2.1.1.6 (Video) Traffic flow control

The classical TCP/IP communication makes use of a single logical channel (path) between the two endpoints. In case there exist multiple paths, it would be worth to use all of them to improve the end-to-end throughput and to enhance the reliability of the communication. The possibility of using multiple paths for a single TCP connection is called Multipath TCP (MPTCP) and it is standardized and documented in RFC8684.



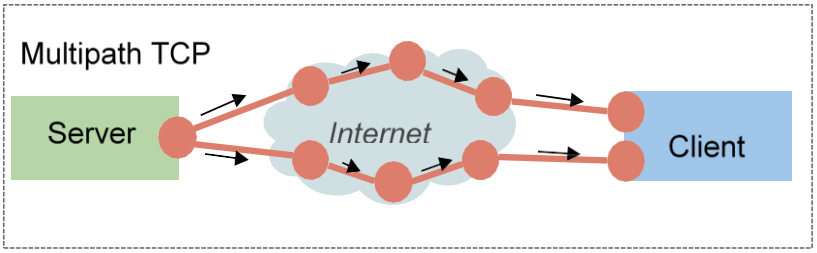


Figure 17. Single and Multi path TCP

The MPTCP protocol is implemented in the Linux kernel as an extension of the classical TCP transport layer: MPTCP acts as a middleware, adding a control layer over multiple TCP connections and exposing them to the upper protocol layers as a single socket, as shown in Figure 18. Each of the TCP connections (named subflow) is independent from the other TCP subflows and from other TCP connections (even towards the same endpoint), but the data to be sent to each subflow is decided by the MPTCP module based on how it is configured.

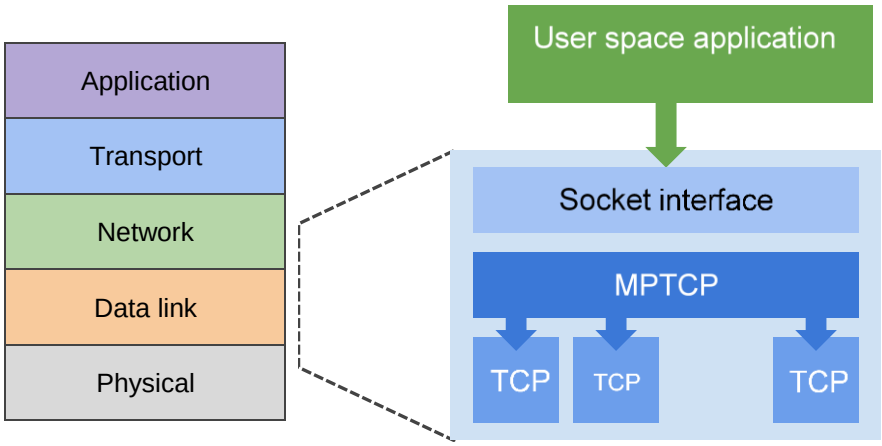


Figure 18. MPTCP protocol stack

MPTCP implementation in the Linux kernel is shipped with some algorithms to schedule the transmission of packets over different subflows. The *path manager* decides how to use the available paths as follows:

- **default**: the path manager works in passive mode, i.e. the host won't initiate the creation of new subflows, but it will accept the creation of new subflows from other hosts;
- **fullmesh**: a full-mesh of subflows among all available peers is created;
- **ndiffports**: this will create N subflows across the same pair of IP addresses, modifying the connection source port;

The *scheduler* module, instead, regulates how the traffic should be distributed among different connections:

- **Default**: the data is sent on subflows with the lowest RTT until their congestion window is full; then, it will start transmitting on the subflows with the next higher RTT;
- **Round-Robin (RR)**: it transmits traffic in round-robin over all the available subflows; it can be tuned using these sysctl parameters:
    - "*num_segments*": the number of consecutive segments to be sent at each round;

- "*cwnd_limited*: if true, it the scheduler tries to fill the congestion window on all subflows; if false, it prefers to leave open space in the congestion window to achieve real round-robin (even if the subflows have very different capacities);

- **Redundant**: the traffic is transmitted on all available subflows in a redundant way; with this approach the transmission latency is minimized, at the expense of no throughput increase than using a single path TCP connection.

2.2.1.1.6.1 The modified Round-Robin (mRR) scheduler

To the purpose of proving a Multipath implementation allowing to decide on user space the quota of data to be sent over multiple connections (Satellite and Terrestrial 5G), a modified version of the Round-Robin scheduler has been implemented for the purpose of the project.

While in the traditional Round-Robin implementation the data segments are queued from upper protocol layers and are sent to one path until *num_segments* segments have been sent to that path, then they are sent to the other path and so on, in the customized Round-Robin (mRR) scheduler developed for the project it is possible to tweak the *num_segments* variable for a particular IP address, so to change the percentage of traffic that goes in one path or in the other one. This implementation makes use of a *quota_table* that contains the following information for each subflow:

- **daddr**: the subflow destination IP address;
- **num_segments**: the maximum quota value to use for scheduling over the subflow;
- **count**: the total number of segments sent over the subflow.

The *quota_table* array is updated at each segment sent and the settings are exposed to the user-space virtual file "/sys/kernel/mptcp_rr/config". Reading from "/sys/kernel/mptcp_rr/config" allows to obtain a list of the latest active subflows with the number of segments sent; writing to "/sys/kernel/mptcp_rr/config" changes the "num_segments" value for the selected destination IP address.

2.2.1.1.6.2 The Multi-Connectivity Algorithm implementation

The Multi-Connectivity Algorithm chosen for the implementation in the PoC is the Wardrop Equilibrium based control has been implemented on top of the modified Round-Robin scheduler in order to dynamically change the quotas of the two available paths dynamically and in real-time in response to variation on the channels' capacity and/or short-term risk of failure.

The algorithm receives periodically the status of the channels from the cRRM with a JSON message (specified in §2.2.1.1.1.3), computes the new quotas and applies them in the mRR scheduler, so the user-plane traffic (video streaming) is sent through the two paths with equalized latency.

In detail, the latency function that is equalized for the two paths is the following one:

$$l_i\big(x_i(t)\big) = \frac{x_i(t)R_i(t)}{C_i(t)},$$

where x_i is the amount of traffic sent on the i-th path, C_i is the capacity of the i-th path and R_i is a short-term risk of failure of the i-th path. This means that if the capacity of a path decreases, then part of the traffic going on that path will be routed on the other available path to equalize latencies, as well if the risk for a path to fail increases, then then traffic will be partially moved accordingly, until latencies are again equalized. More details on the algorithm mathematics can be found in document D4.2 §4.3, while the customization of the algorithm for the Testbed can be found in document D4.3 §6.1.1

2.2.1.1.7 Inter-testbeds VPN

This module is required to put into communication the Core EU Testbed, located in Grenoble (France), with the SAT platform, located in Erlangen (Germany).

2.2.1.2 Configuration parameters

This section describes the configuration parameters for each one of the SW modules of the EU Testbed.

2.2.1.2.1 Testbed setup

The overall EU testbed configuration parameters are reported in Table 15.

Table 15. Overview of the configuration parameters of the EU Testbed

			TERR	SAT
Frequency (IF) *Note I			1 GHz	DL 2169.08 MHz UL 1769.08 MHz
Duplex Mode / Channel Bandwidth			FDD / 100 MHz	FDD/ 40 MHz
Resource Definition	NB of Sub-bands or RBs		16 Sub-bands	106 RBs
	RB	Numerology/ SCS	n=1/ 30 KHz	n=1/ 30 KHz
		Subcarriers	32	12
		Symbols	14	14
	NB of RBs per sub-band		8	N/A
MCS order			2,4,6,8	MCS table 5.1.3.1-1 of [7]

Note I: Channel emulators not enabled to emulate mmWave channels

In the start-up configuration the frequencies and sub-bands are initially set as shown in Figure 19, that is 5 sub-bands (#6, #7, #8, #9, #10) in the middle of the total band assigned to SAT and the remaining sub-bands assigned to TERR, except #0, #1 and #14, #15 set as guard-bands

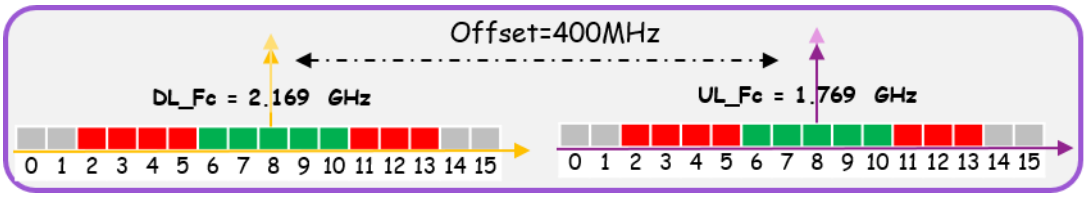


Figure 19. Initial resource assignment of the TERR (Red) and SAT (Green) modems in the frequency domain

2.2.1.2.2 Configuration parameters for the TERR gNBDU_dRRM

List of parameters, and of their ranges, requires by the CEA testbed. Configuration parameters can be either static (reconfiguration requires a reboot of the scenario) or they can changed dynamically when running a scenario (see Table 16)

Table 16. Parameters configuration overview

<table>
<tr><th>Parameter</th><th>Configuration</th><th>Values</th></tr>
<tr><td>Carrier frequency (Fc)</td><td>Static</td><td>For the testbed, any carrier frequency is supported in the K and Ka bands.</td></tr>
<tr><td>Numerology (μ)</td><td>Static</td><td><table><tr><td>μ</td><td>SCS (kHz)</td></tr><tr><td>0</td><td>15</td></tr><tr><td>1</td><td>30</td></tr><tr><td>2</td><td>60</td></tr><tr><td>3</td><td>120</td></tr></table></td></tr>
<tr><td>Sampling frequency (Fs)</td><td>static</td><td><table><tr><td>Fs(MHz)</td><td>Band (MHz)</td><td>Useful Band(MHz)</td></tr><tr><td>61.44</td><td>50</td><td>46.08</td></tr><tr><td>122.88</td><td>100</td><td>92.16</td></tr><tr><td>245.76</td><td>200</td><td>184.32</td></tr></table></td></tr>
<tr><td>Sub-bands</td><td>static</td><td>The whole band is divided into 14 sub-bands. Those sub-bands can be switched OFF individually to let room for satellite transmission.</td></tr>
<tr><td>Modulation and Coding Scheme (MCS)</td><td>dynamic</td><td>The selected MCSs regularly cover CNRs from -2 dB to 29 dB providing TBLER<10-2 in the AWGN channel.<table><tr><td>MCS#</td><td>Bits/symb.</td><td>CodeRate(x1024)</td></tr><tr><td>0</td><td>2</td><td>120</td></tr><tr><td>1</td><td>2</td><td>193</td></tr><tr><td>2</td><td>2</td><td>308</td></tr><tr><td>3</td><td>2</td><td>449</td></tr><tr><td>4</td><td>2</td><td>602</td></tr><tr><td>5</td><td>4</td><td>378</td></tr><tr><td>7</td><td>4</td><td>490</td></tr><tr><td>9</td><td>4</td><td>616</td></tr><tr><td>11</td><td>6</td><td>466</td></tr><tr><td>13</td><td>6</td><td>567</td></tr><tr><td>15</td><td>6</td><td>666</td></tr><tr><td>17</td><td>6</td><td>772</td></tr><tr><td>19</td><td>6</td><td>873</td></tr><tr><td>21</td><td>8</td><td>711</td></tr><tr><td>23</td><td>8</td><td>797</td></tr><tr><td>25</td><td>8</td><td>885</td></tr><tr><td>27</td><td>8</td><td>948</td></tr></table></td></tr>
<tr><td>Combination of RBs (RB combination)</td><td>dynamic</td><td>$14 \times N$ RBs are available. There are N RBs in each of the 14 sub-bands. Any combination of RBs is possible in the sub-bands switched ON.
N depends on the combination $\{Band, \mu\}$:
$\{50\ MHz, 0\} \rightarrow N = 8$
$\{50\ MHz, 1\} \rightarrow N = 4$
$\{50\ MHz, 2\} \rightarrow N = 2$
$\{50\ MHz, 3\} \rightarrow N = 1$
$\{100\ MHz, 1\} \rightarrow N = 8$
$\{100\ MHz, 2\} \rightarrow N = 4$</td></tr>
</table>

		$\{100\ MHz, 3\} \rightarrow N = 2$ $\{200\ MHz, 2\} \rightarrow N = 8$ $\{200\ MHz, 3\} \rightarrow N = 4$ Note: the combinations $\{Band, \mu\}$ not listed above are not possible.

2.2.1.2.3 Configuration parameters for SAT gNBDU_dRRM

The implementation of the SAT gNB-DU_dRRM is assuming:

- Frequency within the NR FR1 range
- numerology µ=1 (30 KHz)
- MCS according to MCS table 5.1.3.1-1 of [7]
- Bandwidth of 40 MHz
- Max assignment of 106 RBs

Configuration is performed through configuration file fed to the gNB, this will setup the servingCellConfigCommon parameters. The list of parameters available for configuration is reported in Table 31 of the Annex. The generated binary file has to be shared with the UE in order to get the RRC configuration.

What is not in the configuration file is either derived from the configuration according to the 3GPP procedures or hardcoded in the code. Therefore, this current implementation do not support runtime reconfiguration of the gNB and UE. In order to reconfigure the modems in runtime, the following steps are required:

1) Update the .conf file
2) Restart the gNB
3) Transfer the generated binary file to the UE

This process can take up to 30 seconds.

2.2.1.2.4 Other configuration parameters

The list of input parameters desired for control of the 5G-ALLSTAR core are reported in Table 17.

Table 17. Configuration parameters of the 5G-ALLSTAR core

Configuration parameters	Description
UE Type and Capabilities	• Fixed or Mobile • Terrestrial only, Satellite only, or both; (i.e. reflect the possibility to use the 2 modems, not necessarily simultaneously) • Single Connectivity or Multi-Connectivity (MC supports satellite and terrestrial access simultaneously) • Supported (or allowed) Frequency Channels, per access • Supported (or allowed) bandwidth per access • Supported (or allowed) MODCOD types
Radio Frequency Range	FR1 (sub 6 GHz)

Configuration parameters	Description
UE SINR and Channel	Scenarios with Channel Emulator: • Constant SINR Or • UE SINR time series, per access type [non shared frequency scenario] Or • 3GPP Channel Models as per 3GPP TR 38.901

UE Satellite Delay Model	Parameters:
	• SAT channel one-way propagation delay
	• Doppler
	• Gain
	• Simulation Centre Frequency
	• RF Centre Frequency
	Satellite Delay model input in the Channel Emulator, among:
	• Fixed models
	  ○ GEO (delay=250 ms)
	• Variable model
	  ○ More realistic GEO (periodic delay model, with slow variations)
	More details in §3.2.1.

UE mobility	• A route for the UE can be defined through the PROP-SIM Geometric Channel Modelling tool available for the terrestrial link. • Time series indicating how the mobile UE moves within he coverage for a given scenario [mobile scenarios only] NB: time series shall be consistent with UE SINR time series
Server/client Application parameters	High level parameters required to start servers application communication (IP configuration, (if applicable), port, +) as relevant. Also depends on the exact application tested Category of user traffic, and their key parameters : • Adaptive Video streaming (number and characteristics of the video layers: quality; bit rates) • Background File transfer (TCP-based ; UDP based streaming with data rate to be set) • Audio streaming (TCP-based ; UDP based streaming with data rate to be set • Maybe: interactive Video session (parameters depending on the software application used)

2.2.1.3 Video Server and Client

The Chromium browser is started with the automated scripts at UE side with the following command line switches:

- --**headless**: run without rendering the graphical user interface
- --**disable-gpu**: disable rendering using GPU
- **--kiosk**: removes status bars and other interface elements;
- **--ignore-certificate-errors**: allow to visit self-signed certificates HTTPS pages

- **--no-user-gesture-required**, --autoplay-policy=no-user-gesture-required: allow to start the playing without any user interaction
- **--window-size**=width,height: the window size applied to the browser
- **--disable-application-cache**: disable the application content caching; in order to start without any data from the previous test runs, the ~/.cache/chromium directory is removed before any test run.

The Video Server at CN side accepts the client requests sent via HTTP with this scheme:

```
https://<server_ip>/embed/<stream_id>?autoplay=1
```

Where:

- **<server_ip>** is the server IP address visible to the client;
- **<stream_id>** is the VOD stream unique identifier;
- **autoplay=1** allows the playing to be started automatically when the page is loaded.

Configuration parameters for the MPTCP module:

During the operation of the MPTCP module it is possible to change the parameters of MPTCP. In particular it is possible to assign a quota for a particular IP address writing on the file "/sys/kernel/mptcp_rr/config".

```
# Usage: echo "daddr=quota" > /sys/kernel/mptcp_rr/config
> echo "33597632=80" > /sys/kernel/mptcp_rr/config
> echo "50374848=20" > /sys/kernel/mptcp_rr/config
```

For example, as shown the figure, the mRR module will schedule the 80% of input segments to the "33597632" (that is the int representation of address 192.168.0.2) destination IP and the 20% to "50374848" (the representation of address 192.168.0.3). These settings can be changed in real time from user space, changing the behaviour of the modified RR scheduler.

It is also possible to read usage statistics and current settings from "/sys/kernel/mptcp_rr/config":

```
> cat /sys/kernel/mptcp_rr/config
192.168.0.2 daddr=33597632 num_segments=80 count=10
192.168.0.3 daddr=50374848 num_segments=20 count=2
```

2.2.2 Implementation on VMs

All the high-layers components have been implemented inside VMs, so as the cRRM module and the Network Traffic Flow Controller module. This choice has been made in order to keep separate and independent the code running on the various Virtual Machines, making the whole Testbed integration process much more flexible. This choice has the drawback of using more hardware resources than a solution with all the software components installed inside the same physical machine. Anyway, the solution of keeping the software components divided in separated VMs reduces configuration issues and conflicts (e.g., on binding network ports).

The VMs are still managed by the OpenBACH controller, as they can act as OpenBACH agents, so it is possible to orchestrate them in the same way as in case of implementing all the modules inside the same physical machine.

As mentioned before, the Testbed will include:

- a VM for CRAT Video Server and Traffic Flow Controller modules, acting as CN and TAS cRRM, Network Traffic Flow Controller and OpenBACH Controller modules, on PC5
- a VM for CRAT Video Client and Network Traffic Flow Controller, that takes the role of the UE, on PC6

2.2.3 Test automation

2.2.3.1 OpenBACH general description

2.2.3.1.1 Overview

OpenBACH is an open-source tool, dedicated to network metrology and testing, oriented for R&D testbed environments. It is intended to be highly customizable, programmable (open), and evolutive.

Its goal is to support all tasks in the **Configuration and Supervision** domain, in particular:

- to centralize the different test parameters
- to orchestration of the different tasks to be performed in a scheduled and organized manner
- to get logs and statistics
- to display and graph collected data (online, i.e. during tests or offline, after tests), and ease post-processing analysis in general

OpenBACH is intended to be a fully programmable tool. Users can relies on the existing test tools available (though GUI or scripts). In the most advances cases, OpenBACH users may have to write themselves their own scripts to fit their specific needs, but this is made facilitated thanks to its comprehensive framework.

A complete description is available at www.openbach.org

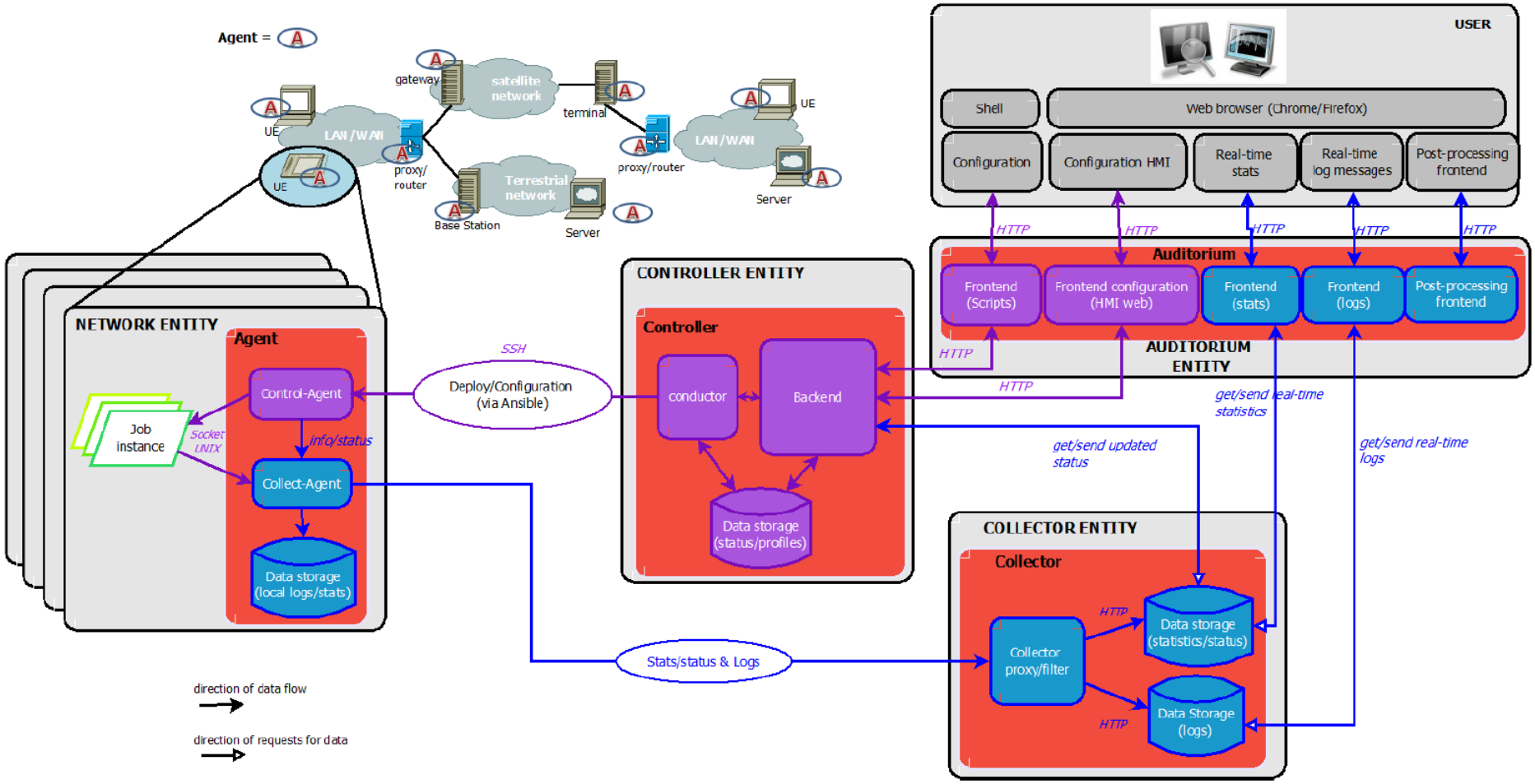


Figure 20. OpenBACH testbed overview

2.2.3.1.2 Main technologies and building blocks

OpenBACH integrates multiple open-source building blocks, that have been selected for their recognized efficiency and widespread use in Linux:

- **Ansible** is one of the most important and "visible" technology in OpenBACH. It is used in the domain of overall orchestration. OpenBACH relies on particular on Ansible playbooks (YAML syntax (.yml files)) for configuration management and multi-machine system deployment
- **Logstash** is the tool used to centralize, ingest and pre-process log data from the different machines
- **Elastic Search** is a multi-criteria, log search and analysis engine
- **Kibana** is the companion tool of Elastic Search, and is used for data visualization
- **InfluxDB** is database used to store and manage the time series data
- **Django** is the Python web framework, used within the OpenBACH controller
- **NGNIX** is a HTTP application platform, used for the OpenBACH Web client and server
- **Grafana** is the open graph tool supporting plot display, in particular for visualization of time series based outputs
- **Python,** finally is the most used scripting language (in particular for writing jobs).
- **C++** alternatively can be used

2.2.3.1.3 Architecture

OpenBACH is made of three main types of entities:

- One or multiple Agent(s)
- one Controller
- one collector

Each of these component are fully software and fully dissociated from the hardware infrastructure. Constraints is that an OpenBACH testbed requires one (and only one) controller, one (and only one) collector, and one or more agents. An overview of the architecture is illustrated in Figure 20.

Agents

The interface between the OpenBACH orchestration system and the commands to be run and scheduled on a machine belonging to the "core" testbed[1] (e.g. start or stop a client, a server, configure some local OS parameters etc.) is assured through the Agent entity.

An Agent is a process (a Linux daemon) that supports low-level communications with the OpenBACH supervision system. Agent must be installed prior to any orchestrated tests, and one agent only is necessary to be installed per machine (physical or virtual machine). Agent shall be able to monitor OS activity (through file OS file sytem, or through local socket) to get locally low-level information that are further aggregated and post-processed in order to report high-level status and logs to user, and build KPIs

Controller

[1] Also named « Network Under Test » in OpenBACH

Controller is the central server of the OpenBACH platform. It host the resources and tools (log supervision, stats, display …) needed to control the distributed components of the testbed.

Auditorium

Auditorium hosts the Web client user interface resources. It directly interfaces with the main controller.

Collector

The Collector is another centralized component maintaining a central database, fed by data sent from the distributed machines composing the Network Under Test.

2.2.3.1.4 Main Concepts

OpenBACH introduces three important concepts: **Jobs, OpenBACH-functions** and **Scenarios**.

Jobs are elementary tasks to be performed, on one single host, and generally executed through a system command. Jobs needs to be installed on each desired host before being run. A job instance is an execution of one job at a particular time, with a given set a parameters and values.

OpenBACH-Function are defined and launched in the Controller, allowing to perform tasks such as install agents/Jobs, configure and schedule Job/scenario instances, perform information/status requests regarding Agents/Jobs/Scenarios and their instances, etc.

Scenarios are a logical set of OpenBACH-Function, organized in a logical way, possibly including scheduling conditions. Scenarios are typically executed from scripts or actions, all from the Controller. Writing and executing scenarios shall be the main ta sks for the OpenBACH operators. A simplified example of scenario could be:

1. **Start an HTTP server** on a machine M1
2. **Generate an object O** of size S on the server
3. Wait until the starting of this server is confirmed (alternatively : after SEC seconds), then **request object O from an HTTP client** on machine M2
4. Wait until the transfer is finished, then **stop the HTTP server** on machine M1 [and get the transfer time]

In this simplified case, 4 jobs would be called in the scenario (bold part of the text). Parameters and scheduling conditions are structured and provided by the OpenBACH user.

2.2.3.2 Integration of OpenBACH and core 5G ALLSTAR components

2.2.3.2.1 Overall description

OpenBACH drives all the orchestration operations used in the 5G-ALLSTAR testbed and for demos. In the rest of this document, any OpenBACH communications will be referred to as the 5G-ALLSTAR platform *management plane*.

Traffic planes (User IP traffic) and control planes interaction.

OpenBACH is mainly be responsible for:

- setting the high-level configuration parameters before start-up of the different components. i.e. by deploying the correct configuration file on the expected locations
- start and stop all the executable on the different machines instances (which may run VM, physical machine, containers, or on dedicated HW (channel emulator, radios), in a scheduled and coordinated way

- centralize the output low-level parameters in view of post-processing (logs, stats)
- manage the "scenarios" by orchestrating the different events

A unique **openbach-agent** daemon is installed per VM or physical machines.

In our proposed software design, **only** local openbach-agent is capable of interfacing with process(es) running on the different hosts on the management plane.

Further, we assume that each software components (see list in next section) is capable of:

- reading (at least) one configuration file, stored on a local (default) folder
- has capabilities to interface in real-time with updated information (events etc.) through a local socket exchange.

Real-time configuration and events notification from OpenBACH to core 5G-ALLSTAR components shall be done through local, socket-based, communication.

2.2.3.2.2 OpenBACH entities

The list of entities to be controlled by OpenBACH in the 5GALLSTAR testbed is as follows.

Note: this is a subset of the components list (from D5.1). Indeed, some components are not relevant for management purposes (i.e. video screen, etc.).

The model assumes only 1 executable per logical SW component.

An OpenBACH agent may interface with one or more executable on the same host.

Table 18. OpenBACH mapping to the EU Testbed PCs

Component ID (supported software)	Component name	OpenBACH component
PC1 (UE Satellite L1/MAC)	UE SAT Modem	OpenBACH Agent
PC2 (UE Terrestrial L1/MAC)	UE TERR Modem	OpenBACH Agent
PC4 (SAT gNB L1/MAC; Satellite dRRM)	SAT gNB-DU	OpenBACH Agent
PC3 (TERR gNB L1/MAC; Terrestrial dRRM)	TERR gNB-DU	OpenBACH Agent
PC6 (UE traffic flow manager ; 8K video player)	UE core	OpenBACH Agent
PC5 (Traffic flow controller ; cRRM; 8K video server)	gNB-CU & CN Testbed C&M 8K video server	OpenBACH collector OpenBACH controller OpenBACH Agent

3 Testbed Integration and Validation process

The 5G-ALLSTAR project aims at proving the viability of the two following System-level concepts:

- Possibility of NTNs seamless integration into 5G TNs: satellites coverages can be considered as fully standard 5G components complementing the terrestrial cells. For this, Service Links (Forward and Return) offered by the Satellite to UEs shall be done in Direct Access using the 5G NR waveform instead of a standard Satellite-optimized waveform (i.e. DVB S2/RCS) and in the same frequency bands as the ones allocated to the TNs.
- Added-value, at a services standpoint, for a hybrid connection involving a satellite link: NTNs complementing TNs shall bring an improvement to Service Qualities and/or User Experience Quality, such as better data rates, or better availabilities.

For this, a series of key technical concepts were identified by the 5G-ALLSTAR partners:

- FhG IIS:
    - 5G NR L1 and MAC layers compatible with the large propagation delays introduced by GEO Satellites
    - Compatibility of the 5G NR with transparent Payload (i.e. the impact of the large 5G NR Peak-to-Average Power Ratio with the End-to-End link budgets)
- CEA:
    - Transmissions in BF-OFDM for a more spectral-efficient use of the Radio Resource (narrower channel spacing)
    - Full compatibility of BF-OFDM transmissions with CP-OFDM receivers (standard for TN UEs)
- TAS:
    - A cRRM to coordinate efficiently the NTN with the TN in a common Frequency Division Duplex (FDD), despite the large propagation delays introduced by GEO Satellites (large delay control loops)
    - An hybrid TN-NTN connectivity management by Traffic switching and steering, based on links respective qualities monitoring
- CRAT:
    - Traffic control based on a “Wardrop Equilibrium” algorithm

These technical concepts are implemented through a series of Core Components each partner was in charge to prototype. In addition to these Core Components, partners had also to develop some specific components allowing the Core Components testing and the adequate interfacing to each-other, as well as their Control and Monitoring when integrated into Testbeds.

The reason of being of the Testbeds and Trial defined and set-up through the 5G-ALLSTAR project is to validate the Core Components, to interconnect them to find out their most appropriate tuning and settings and, eventually, to prove the project’s System-level concepts relevancy.

For this, an incremental integration and validation approach was defined and followed by the partners (see Figure 21):

1) Core Components prototyping and unitary validations, involving Testing and Validation means on each partners’ site
2) Low (~ waveform) and High (~service) layers independent functional and performance validations, involving the following testbeds:

- Terrestrial low layers testbed at CEA in Grenoble
- Satellite low layers testbed at FhG IIS in Erlangen
- Adaptive video streaming testbed at CRAT in Rome

3) Single connectivity testbeds, validating the low and high layers components interfacing but without the introduction of multi-connectivity at this stage:
   - Terrestrial connectivity testbed at CEA in Grenoble
   - Satellite connectivity testbed at FhG IIS in Erlangen
4) Multi-connectivity mechanisms tuning and System-level concepts proving with the European Multi-connectivity testbed

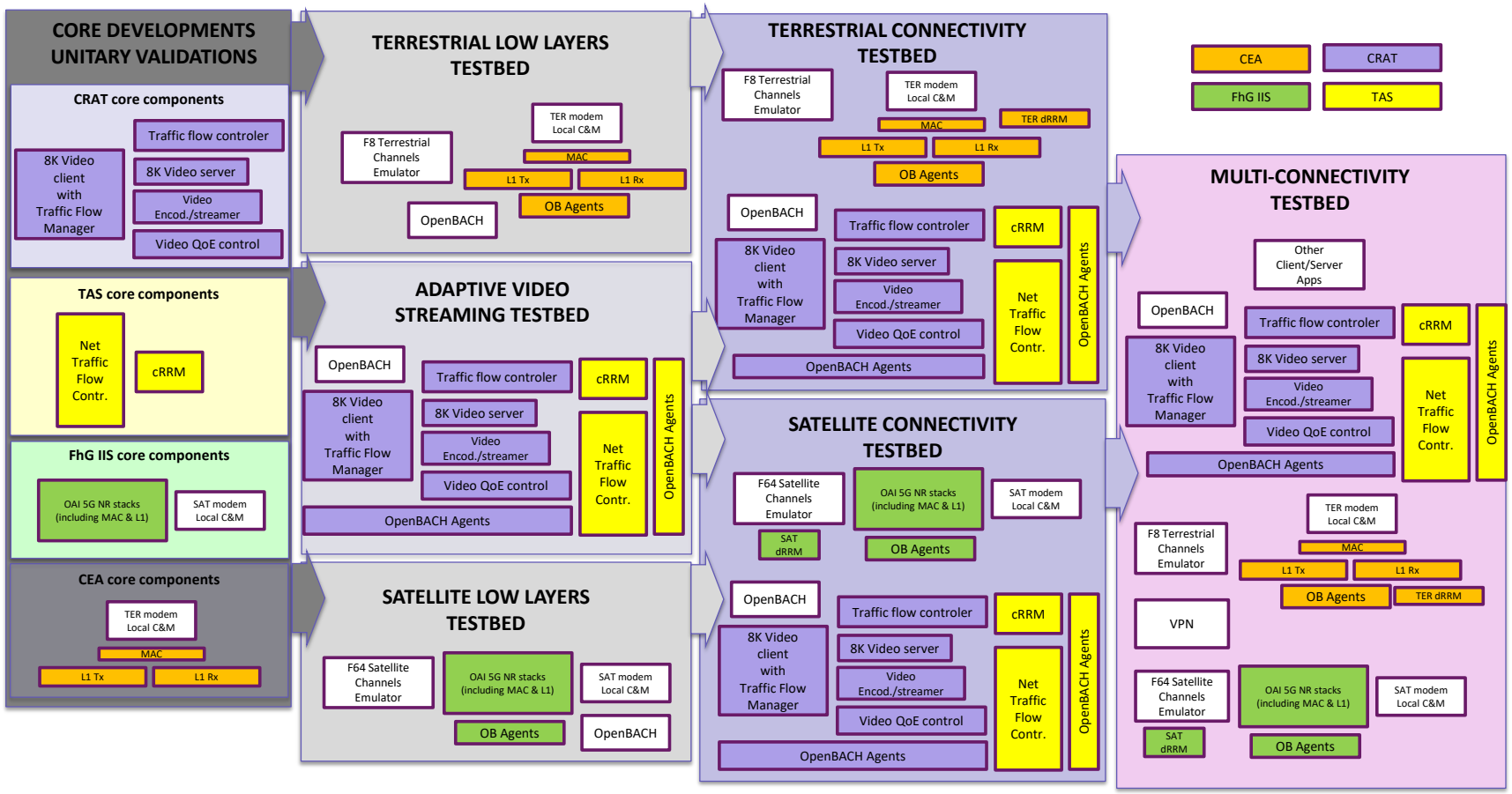


Figure 21. Core Components and incremental Integration & Validation testbeds

3.1 Partners components unitary Integration & Validation

3.1.1 CEA Terrestrial Modems

The terrestrial physical layer is BF-OFDM. This waveform and its advantages in the multi-connectivity scenario have been described in [6]. It has been designed in Hardware Description Language (HDL) and implemented in the Field Programmable Gate Array (FPGA) of a Xilinx System on Chip (SoC). The main obstacles for such a work are (i) the support of multiple configurations and parameters, (ii) the high bandwidth w.r.t the board clock frequency and (iii) the intrinsic complexity of BF-OFDM.

In order to better understand the following implementation description, we recall in Figure 22 the structure of the BF-OFDM transmitter. It is composed of a filter bank (PPN) fed by M CP-OFDM modulators – corresponding to M sub-channels – operating in parallel. A framing stage maps $N/2$ OFDM sub-carriers in order to respect an intra-carrier orthogonality condition. An additional pre-distortion stage compensates the distortion introduced by the filter at the transmitter side. This channel bonding structure highlights the capacity of BF-OFDM to support multi-service scenarios. A service is characterized by a given MCS and a given numerology μ suited to its requirements. Each BF-OFDM sub-channel (or group of adjacent sub-channels) can therefore carry a different service without interfering on the adjacent sub-channels. The structure of the transmitter makes also possible to reconfigure on the fly the sub-channel allocation. The receiver, not shown, actually corresponds to a CP-OFDM receiver.

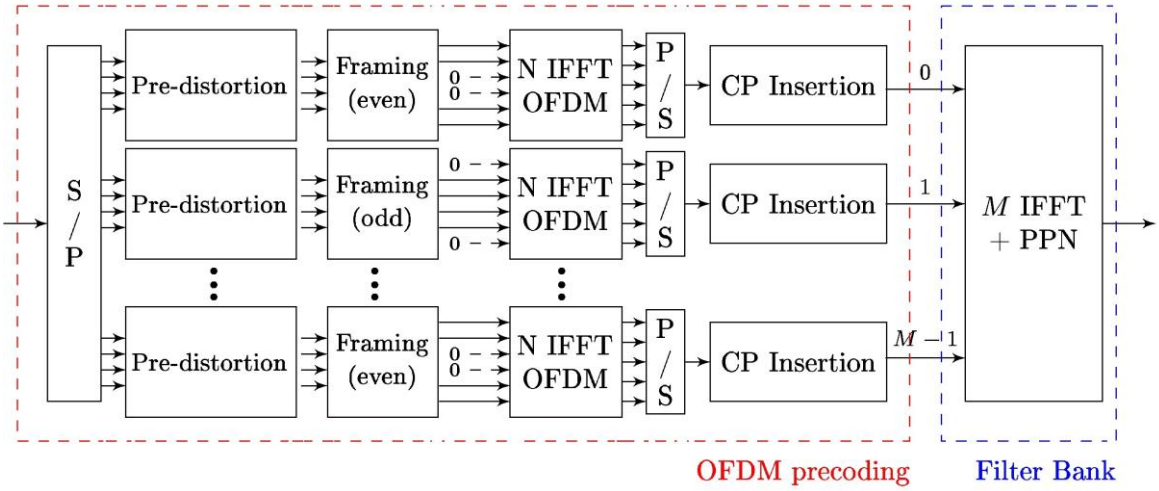


Figure 22. BF-OFDM transmitter

We detail in §3.1.1.1 the strategy we used for the design of the hardware: first, we describe how we dealt with the multiple possible configurations / parameters; and second, we introduce the parallel implementation architecture. FPGA implementation results in terms of complexity and measurements are then reported. In §3.1.1.2 the validation tests are provided.

3.1.1.1 Design and integration

A. Flexible numerology implementation

A characteristic feature of 5G is the possibility to use different numerologies; the carrier spacing and the symbol length can be changed dynamically. We describe here the design for flexible numerology support, using up-sampling processing. The main idea is to keep, in the BF-OFDM architecture, a fixed size of FFT processing and PPN function. We therefore propose an architecture that allows:

- to use only one size of Inverse Discrete Fourier Transform (IDFT), N_{IDFT}, regardless of the numerology (in sub-channels or over time) or the bandwidth.

- to use only one size of Guard Time, N_{CS}, regardless of the numerology;
- to simplify the alternation of the position of the zeros during framing, see Figure 22.

This architecture is depicted in Figure 23. The IDFT module performs a constant size IDFT regardless of the number of carriers N_c in the sub-channel. Zero-insertion at the input of the IDFT corresponds to an oversampling of the signal in the time domain. At the output of the IDFT, it is then necessary to decimate by this factor in order to find the desired sampling rate back. The module "Guard Time Insertion" inserts a guard time of constant length N_{CS}. The module "Spectrum shift", associated with the module "Zeros insertion", allows the insertion and alternation of zeros in a manner equivalent to the Framing in Figure 22. This operation is performed in the time domain by rotating the signal; it does not require any interruption of the data flow. In addition, the shift to the right or left of the spectrum according to the parity of the sub-channel index is controlled simply by the sign of the rotation. This module uses the property that a rotation in the time domain is equivalent to a spectral shift.

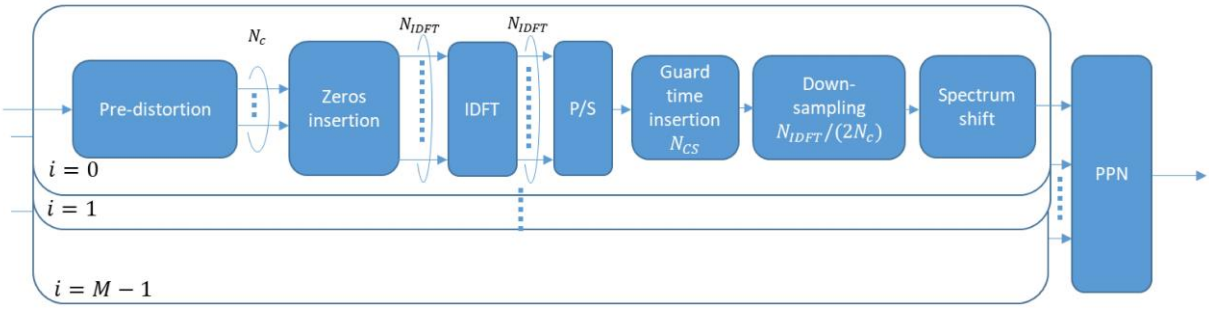


Figure 23. Actual structure of the implemented BF-OFDM transmitter

Table 19 shows the possible parameters for the hardware setup, referring to Figure 23 for the notations. A unique IDFT size of 256 and a guard time size of 16 allow to address the 10 combinations in this table. The whole bandwidth has been divided into $M = 16$ sub-channels – regardless of the numerology and the band – each with N_c subcarriers. Finally a prototype filter with an overlapping factor of $K = 3$ has been chosen. The filter, of Gaussian type, has been optimized to maximize the adjacent channel leakage ratio (i.e. to minimize out-of-bands emissions) while minimizing the complexity of the filtering processing. For the testbed setup, the combination $\{Band = 100\ MHz, \mu = 1\}$ has been chosen.

Table 19. BF-OFDM parameters for the hardware implementation

	Band			
μ	50 MHZ	100 MHz	200 MHz	400 MHz
0	$N_c = 256$	N/A	N/A	N/A
1	$N_c = 128$	$N_c = 256$	N/A	N/A
2	$N_c = 64$	$N_c = 128$	$N_c = 256$	N/A
3	$N_c = 32$	$N_c = 64$	$N_c = 128$	$N_c = 256$

$N_{IDFT} = 256, N_{CS} = 16, M = 16$

B. Wideband architecture

To deal with very high throughput, far from the hardware clock rate, efforts have to be done at the architecture level. Interestingly, the BF-OFDM transceiver architecture can be naturally implemented in parallel as depicted in Figure 22. Basically, a BF-OFDM transmitter can be shown as an aggregation of M OFDM sub-channels of size N combined with a PPN. The associated architecture is then straightforward to derive. The main difficulty lies in the design of the PPN function, which has to deal with M parallel streams to build a high throughput signal to the Digital to Analog Converter (DAC).

At the receiver side, a similar strategy can be applied. The FFT processing can be envisaged working in parallel to create P parallel streams, each one being processed independently by a Frequency Domain (FD) processor. A description of the FD processor is given in Figure 24. After the FFT, physical channels are de-multiplexed. A channel estimation processing is performed on pilot symbols with a frequency and time interpolation and then provided to the equalizer and Log Likelihood Ratio values calculation. It must be noted that specific processing can be performed with neighbour sub-channels processors to avoid bad estimates of the channel coefficients at the edges. Despite the parallel structure of this architecture, some functions must collect information from the P streams: (i) Symbols from the Physical Downlink Control Channel (PDCCH) are sent to a dedicated processing function that aggregates them to perform the PDCCH decoding and (ii) the tracking of the synchronisation – Carrier and Sampling Frequency Offsets (CFO and SFO) – is managed by a unique module that feeds back to the Time Domain (TD) processor the estimation of the value to use for correction.

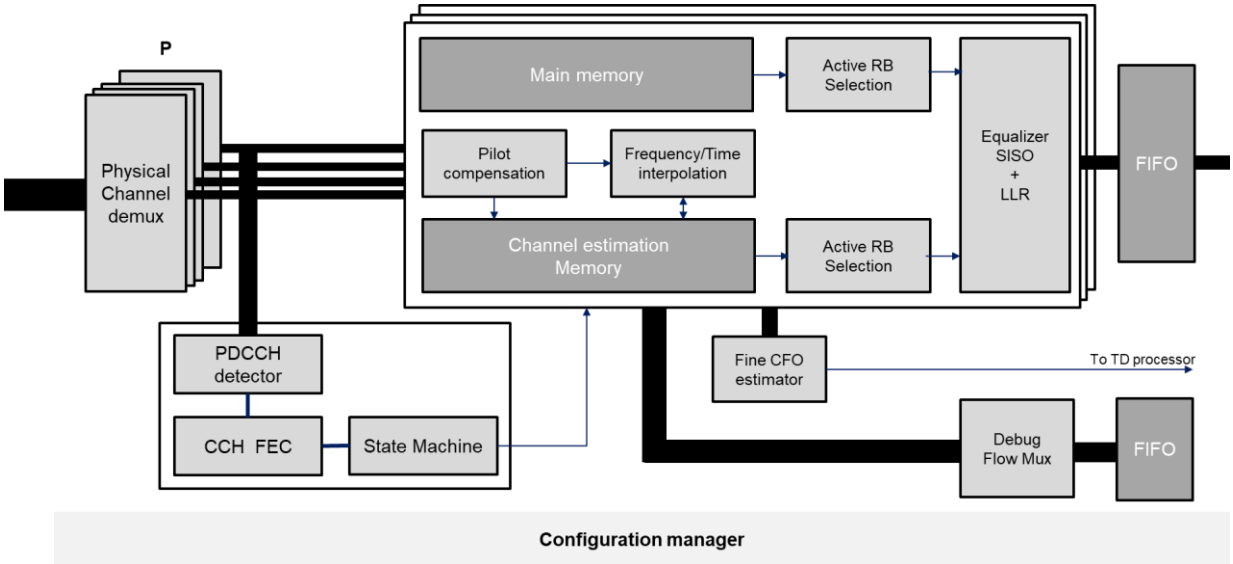


Figure 24. Frequency domain processor architecture

Due to lack of space, the architecture of the TD processor as well as the Low Density Parity Check (LDPC) encoder/decoder, which is a Xilinx IP, is not detailed here.

C. Complexity evaluation

The transceiver has been mapped to a Radio Frequency System-On-Chip (RF-SOC) platform from Xilinx. The resource usage of the transceiver is given in terms of Configurable Logic Blocks (CLB) that constitute the basic FPGA cell. Digital Signal Processing (DSP) cells are combinatorial logic cells dedicated to multiplication and accumulation operations. The number of block Random Access Memory (RAM) tile is also given. Figure 25 and Figure 26 detail the FPGA resource usage per function. For the transmitter, the implementation of the $M = 16$ OFDM transmitters (FFT of size$N_{IDFT} = 256$) is the most complex function. However, it should be noticed that the parallel implementation of the PPN of size $M = 16$ consumes one third of resource. Interface and management cause small overhead. Last, the interface (dispatcher and modulation mapping) function between the LDPC IP and the 16 OFDM transmitter represents less than 8 % of the complexity.

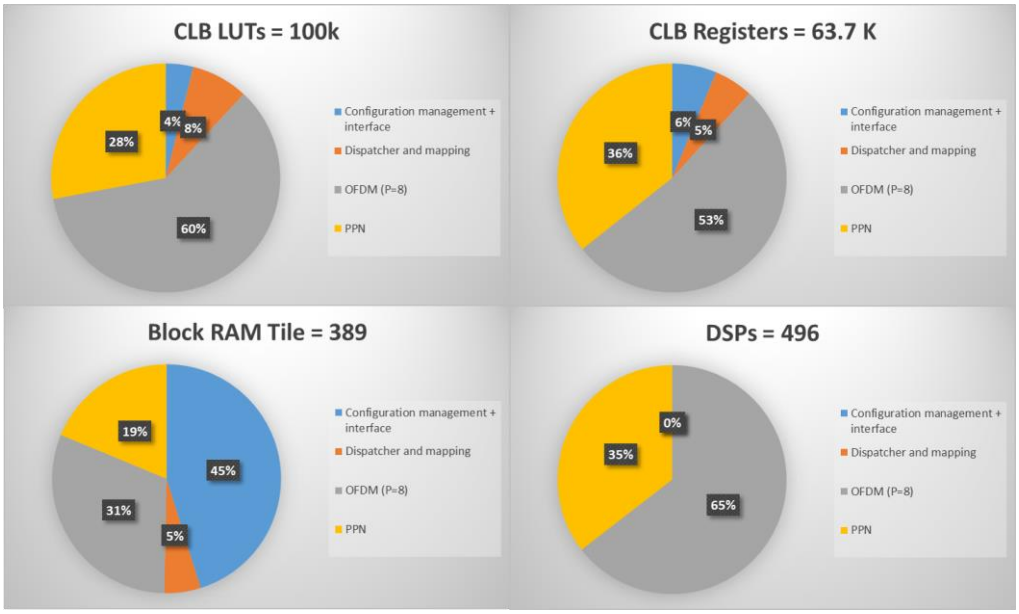


Figure 25. Complexity of the transmitter

When it comes to the receiver, most of the complexity comes from the FFT. It is worth pointing out that the FFT (size $M \times N_{IDFT} = 4096$) with $P = 8$, was identified as the most complex module of the receiver and consumes around 50 % of the actual receiver FPGA implementation. The FD processor, that includes physical channel demultiplexing, channel estimation, PDCCH decoder, equalization and de-mapping up to 256-QAM consumes 53 % of the DSP. The TD processor, in which time and frequency synchronizations are implemented, uses most of the memory to support fast re-synchronization procedure. This feature, that requires large delay lines, is particularly interesting in millimeter wave band when users are spatially multiplexed (if users are dynamically multiplexed in space by beamforming, time and frequency synchronisation must be acquired in real time without going through a dedicated synchronisation mechanism which would reduce the spectral efficiency and benefits of spatial multiplexing). A parallel, $P = 8$, digital downconverter was implemented and add a non negligible complexity especially in terms of DSP resource.

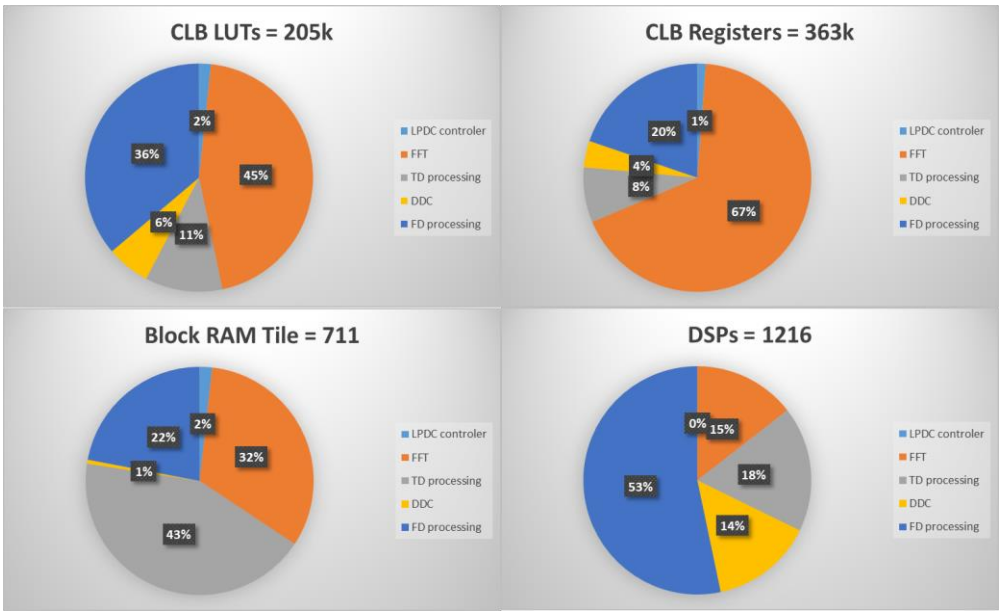


Figure 26. Complexity of the receiver

3.1.1.2 Unitary validation

User interfaces have been developed and used for validation and debug of the terrestrial modems. Figure 27 and Figure 28 respectively show the transmitter and the receiver interfaces. On the transmitter side, one can see the possibility to select a combination of sub-bands and the MCS. On the receiver side, real-time KPIs can be displayed: synchronization, equalized constellation, measured SNR, CRC error rate, throughput, etc.

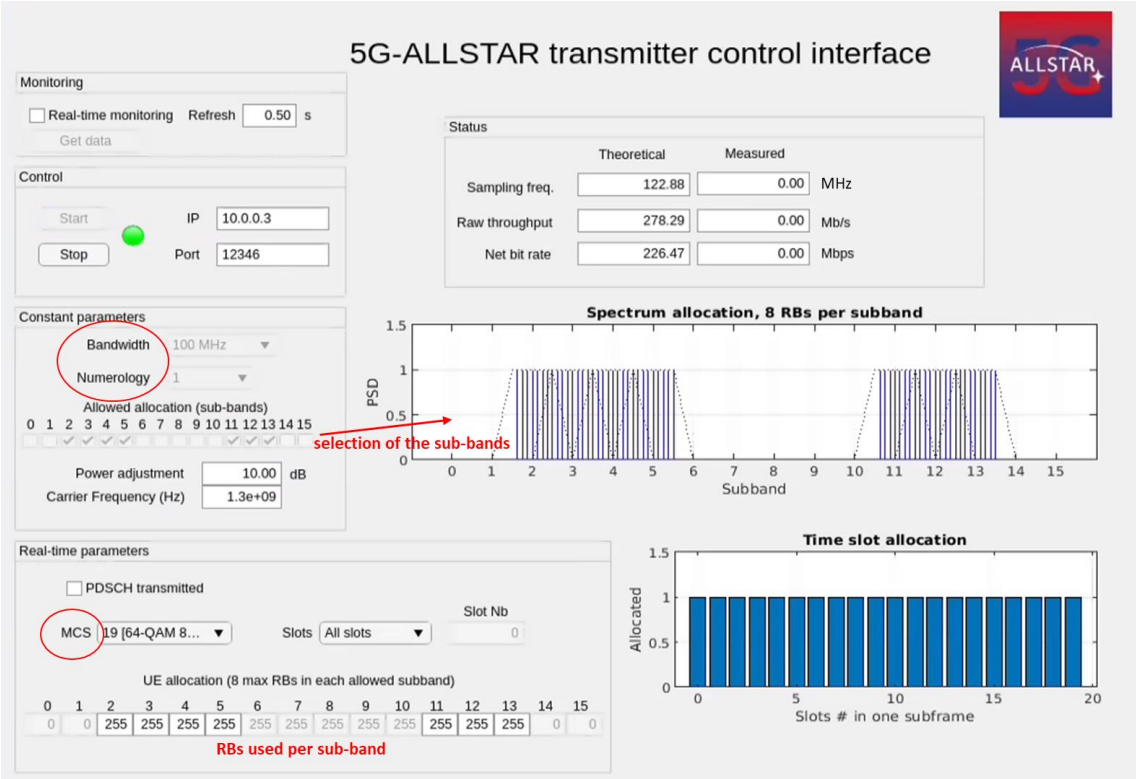


Figure 27. Terrestrial transmitter control interface for validation and debug

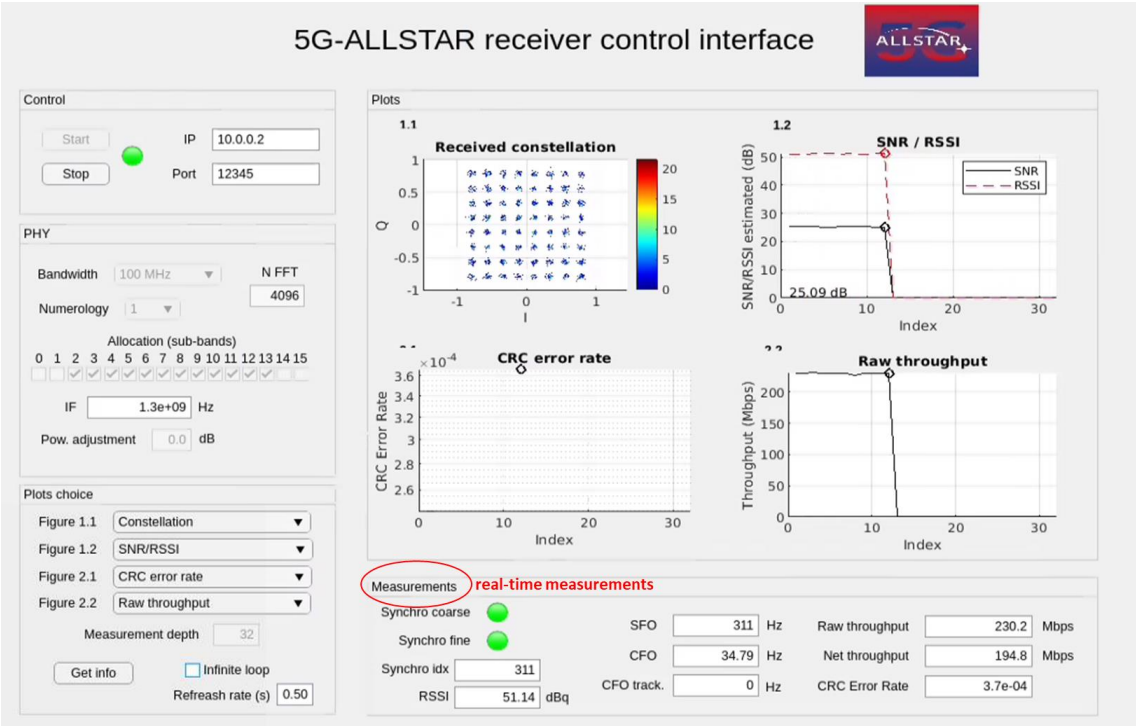


Figure 28. Terrestrial receiver control interface for validation and debug

For the project purpose, a selection of combinations of {bandwidth configuration, MCS} have been validated (i.e. correct synchronization and demodulation of data with expected throughput have been checked), for 100 MHz band with numerology 1 (which is the scope of the testbed). These combinations are shown on Figure 29, where the blue rectangles show the sub-bands allocated to terrestrial system (white rectangles are zeros) and the MCS in green are the one that are available (see Table 7). The choice of the bandwidth allocation with an even number of null sub-bands in the middle of the spectrum is driven by the terrestrial / satellite spectrum sharing strategy: the satellite, with a maximal bandwidth of 40 MHz, will occupy the center of the 100 MHz band.

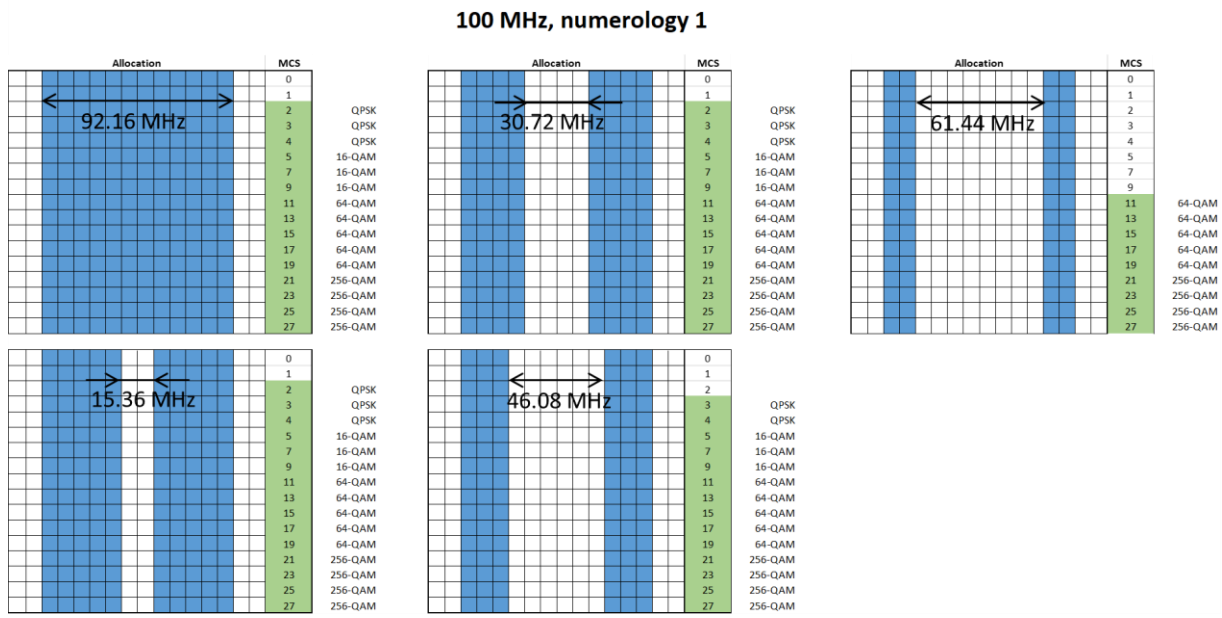


Figure 29. Validated combinations of {bandwidth configuration, MCS} for the terrestrial system

3.1.2 FhG IIS Satellite Modems

PC1 and PC3 will come with all the necessary drivers and libraries installed for a proper functioning of the USRP devices. The two X300s USRP must be connected through PCIe to the two PCs. Also, TX and RX must be inter-connected with the provide RF cables to achieve an UL/DL configuration. Please note that 50 dB attenuators must be connected to both DL and UL. This setup is displayed in Figure 6.

The executable of the OAI modem to run on the USRPs can be built by running the following command from the folder openairinterface5g/cmake_targets/ran_build/build:

On the gNB machine (PC3):

```
./build_oai --gNB -w USRP --build-lib all
```

On the UE machine (PC1):

```
./build_oai --nrUE -w USRP --build-lib all
```

This will build the executables on both sides and shall be done only the first time.

The following steps can be followed to validate the modems:

On PC3, run:

```
sudo -E ./nr-softmodem -O ../../../targets/PROJECTS/GENERIC-LTE-
EPC/CONF/gnb.band66.tm1.106PRB.usrpn300.conf --do-ra -E –nokrnmod -d
```

Two files, rbconfig.raw and reconfig.raw will be generated in the local folder. They shall be copied to the local run folder at the UE side (a simple scp command will do the job). Then on PC1 run the following command:

```
sudo -E ./nr-uesoftmodem --usrp-args "type=x300,clock_source=external" --ue-rxgain
110 --ue-txgain 0 --nokrnmod –E –d
```

The dRRM module is embedded in the gNB and will be initialized automatically.

At this stage, the OAI scope should have popped up at both sides and the user shall be able to see the logging in output on the terminal. If no failure is experienced and the DL and UL constellation are clearly visible on the scopes, then the setup is validated and ready for the integration with other modules in the testbed.

The validation of the SAT modems can be also automated by means of the deployment of OpenBACH. A test scenario with relevant OpenBACH jobs has been developed in order to perform the above-mentioned steps automatically.

3.1.3 TAS cRRM

For the cRRM testing and unitary validation, dummy components where developed to emulate each one of the Testbed components interfacing with it:

- Terrestrial dRRM
- Satellite dRRM
- Traffic Flow Controller (dedicated to and optimized for adaptive video streaming)
- Net Traffic Flow Controller (for any kind of IP traffic routing)

As, the full testbed, emulating the reference System architecture, is only emulating a single UE, cRRM performance is not really an asset as its management role is limited to driving frequency sub-bands and MCS to be used by only two distributed gNBs (gNB-DUs), whereas, in a real system, this would extend to several. Thus, cRRM validation is more a functional validation, making sure that cRRM behaves as defined in terms of signalling procedures.

The cRRM unitary validation tests performed before its delivery to 5G-ALLSTAR partners are the following:

- TCP connection establishment (initiated by external interfacing components)
- dRRM initial configuration procedure (same procedure and messages structures for Terrestrial and Satellite dRRM)
- gNB-DU monitoring procedure
- Resource reconfiguration in reaction to gNB-DU U/L SNR evolution: periodical degradation/improvement profiles applied (triangular and sine)
- Resource reconfiguration in reaction to gNB-DU D/L SNR evolution: periodical degradation/improvement profiles applied (triangular and sine)
- Full resource allocation to the other gNB-DU in case of a gNB-DU U/L Failure

For each test, the following points where assessed:

- Transmitted messages (requests and acknowledgements) compliant to the predefined structures
- Received messages correctly interpreted:
  - Correct parameters extraction from the received messages
  - Rejection/ignoring with no failure of messages non-compliant to the predefined structures
- Respect of signalling procedures sequences
- Running with no failure and cRRM resilience to lack of answers or non-compliant answers to its requests

3.1.4 TAS Network (Net) Traffic Flow Router

Similarly to the cRRM, the Net Traffic Flow Router was tested and unitary validated under a pure software testing environment, involving “dummy” components for each one of its external interface.

As the final emulated System configuration is only involving one UE, performance was not really considered as an asset and the Net Traffic Flow Controller focused on functional tests:

- Validation of the routing of a single forward flow, defined by its destination IP address, the transport protocol and the destination port number:
  - Symmetrical routing case: the return and forward traffic flows are routed to the same access, either the Satellite or the Terrestrial one, depending on the predefined default IP route
  - Asymmetrical routing case: the return and forward traffic flows are routed to different accesses, either the Satellite or the Terrestrial one, depending on the predefined IP routing scheme
- Validation of the traffic load distribution of a single flow, defined by its destination IP address, transport protocol and destination port number, to both Satellite and Terrestrial accesses, according to predefined traffic load ratios (30% over the Satellite access and 70% over the Terrestrial access, in the test)
- Validation of the dynamic routing of a forward traffic flow defined by its destination IP address, the transport protocol and the destination port number, to the predefined secondary access. The predefined “secondary” access (Satellite in test) is in standby (no traffic flow) when the “primary” (Terrestrial in the test) is active (e. g. available). When the “primary” access becomes unavailable, all the forward traffic is routed automatically to the “secondary” access.

Each validation test starts with ping tests over all the possible data paths to check connectivity and is then followed by aniperf3 test of 1000 seconds with the Net Traffic Flow Controller in the configuration corresponding to the test use case.

3.1.5 CRAT adaptive Video streaming client & server

For the adaptive Video Streaming client and server testing and unitary validation, two virtual machines are put in place (one for the server and one for the client). Both the VMs acted as OpenBACH agents and were connected to an OpenBACH controller, that was used to run the tests and show the results in the Grafana environment provided by OpenBACH.

These tests were performed both using just one (virtual) path and using two (virtual) paths, but with fixed quotas configurations for MPTCP.

The unitary validation tests that have been performed are meant to test the full end-to-end chain of high-layer components related to the Video Streaming and reported in §2.2.1.1.6, and in particular:

- API service
- VoD packager
- Web interface
- Adaptive Video Player

The validation tests were also meant to validating the capability of the Video Streaming module to transmit 8K video stream and to maximize the perceived QoE at UE side.

Figure 30 shows some KPIs collected at Video Client side on the Grafana environment, and in particular shows the quality level currently played (that is 6, i.e., 8K, after a small settlement period at the start of video playing) and the Playout buffer queue (that reaches in few seconds the target value of 18 seconds of video packets)

Figure 30. Video Server KPIs on Grafana

3.1.6 CRAT Traffic Flow Controller

In order to validate the Traffic Flow Controller and Traffic Flow Manager modules, the two virtual machines used to validate the Video Server have been employed but the Server VM has been connected to the Client VM with two (virtual) paths, as shown in Figure 31.

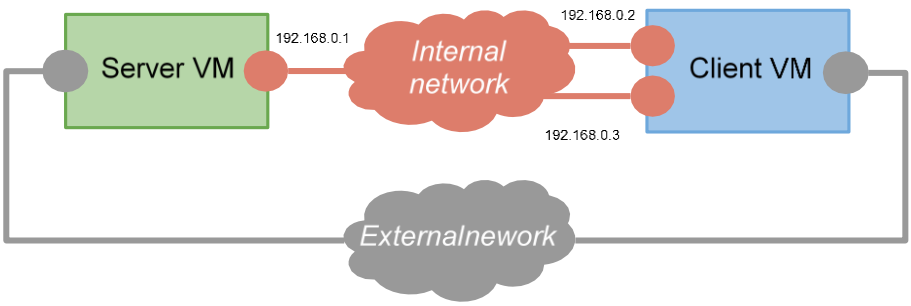


Figure 31. Validation setup for Traffic Flow Controller

As a first validation the Traffic Flow Controller at server side has been validated changing the quotas for the two paths from the default 80%/20% quotas respectively for “192.168.0.2” and “192.168.0.3” paths to the new fixed value of 90%/10%. From Figure 32 it is possible to see the new quotas (green line for “192.168.0.2” path and yellow line for “192.168.0.3” path) and all the KPIs measured by the Video Streaming module on perceived QoE.

Figure 32. Video Streaming and Traffic Flow Controller validation in multi-path scenario

Then the Traffic Flow Controller has been validated in the same standalone setup shown in Figure 33 but using the cRRM module delivered by TAS together with the dummy dRRMs for satellite and terrestrial paths. In Figure 33 it is possible to see that the Traffic Flow Controller changes dynamically and in real-time the MPTCP quotas (in green the one related to “192.168.0.2” path and in yellow the one related to “192.168.0.3” path) and the video streaming is able to reach quality level 6 (i.e., 8K). More details about the validation tests for the Traffic Flow Controller Algorithm can be found in document D4.3, §6.1.1

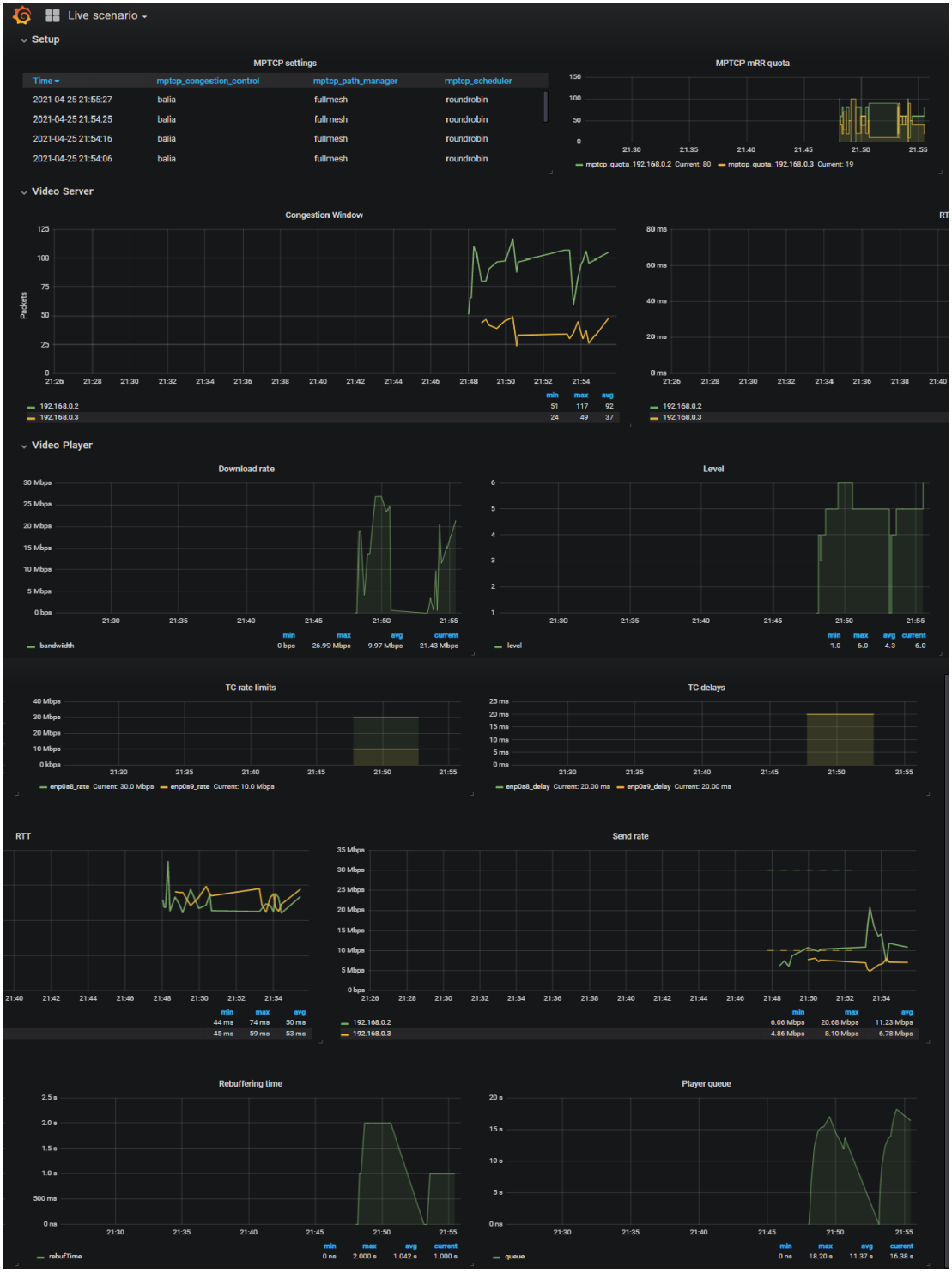


Figure 33. Video Streaming and Traffic Flow Controller validation in multi-path scenario with cRRM

3.2 Low Layers Integration

This section aims at describing the steps of the EU Testbed integration with focus on the low layers of the SW stack. An incremental approach is followed, with parallel standalone integration processes of pairs of unitary testbed components.

3.2.1 Satellite Low Layers Testbed

The integration of the SAT platform entails the integration of the SAT modems with the SAT channel emulator. This step aims at setting up a bidirectional communication link between the SAT UE and the SAT gNB with a radio channel emulated by the PROPSIM F64, and eventually validating the full operation with an UL and DL data transmission.

The HW setup consists of PC1 and PC4 connected through 4 RF connectors to the first channel emulation unit of the F64 channel emulator, as pictured below:

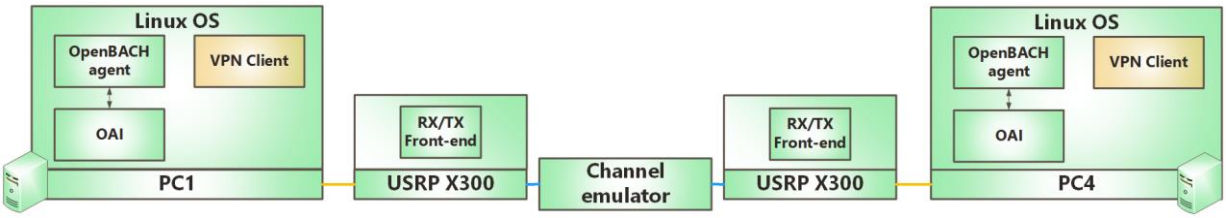


Figure 34. Satellite Low Layers platform architecture in Erlangen, Germany

The following steps can be performed in the Keysight Channel Emulator to configure the PROPSIM in order to emulate a satellite channel with propagation delay equivalent to the delay of a GEO satellite channel:

1) Generate the satellite model in the satellite model editor or selected a custom made file compliant to the ASO data types as defined in §2.1.1.1.1.1. Within the scope of 5G-ALLSTAR, an arbitrary model is used. The parameters listed in Table 20 are needed for the channel model configuration to define the time series of LOS points of the trajectory. The GEO satellite channel deployed in the EU Testbed features constant delay, null Doppler and gain.

Table 20. Example input parameters for the PROPSIM ASO channel model editor tool

Inputs parameters for the channel emulator	
SAT channel one-way propagation delay	240 ms
Doppler	0 Hz
Gain	0 dB
Simulation Centre Frequency	2169.08 MHz
RF Centre Frequency	2169.08 MHz

2) Create the emulations for the ASO in the Editing View:
   a. add the input and output RF elements of the Propsim RF port mappings
   b. load the ASO model file in the channel model block
   c. connect all the blocks together accordingly
   d. Setup the power and gain parameters for each emulation block
   e. Build the SAT emulation and save
3) At the end of the process a SIM file is generated.

Once the setup is ready, the integration consists of the following steps:

1) Run the SAT gNB and UE modems with the proper OAI commands (as per 3.1.2)

2) Load the SAT PROPSIM emulation in the Running View and click Play to start the emulation
3) Check the output power levels as seen by the PROPSIM and displayed on the relevant indicators
    a. If the power levels are too low or are saturating, adjust the RF configuration parameters at PROPSIM side
    b. Else, continue
4) Verify on the OAI UE and gNB scope that the DL and UL are both working
5) Ping the gNB from the UE, and vice versa, and verify the proper behaviour
6) Setup a data transmission in DL and UL, and verify the reception of data on both sides

The integration of the SAT Platform, Figure 6, is considered done once all steps have been successfully completed.

The SAT platform is validated by the test run and described in §4.3.2 and §4.3.7.

The selected extensions of 5G NR protocol stack to support NTNs funded by the EU Horizon 2020 EU-Korea 5G-ALLSTAR project have been validated both in a lab environment by emulating a SAT channel through the PROPSIM F64 channel emulator and by transmitting a 5G NR waveform over a real GEO satellite. In February 2021 bidirectional over-the-air tests based on the satellite modems provided by FHG IIS were conducted within the ARTES 5G METEORS program by the Munich Center for Space Communications (Bundeswehr University Munich) together with Fraunhofer IIS. During the joint over-the-air tests, the OAI UE performed the initial connection setup to the OAI gNB by using a specifically adapted RA procedure for 5G over satellite. After the successful setup of the connection, the 5G UL and DL transmission signals in FDD configuration with a bandwidth of 10 MHz were decoded with two different modulation schemes, namely QPSK and 16-QAM. Furthermore, a 5G timing advance procedure, which worked smoothly, was tested to keep the UE synchronized to the gNB. The DL and UL OAI scopes are displayed in Figure 35. More information are available at [12] and [13].

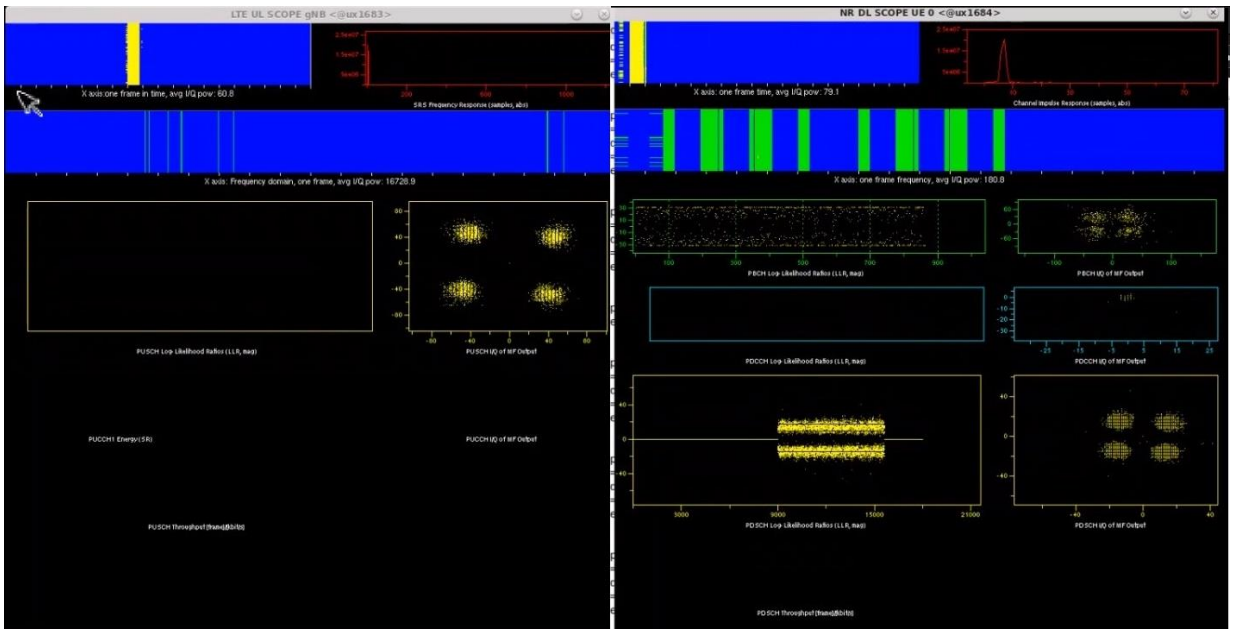


Figure 35. DL and UL OAI scopes during the 5G over satellite trials by FHG IIS

3.2.2 Terrestrial Low Layers Testbed

After unitary validation of the terrestrial link, see §3.1.1.2, this step consists in the validation of the terrestrial platform. The main point for this is the integration of the MAC, see §2.2.1.1.4, to replace the MATLAB interfaces, on PCs 2 & 3. The terrestrial platform is validated by the test run and described in §4.3.6.

3.2.3 cRRM

The I&V of the cRRM follows the logic:

Table 21. I&V of cRRm

Task	Status
Development of the standalone component and interfaces	Validated
Validation of the interfaces and parameters exchanges over the interface with developments of dummy components playing roles of dRRMs, flow controller routers	Validated
Validation of the OpenBACH interface with the OpenBACH framework (correct display of the various metrics and parameters: received SNIRs; allocated MCS ; report of the computed link throughputs;	Validated
Validation of the correct behaviour in an incremental way • Simple setup with only one link (either satellite or terrestrial) : correct assignment of MCS, with static and dynamic mode, capable to react from SNR variation; static then dynamic assignment of RB according to the static then varying band variation profile • Same with multiple links	MCS assignment has been verified during the test reported in §4.3.6. Depending on the SNR reported by the TERR modem, the cRRM adapts the MCS.
Validation with user traffic and correct reports of throughputs to the flow routers	Validated
Validation of capability to report instantaneously link failure in case of such channel profile is used over the interface with dRRM(dummy)	Validated

3.2.4 Network Traffic Flow Router

The I&V of the Network Traffic Flow Router follows the logic:

Table 22. I&V of NetworkTraffic Flow Router

Task	Status
Correct internal identification of traffic profile according to the configured Network Traffic Flow rules	Validated (Standalone)

Correct application of each policy (see 2.2.1.1.2) on the flow sharing over the satellite and terrestrial interface (for each if the network and UE Network Traffic Flow Router)	Validated (Standalone)

3.3 High layers Integration

All the video streaming applicative and control plane components are integrated on an "Adaptive Video Streaming Testbed" at CRAT's premises for validation of their correct inter-working as well as the preparation and validation of the Virtual Machines holding them and to be further delivered for integration on the "Connectivity Testbeds" (Terrestrial at CEA's and Satellite at FhG IIS')

3.3.1 Multi-connectivity Testbed integration strategy

The final stage of the integration process consists in building up the complete EU Testbed in order to be ready for the testing phase as described in §4.

Once validated the preliminary integration steps for the low layers (§3.2) and the high layers (§3.3), we identify three main testbed building blocks that have to be progressively merged:

- two single connectivity testbeds:
  - The Terrestrial Low Layers Testbed that mainly aimed at validating the Satellite MAC and Physical Layers in the Traffic Plane as well as the correct interfacing with cRRM in the Control Plane.
  - The Satellite Low Layers Testbed that mainly aimed at validating the Terrestrial MAC and Physical Layers in the Traffic Plane as well as the correct interfacing with cRRM in the Control Plane.
- The Adaptive Video Streaming Testbed that aimed at validating the adaptive video streaming management components and their interfacing with the Net Traffic Flow Controller in the Traffic Plane (for any other service than adaptive video streaming) and the cRRM in the Control Plane.

At this stage, these three Testbeds can be considered as standalone and fully operative. Therefore, the full testbed integration could then be achieved and validated through the following incremental integration steps leading to the final European Testbed configuration.

With respect to schedule and hardware shipping constraints, it was decided to perform the final phase of incremental integration and validation starting from the Terrestrial Low Layers Testbed at CEA's premises in Grenoble and to following the steps described here-after:

1) Terrestrial Connectivity Testbed:
   - Installation of the Virtual Machines (VMs) holding all the Adaptive Video Streaming Testbed components:
     - VM1 on UE side
     - VM2 on gNB/Network side
   - End-to-end service validation (at least the adaptive video streaming service)
2) Establish a VPN connection between FhG IIS site in Erlangen and CEA site in Grenoble
3) Multi-connectivity Testbed:
   - Replace the dummy Satellite dRRM by the remote real Satellite dRRM with the high layer stack

- Setup a video stream and verify the correct reception and expected performances are confirmed, at the end user side
- Channel dynamicity
  - ON/OFF model
    - Simulate an outage on the TERR link and verify that the communication is switched on the SAT link (terrestrial SNR going instantaneously to 0 (minus infinity dB)
    - Simulate an outage on the SAT link and verify that the communication is switched on the TERR link (terrestrial SNR going instantaneously to 0 (minus infinity dB)
  - Progressive variations of satellite and terrestrials SNR channels: the gain of the channel can be configured in an input file in the channel emulators.

Once all steps are successfully completed, the integration of the full EU Testbed can be considered done, see Figure 36.

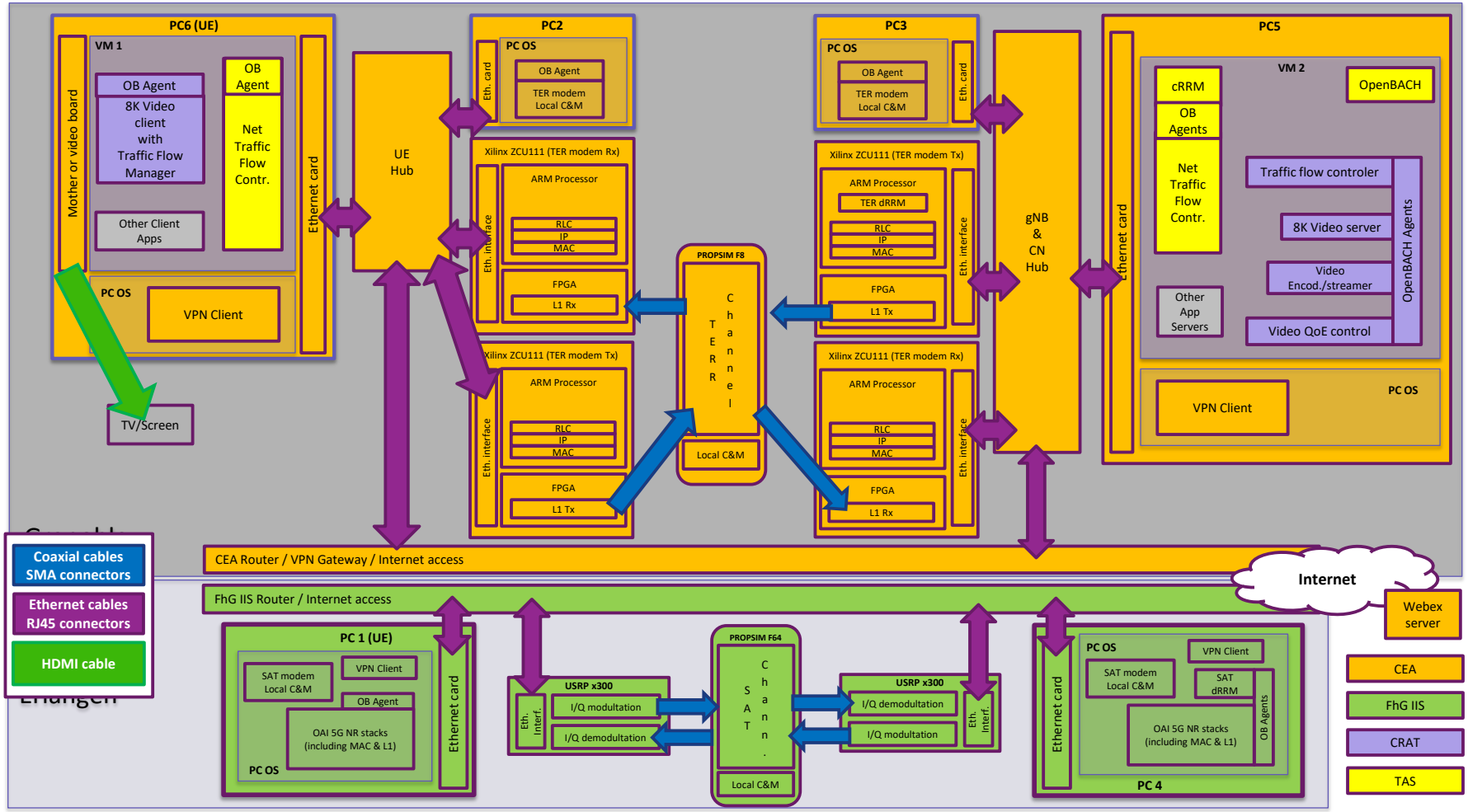


Figure 36. Multi-Connectivity Testbed architecture

3.3.2 Multi-connectivity Testbed validation status

Once the three testbeds described above validated, the main step was to establish the VPN to connect them. As specified in §1, with no router available in Erlangen, the only way to establish a link was the use of a VPN server in CEA's premises and of a VPN client at FhG's. This solution nevertheless required bypassing the security rules at CEA, i.e. obtaining a derogation from the IT department. This unfortunately revealed impossible after many weeks of try.

4 Testbed tests & measurements

Table 23 shows the incremental tests that were planned to get to the validation of the full testbed. Due to the absence of connectivity between Erlangen and Grenoble (see §1 and §3.3.2), only a subset of these tests could have been realized. The details for these test can be found in §4.3.

4.1 List of planned tests

In the table below, the first subset of tests, 1-7, corresponds to case completing unitary test, but with several or all components of the testbed. The second subset, 8-11, corresponds to the most advanced "cases" that can be mapped to D2.4 Use Cases and scenarios.

Some scenarios that could not have been tested can anyway be assessed during the EU trial platform phase to come. Due to the short time available for this phase (the presence of all partners is required), not all scenarios will be tested. See the "Comment" column.

Table 23. List of planned tests

D5.2 Scenario	Description	Mapping to D2.4 scenarios and Use Cases	Comment
Scenario 1	Basic Terrestrial Connectivity	N/A	Validated
Scenario 2	Basic SatCom Connectivity	N/A	Validated
Scenario 3	Basic 8K video streaming	N/A	Validated for the TERR testbed
Scenario 4	Basic stringent QoS transmission	N/A	Not assessed. To be assessed in the TERR part of the EU trial.
Scenario 5	Basic Multi Connectivity	N/A	Not assessed.
Scenario 6	Realistic terrestrial channel with degradations	N/A	Validated
Scenario 7	Realistic SatCom channel with degradations	N/A	Validated
Scenario 8	Realistic multi-connectivity, no handover, single application	Use Case 2 : Simultaneous satellite and cellular multi-connectivity / Scenario 1 : simultaneous cellular and satellite access with enhanced total throughput	To be assessed in the EU trial with video streaming.
Scenario 9	Realistic multi-connectivity, handover, single application	Use Case 1 : Broadband moving hotspot network / Scenario 1 Beam Switching for Vehicular	To be assessed in the EU trial with video streaming.
		Use Case 3 : Public Safety / Scenario 1 : Unpredictable Traffic Switching	Non-priority test. Possibly assessed in the EU trial with stringent QoS service (e.g. camera).
Scenario 10	Realistic multi-connectivity, no handover, multiple applications	Use Case 2 : Simultaneous satellite and cellular multi-connectivity / Scenario 2 : simultaneous satellite non-stringent QoS services and cellular stringent QoS services delivery to UE	To be assessed in the EU trial with video streaming via SAT and/or TERR and live camera or VR game via TERR.

D5.2 Scenario	Description	Mapping to D2.4 scenarios and Use Cases	Comment
Scenario 11	Realistic multi-connectivity, seamless handover, single application	Use Case 3 : Public Safety / Scenario 2 : predictable Traffic Switching	Non-priority test. Possibly assessed in the EU trial with stringent QoS traffic duplication.

4.2 KPIs

The following KPIs were considered as relevant for the testbed.

Table 24. Testbed infrastructure KPIs

KPI	Description
MODCOD / MCS	Instantaneous MODCOD / MCS used, as a function of time.
Measured SINR	Instantaneous measured SINR measured by UE and/or gNB on active access(s)((satellite and/or terrestrial)
Decoding Errors	Any indication of decoding errors (CRC fails etc.)
Estimated QoE/ QoS	Video quality (see Table 25)
Radio link flow decision	Real-time selection of RAN per flow Satellite-access only Terrestrial--access only Simultaneous Satellite access / terrestrial-access

4.2.1 Video Streaming KPIs

The KPIs specific to video are:

Table 25. Video Streaming KPIs

KPI	Description
UE Video Buffer	Video Buffer length at UE (video player) side
Dropped Video Frames	Number of dropped video frames at UE side during video streaming
Total Video Frames	Total number of video frames received at UE side
Bandwidth	Estimated video streaming bandwidth at CN side
Bitrate sent	Video streaming bitrate sent by video server at CN
Number of bytes sent/received/retransmitted	Number of bytes sent, received, and retransmitted by video server at CN side
Congestion window size	TCP congestion window at video server side
Video Level	Indicator of what video quality is actually sent by the video server and played by the video client (ranging from 1 – 360p, to 6 – 8K)
Video server queue	Queue length at video server side
Re-buffering number	Number of re-buffering events happened during video streaming

KPI	Description
Round-trip-time	RTT measure including real-time values, standard deviation and minimum values
Screen measures	Screen height, width and size of the virtual screen at UE side

4.2.2 Other Application-level KPI

Applicative-level KPIs	Depends on each application/particular service Basic: failed / not failed (when relevant, i.e. file transfer completed or not) RTT or 1-way latency For tests with File Transfer: download time For tests with QoS stringent service: perceived quality ; packet loss.

4.3 Detailed tests description and validation status

For each of the scenarios of Table 23, a high level description of the objectives is provided below. The required input parameters and main expected KPIs are also detailed. This is done even if the scenario could not have been tested.

4.3.1 Scenario 1: Basic Terrestrial Connectivity

Status: validated.

High-level description and objectives:

- Ensure end-to-end connectivity is available through terrestrial path only
- Simple application test (e.g. UDP data transfer, with iperf3 client and server)
- Run successively two tests in both directions (forward and return links)

Input Parameters

- Maximum UE SINR
- Constant single path channel
- Different bandwidth and MCS configurations (see §3.1.1.2)

Main KPIs checked:

- Correct synchronization
- Applicative transfer throughput
- No packet loss (CRC error rate=0)
- Coherent measured SINR

4.3.2 Scenario 2: Basic SatCom Connectivity

Status: validated

High-level description and objectives:

- Ensure that end-to-end connectivity is available through simulated SatCom path
- Simple application test (i.e. ping test)
- Run successively two tests in both directions (forward and return links)

The test consisted in running the SAT OAI modems with a simulated long delay in the RFSIM mode, which means the RF front-end and the propagation delay effects of the GEO SAT channel are simulated in the OAI code. The selected channel parameters were:

- SAT channel one-way propagation delay: 240 ms
- Simulation centre frequency: 3619.08 MHz (TDD)
- 50 dB attenuation DL/UL

During this test a ping was run in both DL and UL (i.e. forward and return links) to verify proper functioning of the IP data traffic. KPIs were not measured since they are not relevant in the RFSIM mode.

The test was successful and validated basic connectivity of the SAT modems. Since the gNB has the dRRM module integrated, which act as an interface with the cRRM, the integration of the SAT modems with the cRRM module was also validated. Figure 37 shows the OAI scope during the performance of the tests in this scenario.

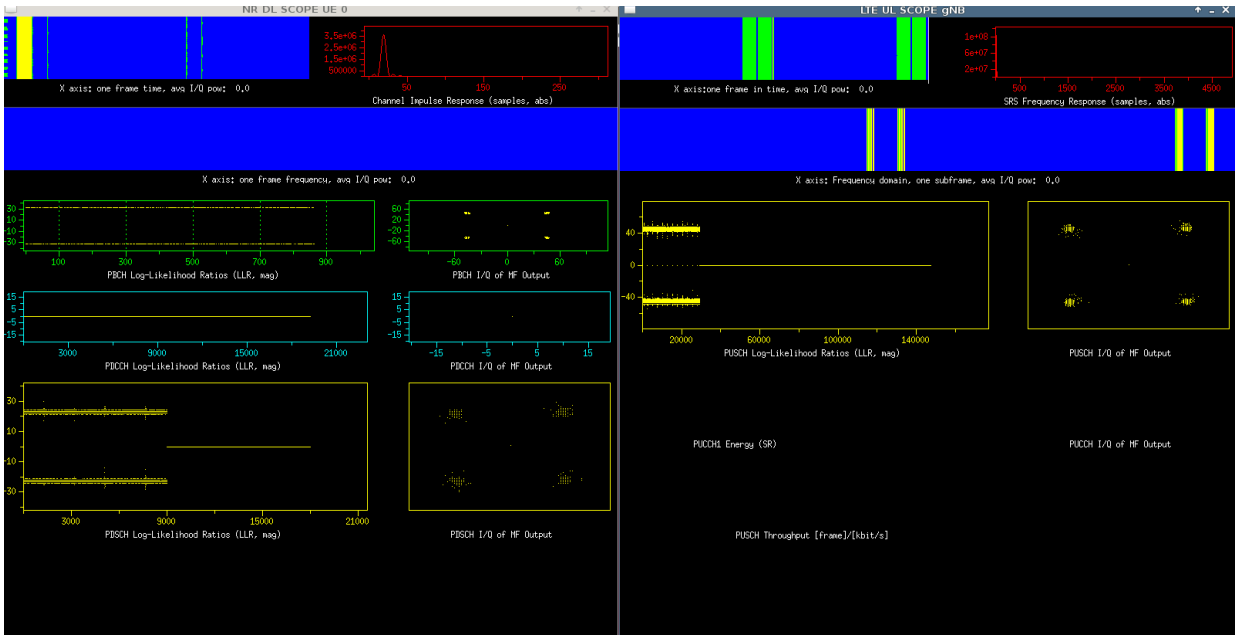


Figure 37. OAI scope of the SAT modems during Scenario 2 test

4.3.3 Scenario 3: Basic 8K video streaming

Status: validated in the TERR testbed and in the high-layer testbed.

High-level description and objectives:

- Work on the specification of the video service to be used (which encoder/decoder, which protocols, which target data rate quality, …)
- Check 8K video can be sent and received over the testbed
- Validate the multipath control algorithm on top of MPTCP
- Master the software input and output 8K video streaming parameters

Input Parameters:

- Video parameters
- cRRM channel capacity estimation
- Other parameters as per Scenario 1

Main KPIs checked:

- Video KPIs (see §4.3.6), especially “level” of QoE, which was checked to be 6 (i.e. 8k).
- MPTCP KPIs in response to channels capacity variations reported by cRRM

4.3.4 Scenario 4: Basic camera transmission

Status: not assessed

High-level description and objectives:

- Check QoS stringent service can be setup over a single access (e.g. terrestrial)
- Master the input and output camera parameters

Required Parameters:

- Parameters as per Scenario 1
- camera parameters

Expected KPIs:

- Minimal latency (no interruption of the real-time camera transmission)

4.3.5 Scenario 5 : Basic Multi-Connectivity

Status: not assessed

High-level description and objective:

- Ensure end-to-end connectivity is available through terrestrial path and radios
- Check that service are correctly transported over the preferred Access
- Run test for two applications one after the other

Required Parameters:

- Other parameters as per Scenario 1, 2

Main Expected KPIs:

- Applicative transfer duration / throughput
- Tx and Rx losses (expectation: no losses)
- Measured SINR per access
- Estimated QoE/ QoS
- RAN flow decision (expectation: conformance to preferred RAN mapping)

4.3.6 Scenario 6 : Realistic terrestrial channel with degradations

Status: validated

High-level description and objectives:

- Apply advanced 3GPP channel model for terrestrial access and check impact on application
- Check synchronism between forward / return channels
- Check the ability for the testbed to ingest this model and applies it at relevant execution time
- Check connectivity is provided under expected performances and variations related to the channels characteristics

Input Parameters:

- 3GPP Terrestrial Channel Model in both UL and DL
- Frequency Ranges
    - Test low band (<6 GHz)
    - Then test high band (>24 GHz)

Main KPIs checked:

- Video KPIs, see Figure 39
- Measured SINR: consistent with shadowing variation
- MCS selection by cRRM: consistent with shadowing variation
- Decoding errors: null CRC error rate with adequate MCS

The TERR testbed (modems, MAC and connection with cRRM and Traffic control) has been tested and validated with the transmission of the video streamer described in §2.2.1.1.5. For this test, the selected channel is 3GPP Evolved Pedestrian A (EPA), see Table 26. Maximum Doppler frequency is 5 Hz and user speed is 0.68 m/s.

Table 26. EPA delay profile

Excess tap delay (ns)	Relative power (dB)
0	0
30	-1.0
70	-2.0
90	-3.0
110	-8.0
190	-17.2
410	-20.8

Figure 38 shows a snapshot of the channel realization from the channel emulator. In this scenario, additional shadowing is set with a gain changing over the time, blue curve. The maximum shadowing is 20 dB.

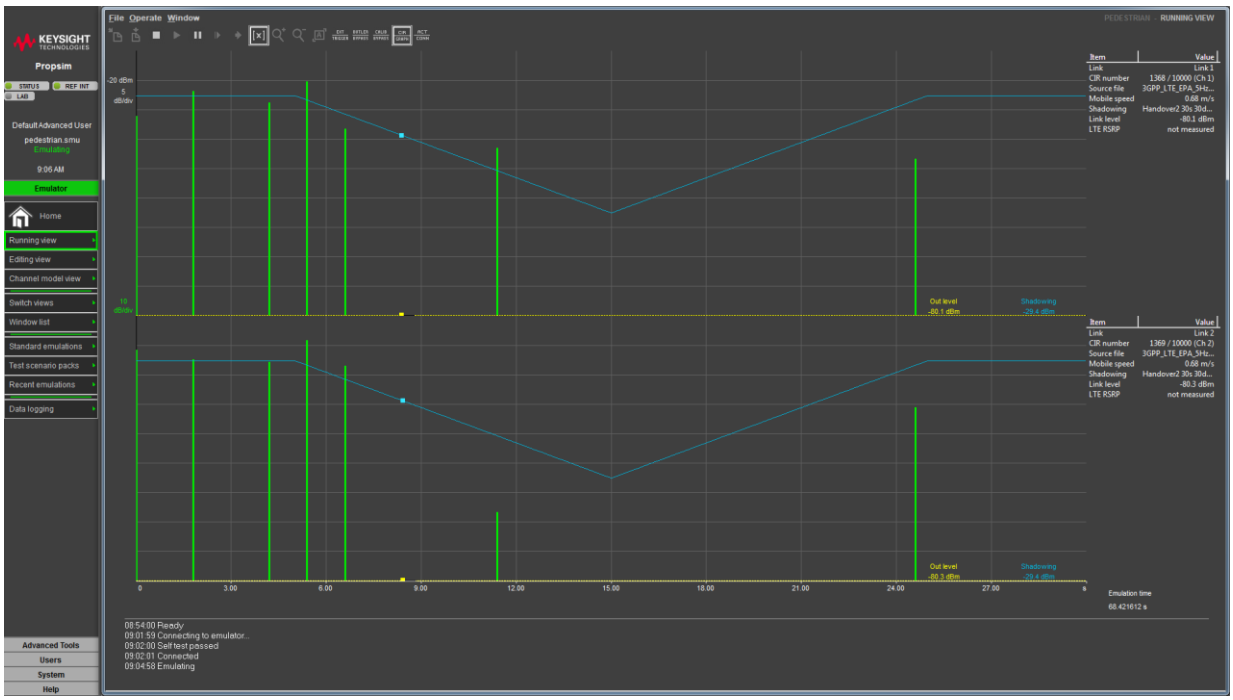


Figure 38. Simulated channel for TERR testbed validation (DL and UL)

Here, 6 terrestrial sub-bands (over 12) are active, that corresponds to the bottom-right configuration on Figure 29. The carrier frequency (IF) is 1 GHz.

When deep shadowing occurs, the TERR modems are losing synchronization. The modems automatically synchronize again when the shadowing level goes above a given threshold. Nevertheless, as show on Figure 39, these traffic interruptions do not affect the perceived video quality for the user (the highest level of quality is reached): when synchronization is OK the download throughput is high enough to fill the buffer.

Figure 39. KPIs for a video transmission on the TERR link with EPA channel (varying gain)

4.3.7 Scenario 7: Realistic SatCom channel with degradations

Status : validated

High-level description and objectives:

- Ensure that end-to-end connectivity is available through realistic GEO SatCom path
- Simple application test (i.e. UDP data transfer)
- Run successively two tests in both directions (forward and return links)

The test consisted in running the SAT OAI modems with an emulated GEO Satellite channel. In the test setup the SAT modems were connected with the PROPSIM F64 channel emulator. The selected channel parameters were:

- SAT channel one-way propagation delay: 240 ms
- RF and simulation centre frequency: 2169.08 MHz DL, 1769.08 MHz UL
- 20 dB attenuation DL/UL

During this test an UDP Iperf throughput test was performed both in DL and UL (i.e. forward and return links). Different MCS were tested and the performance of the SAT modems proved to be stable up to MCS 16 of table 5.1.3.1-1 of [7]. The KPIs outputs are reported in Table 27.

Table 27. KPIs outputs of basic SatCom connectivity test

KPIs	Result
Max throughput	DL: 3 Mbps UL: 14 Mbps
Tx and Rx losses	No losses
Measured SINR	DL: 45 dB

Figure 40 shows the OAI Scope of the SAT modems during a 16 QAM transmission (DL and UL) over an emulated GEO satellite channel.

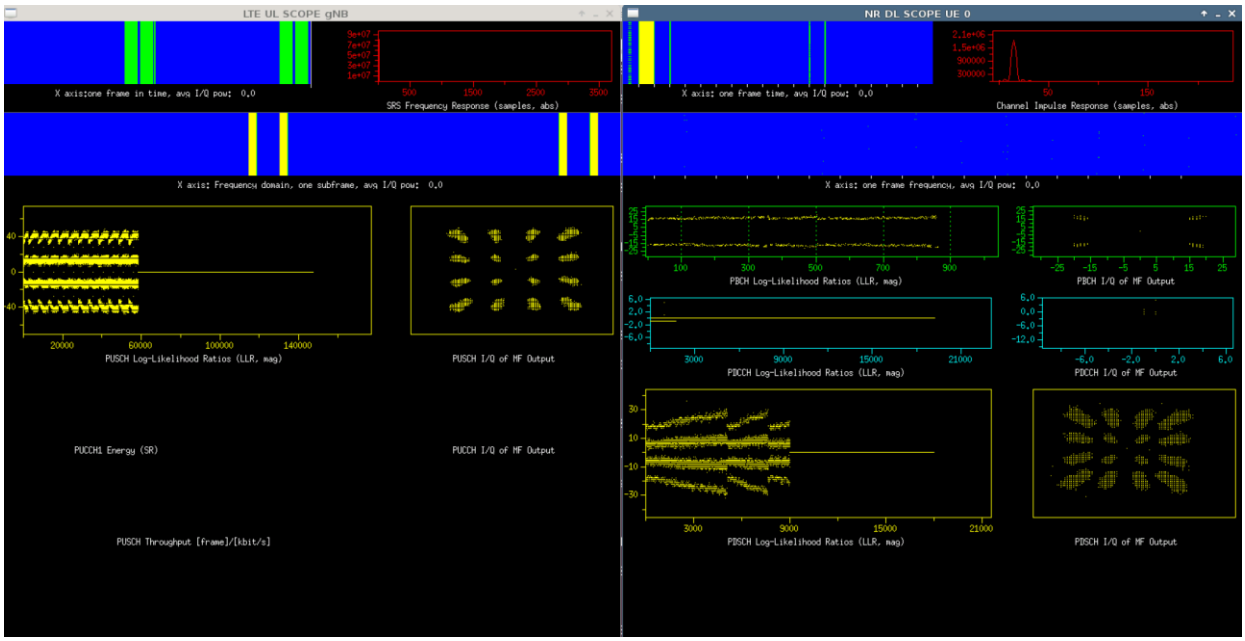


Figure 40. Satellite connectivity shown in OAI scopes with 16 QAM transmission (DL/UL)

4.3.8 Scenario 8: Realistic multi-connectivity, no handover, single application

Status: To be assessed in the EU trial with video streaming.

High-level description and objective:

- Check multi-connectivity when UE (fixed or mobile) is within the coverage of both terrestrial base station and satellite macro cell.

Required Parameters:

- 3GPP terrestrial channel
- Advanced satellite Channel models
- Other parameters as per Scenario 5
- Backhaul connectivity is model by 2 single UDP Flows (one for each direction) with a sending rate chosen accordingly to the known resource

Main Expected KPIs:

- Applicative transfer duration / throughput
- Tx and Rx losses (expectation: no losses)
- Measured SINR

4.3.9 Scenario 9 : Realistic multi-connectivity, handover, single application

Status:

With video streaming: to be assessed in the EU trial.
With stringent QoS service (e.g. camera): non-priority test. Possibly assessed in the EU trial.

High-level description and objective:

- Build 2 independent channel models (i.e. attenuation profiles that needs to be synched) showing a full seamless handover in 3 stages:
  - UE is within 1 cell served by terrestrial access only
  - From there, UE moves in a cell area where satellite access is available
  - From there, UE moves to an area when only the satellite access is possible
- Check the multi-connectivity process

Required Parameters

- 3GPP terrestrial channel
- Advanced satellite Channel model
- Other parameters as per Scenario 8

Main Expected KPIs

- Applicative transfer duration / throughput
- Tx and Rx losses (expectation: no losses)
- Measured SINR
- RAN flow decisions
- Interruption time

4.3.10 Scenario 10 : Realistic multi-connectivity, no handover, multiple applications

Status: To be assessed in the EU trial with video streaming via SAT and live camera via TERR.

High-level description and objective:

- Verify how the system behaves and allocates the flows with realistic changing channel conditions

Required Parameters:

- 3GPP terrestrial channel
- Advanced satellite Channel / Doppler / Time models
- Other parameters as per Scenario 8

Main Expected KPIs:

- Applicative transfer duration / throughput
- Tx and Rx losses (expectation: no losses)
- Measured SINR
- RAN flow decisions

4.3.11 Scenario 11: Realistic multi-connectivity, handover, multiple applications

Status: non-priority test. Possibly assessed in the EU trial with stringent QoS traffic duplication.

High-level description and objective:

- Ultimate test combining all the features
- Check Quality of 8K Video streaming is sustainable over time provided that the overall resources are available

Required Parameters

- Channel Model: ideal channel (terrestrial)
- Channel bandwidth + maximal CQI (SNRs)

Main Expected KPIs

- Applicative transfer duration / throughput / QoE
- Tx and Rx losses (expectation: no losses)
- Measured SINR
- RAN flow decisions
- Interruption time

5 Conclusion

This document compiles the work that has been realized on the EU testbed. Even if the connection between the TERR testbed in Grenoble and the SAT testbed in Erlangen was not set (see §1 and §3.3.2), all the sub-systems were validated: SAT testbed, TERR testbed and adaptive video testbed. These testbeds allowed us to run tests that achieved, or partially achieved, the following objectives:

- Objective 1, “Implement a 5G cellular mmW radio access” (part of objective 5.1).

  The TERR testbed implements of 5G NR compatible waveform on FPGA. A MAC has been designed and demonstration has been provided with fragmented spectrum at 1 GHz carrier frequency. mmW demonstration is planned for the final demonstration. See §4.3.6.

- Objective 2, “Demonstrate feasibility of New Radio based mmW satellite access for providing broadband and reliable 5G services” (part of objective 5.1).

  The performance of the SAT modems have been validated in the SAT Platform of the UE Testbed through the transmission of data traffic over a GEO Satellite channel emulated with the channel emulator. See §4.3.2 and §4.3.7.

- Objective 3, “Implement multi-connectivity support and integrate 5G cellular access and satellite access systems along with its function testing” (part of objective 5.1).

  Traffic flow controller and manager have been implemented and validated in a dedicated testbed. See §3.1.6.

- Objective 4, “Demonstrate the PoC of multiple access systems in shared spectrum context supporting multi-connectivity at a key event” (part of objectives 5.2 and 5.3).

  The testbed described in this document, which is made of TERR testbed, SAT testbed, traffic control and cRRM, is a step toward the EU trial setup first and then to the final intercontinental trial.

The next steps are the implementation of the EU trial platform and the final intercontinental demonstration. These two steps require all the hardware described in this document to be located in the same place (CEA Grenoble), so as the presence of all involved partners. Due to these constraints, little time will be available (i.e. few days). The initial plan was to allocate this time to the final demonstration only, but we’ll use some time to validate all the scenario we can, see Table 23.

6 References

[1] "Preliminary document of 5G-ALLSTAR architecture, API and interface specifications," 2019. [Online]. Available: https://5g-allstar.eu/results/deliverables/.

[2] "D5.1 Specification of the European testbed of 5G cellular and satellite access networks," 2019. [Online]. Available: https://5g-allstar.eu/results/deliverables/.

[3] "Keysight Technologies," [Online]. Available: https://www.keysight.com.

[4] 3GPP, "TR 38.901 Study on channel model for frequencies from 0.5 to 100 GHz (Release 14)," 3GPP, 2018.

[5] 3GPP, "TR 38.811 Study on New Radio (NR) to support non terrestrial networks (Release 15)," 2019.

[6] "D3.3 Interference Mitigation Techniques," 2020.

[7] 3GPP, "Physical layer procedures for data," 3GPP, 2020.

[8] "OpenAirInterface Software Alliance," Eurecom, 2021. [Online]. Available: https://www.openairinterface.org/.

[9] 3GPP, "TR 38.821: Solutions for NR to support non-terrestrial networks (NTN) (Release 16)," 3GPP, 2020.

[10] 3GPP, "TS 38.213: Physical layer procedures for control (Release 16)," 3GPP, 2020.

[11] 3GPP, "TS 38.214: Physical layer procedures for data (Release 16)," 3GPP, 2020.

[12] F. Kaltenberger, T. Schlichter, T. Heyn, G. Casati, F. Völk, R. T. Schwarz and A. Knopp, "Building a 5G Non-Terrestrial Network using OpenAirInterface Open-source Software," in *Joint EuCNC & 6G Summit*, Porto, 2021.

[13] Fraunhofer IIS, "First software-defined 5G New Radio demonstration over GEO satellite," Fraunhofer IIS, 12 March 2021. [Online]. Available: https://www.iis.fraunhofer.de/en/pr/2021/20210312_5G_new_radio.html.

[14] "https://wiki.net4sat.org/doku.php?id=openbach:manuals:2.x:user_manual:basic_user_web_interface:index&s[]=project," [Online].

[15] "https://wiki.net4sat.org/doku.php?id=openbach:manuals:2.x:administrator_manual:web_interface:index#manage_jobs," [Online].

[16] "https://wiki.net4sat.org/doku.php?id=openbach:manuals:2.x:developer_manual:openbach_api:index," [Online].

7 ANNEX

7.1 Working with cRRM

This section details practical things to know for using cRRM.

7.1.1 Implementation

- Modems (gNB-DU and UE) have to be reconfigured periodically, from inputs describing the channel bandwidth value + other static parameters if needed.
- when multiple links are available, the traffic policies set by Traffic Flow controller (e.g. which CoS is served first) complete the work.

You can get source code, debian package and documentation from TAS/Viveris. cRRM is delivered as a ZIP file 5gallstar-cRRM-delivery.zip whose content looks like that: 5gallstar-cRRM-delivery.

7.1.2 Requirements

5gallstar-cRRM and 5gallstar-cRRM-dumy-* are pure python programs. They are distributed in a debian package '5gallstar-cRRM_1.0.0_amd64.deb' compatible with Ubuntu version 16.04 TLS (xenial). The package has been built for running on x86 and x64 processors.

For any other distribution/version/architecture, you may need to build it yourself from the source. If you don't fall in this case, skip the rest of this section.

Copy the zip file to the targeted platform. If the 'unzip' command isn't already installed on your system then run:

```
$ sudo apt-get install unzip
```

After installing the unzip utility, run these commands to extract and build package for your specific Linux platform:

```
$ unzip 5gallstar-cRRM-delivery.zip
$ cd 5gallstar-cRRM-delivery/5gallstar-cRRM
$ sudo dpkg-buildpackage -us -uc -rfakeroot
$ cd ..
```

7.1.3 Installation

You can install 5gallstar-cRRM_1.0.0_amd64.deb package in command line using dpkg like this:

```
$ sudo dpkg -i 5gallstar-cRRM_1.0.0_amd64.deb
```

If you get any dependency errors, you can use the following apt command to fix it:

```
$ sudo apt install -f
```

It will install package dependencies.

During installation, configuration folder (/etc/5gallstar-cRRM/) and logs folder (/etc/var/log/5gallstar-cRRM/) are created. Executable scripts are also deployed to the appropriate directory depending on your debian distribution and required python packages are installed in your python installation.

Once the installation is completed, you can run cRRM and dummy components easily from command line terminal like any Debian command. If you want to change the default configuration you would edit configuration files in configuration folder.

7.1.4 Run

5gallstar-cRRM includes a manual page listing all of the command-line options from a terminal.

```
$ sudo 5gallstar-cRRM  -h

Usage: 5gallstar-cRRM [-h] [-c CONF] [-o] [-l LOG] [-q] [-v] [-d]

5gallstar-cRRM - This script is the main piece of software of the cRRM component.

optional arguments:

-h, --help            Show this help message and exit

-c CONF, --conf CONF  Path to the configuration folder (default: /etc/5gallstar-
cRRM/)

-o, --openbach        Send KPIs to the OpenBACH Collector (default: False)

-l LOG, --log LOG     Path to the log file. (default: /var/log/5gallstar-
cRRM/cRRM.log)

-q, --quiet           Stop printing logs to the console (default: False)

-v, --verbose         Print more informations (default: False)

-d, --debug           Print debug (default: False)
```

All arguments are optional, so you can easily start it in verbose mode invoking:

```
$ sudo 5gallstar-cRRM -v
```

The output:

```
5gallstar-cRRM - INFO  - run dRRM service on 0.0.0.0:5050 [5gallstar-cRRM:147]

5gallstar-cRRM - INFO  - run AppTrfFlowCtlr service on 0.0.0.0:5051 [5gallstar-
cRRM:196]

5gallstar-cRRM - INFO  - run NetTrfFlowCtlr service on 0.0.0.0:5052 [5gallstar-
cRRM:241]

5gallstar-cRRM - INFO  - cRRM is ready !!! [5gallstar-cRRM:406]
```

A cRRM channel runs directly over a TCP/IP connection which shall always be initiated by the other component.

For each type of component (dRRM, application traffic flow controller and network traffic flow controller), a dedicated transport port number is reserved to connect to the cRRM.

7.1.4.1 Configuring 5gallstar-cRRM

The cRRM requires theses configuration files to run correctly:

deamon.conf	cRRM communicates with the other components via a dedicated TCP channel. This file defines ip/port to use to bind/connect sockets to.
core.conf	This file defines options/parameters of the algorithms/schemes for resource (re)allocations that are supported by the cRRM.

system.conf	This file defines the initial configuration of radio resource parameters in both time and frequency domain and for each network access.
MODCOD	This file defines for each modem type the list of supported MODCOD, spectrum efficiently and SINR threshold satisfying the targeted BLER

See each configuration file for further details about supported parameters and options.

7.1.4.2 Running 5gallstar-cRRM manually

When dummy modules are required, cRRM must be run first before running any dummy components.

5gallstar-cRRM includes a manual page listing all of the command-line options from a terminal, accessible through the command:

```
$ sudo 5gallstar-cRRM  -h
```

All arguments are optional, so you can easily start it in verbose mode invoking:

```
$ sudo 5gallstar-cRRM -v
```

The initial output logs are the following lines:

```
5gallstar-cRRM - INFO  - run dRRM service on 0.0.0.0:5050
[5gallstar-cRRM:147]

5gallstar-cRRM - INFO  - run AppTrfFlowCtlr service on 0.0.0.0:5051
[5gallstar-cRRM:196]

5gallstar-cRRM - INFO  - run NetTrfFlowCtlr service on 0.0.0.0:5052
[5gallstar-cRRM:241]

5gallstar-cRRM - INFO  - cRRM is ready !!!
[5gallstar-cRRM:406]
```

After that the run dummy components can be run.

The dummy components can be run with the

Table 28. Commands lines for the dummy components

Component	Command	Command line
5gallstar-dummy-dRRM	Help	`$ sudo 5gallstar-dummy-dRRM  -h`
	Run SAT	`$ sudo 5gallstar-dummy-dRRM -t sat -v`
	Run TERR	`$ sudo 5gallstar-dummy-dRRM -t ter -v`
5gallstar-dummy-apptfc	Help	`$ sudo 5gallstar-dummy-apptfc -h`
	Run	`$ sudo 5gallstar-dummy-apptfc -t net -v`

7.1.4.3 OpenBACH Framework

All components can be run automatically by using the OpenBACH framework. The following OpenBACH components shall be installed in the platform. This includes:

- Controller
- Collector
- Agents

OpenBACH can be used via a Web Interface or via a Command Line Interface. To configure your platform and CLI to build and execute scenarios. Note that, you can manually edit basic scenarios from Web Interface. For more complex scenarios, it is recommended to use CLI. The platform can be configured as explained at [14].

The steps to implement the OpenBACH framework are the following:

1. Create a new empty project in your controller from OpenBACH Web Interface
2. Add entities to your platform and associate each of them to an installed agent
3. Add required 5G-ALLSTAR jobs to controller as external jobs, then install them on desired entities/agents as described here at [15]

For each component, an OpenBACH job has been developed to be run on an entity. The source code is available in the 'openbach_jobs' folder:

- cRRM : this job runs cRRM
- dummy-dRRM: this job runs dummy dRRM component in satellite or terrestrial mode
- dummy-apptfc: this jobs runs dummy application traffic flow controller
- dummy-nettfc: this job dummy network traffic flow controller
- kpi_forwarder: this job sends received KPIs from cRRM to OpenBACH collector

Once installation and configuration are completed, scenarios can be build and run from OpenBACH API in CLI with custom OpenBACH scripts located in 'openbach_scenarios' folder and reported in Table 29.

Table 29. Scripts for OpenBACH

Script	Description
rt_5gallstar.py	It is a python module based on OpenBACH APIs (scenario_builder) in charge to build and configure an OpenBACH scenario in function of the specified arguments. This scenario includes one or more sub scenarios each of them consisting to a configured cRRM job to start first, then all or just a part of other jobs in function of specified arguments.
executor_rt_5gallstar.py	It is executable script to be run. It calls the above module to build scenario, then push it to the OpenBACH controller to be run and wait for scenario instance to finish.

These scripts can be run on any machine that is connected to the controller. So, the running machine can be the controller (this is recommended) or any agent. Any way, it must be a computer running Ubuntu 16.04 TLS, with python 3.5 or newer and OpenBACH-API installed to interact with the controller.

Please refer to [16] to install openbach-apis, setup your environment for using executors and API.

Once cloned the openbach-extra repository which is installed by default with OpenBACH controller. Run these commands to install 5gallstar scenarios:

1) cp openbach_scenarios/rt_5gallstar.py /path/to/openbach-extra/apis/scenario_builder/scenarios
2) cp openbach_scenarios/executor_rt_5gallstar.py /path/to/openbach-extra/executors/reference/
3) cd /path/to/openbach-extra/apis/auditorium_scripts

Now you can use this command to show what arguments are expected to run the executor:

```
$ PYTHONPATH=~/path/to/openbach-extra/apis  python3
../../executors/references/executor_rt_5gallstar.py -h
```

You can run the script skipping backend arguments since they will be found in the 'controller' file that you had created and edited.

Example commands to enable cRRM and satellite dRRM:

```
PYTHONPATH=~/path/to/openbach-extra/apis/ python3
../../executors/references/executor_rt_5gallstar.py 'YOUR_PROJECT_NAME' --
cRRM_entity YOUR_ENTITY_NAME --dRRM_sat_entity YOUR_ENTITY_NAME  -o run
```

To Enable cRRM, both satellite and terrestrial dRRM, and kpi forwarder:

```
PYTHONPATH=~/path/to/openbach-extra/apis/ python3
../../executors/references/executor_rt_5gallstar.py 'YOUR_PROJECT_NAME' --
cRRM_entity YOUR_ENTITY_NAME --dRRM_sat_entity YOUR_ENTITY_NAME --dRRM_sat_entity
YOUR_ENTITY_NAME --kpi_entity YOUR_ENTITY_NAME -o run
```

Where

- YOUR_PROJECT_NAME refers to the name of your OpenBACH project.
- YOUR_ENTITY_NAME must be replaced with the name of the Openbach entity on which you installed the job that is associated to the component to run. (Note that all components can be run on the same entity as in the demo)

From OpenBACH Controller Web Interface, you can see that the '5GALLSTAR' scenario has been correctly created and is running.

From a chronograph dashboard, you can monitor scenario's KPIs to check if everything is working as expected.

7.1.5 Integration of 5G-ALLSTAR modules with cRRM

A cRRM channel is the interface that connects each component to the cRRM. Through this interface, the cRRM configures and manages the dRRMs, receives measurement reports from dRRMs, sends throughput capacity to the application traffic flow controller and traffic steering instructions to the network traffic flow controller.

A cRRM channel runs directly over a TCP/IP connection which MUST always be initiated by the other component.

For each type of component (dRRM, application traffic flow controller and network traffic flow controller), a dedicated transport port number is reserved to connect to the cRRM.

All messages exchanged must comply with JSON according to the associated JSON schema. The message types used by the cRRM are described in Table 30.

Table 30. cRRM message types

Message type	Description
gNBDU_dRRM_configuration_request	The dRRM requests configuration from cRRM by sending a configuration request upon establishment of the cRRM channel.
gNBDU_dRRM_configuration	The cRRM uses this message to reply to a configuration request from dRRM.
gNBDU_dRRM_configuration_confirm	The dRRM sends a configuration confirm to confirm the configuration is complete.
UE_measurement_report	It is an asynchronous message sends by the dRRM to report measurement from the UE or to inform the cRRM of the link status change.
gNBDU_dRRM_allocation	Message sends by the cRRM to reallocate resources to the dRRM.
gNBCU_cRRM_throughput	Asynchronous message sends by the cRRM to the application traffic flow controller to report throughput capacity on each available access

Each message shall be compliant with associated JSON schema and encoded in UTF-8.

JSON schemas are defined in the file 'schema.txt' that is located in the documentation folder.

Since messages can be of variable length and are streamed over a single TCP connection, there are a need to identify where one JSON encoded message ends and the next begins. (The concatenation of two JSON messages does not produce a valid JSON message).

To achieve this, each message MUST be sent prefixed with its length which is encoded into 4 bytes (uint64). Length-prefixed JSON is well-suited for TCP connections.

7.2 SAT Modems installation and run

7.2.1 Installation

The OAI code is available at the public repository https://gitlab.eurecom.fr/oai/openairinterface5g.
The steps to check out the source code locally are:

```
$ git clone https://gitlab.eurecom.fr/oai/openairinterface5g.git
```

Within the framework of 5G-ALLSTAR, FhG IIS deploys an adapted implementation of the develop branch into the EU testbed.

7.2.2 Build & Run

The build script build_oai located at the root of the `openairinterface5g/cmake_targets` directory is developed to build the OAI binaries (executables, shared libraries) for different hardware platforms, and use cases.

To build the OAI modems executables, the steps to follow on each machine are the following:

1) Move to the cmake_targets folder:

```
$ cd <your oai installation directory>/openairinterface5g/cmake_targets
```

2) Source the OAI environment variables:

```
$ source oaienv
```

3) Run the build_oai script:

```
$ ./build_oai -w USRP --nrUE --gNB
```

The `-I` option is to install pre-requisites, you only need it the first time you build the softmodem or when some OAI dependencies have changed.

The `-w` option is to select the radio head support you want to include in your build. Within the scope of this project the radio head is the USRP. Radio head support is provided via a shared library, which is called the "oai device" The build script creates a soft link from `liboai_device.so` to the true device which will be used at run-time (e.g. the USRP one is `liboai_usrpdevif.so`.

The following options are function-specific and shall match the relevant testbed PC:

- `--gNB` is to build the `nr-softmodem` executable and all required shared libraries
- `--nrUE` is to build the `nr-uesoftmodem` executable and all required shared libraries

After completing the build, the binaries are available in the `cmake_targets/ran_build/build` directory. A copy is also available in the `target/bin` directory, with all binaries suffixed by the 3GPP release number, e.g. .Rel15.

7.2.3 Configuration file of the OAI gNB

Table 31. Configuration file for the OAI gNB

Parameters	Range of values
ssb_SubcarrierOffset	1..23
pdsch_AntennaPorts	1
spCellConfigCommon: physCellId	0 - 503
downlinkConfigCommon	
frequencyInfoDL { dl_frequencyBand, absoluteFrequencySSB, dl_absoluteFrequencyPointA }	 FR1 Between 1.2 and 6 GHz
dl_subcarrierSpacing	30 kHz
dl_carrierBandwidth	106 (40 MHz, SCS 30KHz)
initialDownlinkBWP	
initialDLBWPlocationAndBandwidth	Range as per 3GPP TS
initialDLBWPsubcarrierSpacing	30
pdcch-ConfigCommon	
initialDLBWPcontrolResourceSetZero	0..15
initialDLBWPsearchSpaceZero	0..15
pdsch-ConfigCommon	
pdschTimeDomainAllocationList	
initialDLBWPk0_0	0..7
initialDLBWPmappingType_0	0, 1
initialDLBWPstartSymbolAndLength_0	0..127
initialDLBWPk0_1	0..7
initialDLBWPmappingType_1	0, 1
initialDLBWPstartSymbolAndLength_1	0..127
initialDLBWPk0_2	0..7
initialDLBWPmappingType_2	0, 1
initialDLBWPstartSymbolAndLength_2	0..127
initialDLBWPk0_3	0..7

initialDLBWPmappingType_3	0, 1
initialDLBWPstartSymbolAndLength_3	0..127
uplinkConfigCommon	
ul_frequencyBand	FR1 Between 1.2 and 6 GHz
scs-SpecificCarrierList	
ul_offstToCarrier	0..11
ul_subcarrierSpacing	30 kHz
ul_carrierBandwidth	106
pMax	20
initialUplinkBWP	
initialULBWPlocationAndBandwidth	Range as per 3GPP TS
initialULBWPsubcarrierSpacing	30 kHz
rach-ConfigCommon	
rach-ConfigGeneric	
prach_ConfigurationIndex	Range as per 3GPP TS
prach_msg1_FDM	Range as per 3GPP TS
prach_msg1_FrequencyStart	Range as per 3GPP TS
zeroCorrelationZoneConfig	Range as per 3GPP TS
preambleReceivedTargetPower	Range as per 3GPP TS
preambleTransMax	3…200
powerRampingStep	Range as per 3GPP TS
ra_ResponseWindow	1…80
ssb_perRACH_OccasionAndCB_Pream-blesPerSSB_PR	1..16
ssb_perRACH_OccasionAndCB_Pream-blesPerSSB	4:4:64
ra_ContentionResolutionTimer	Contention-free
rsrp_ThresholdSSB	19
prach_RootSequenceIndex_PR	139
prach_RootSequenceIndex	1
msg1_SubcarrierSpacing	30 kHz
restrictedSetConfig	0, unrestricted 1, restricted type A 2, restricted type B
pusch-ConfigCommon	
initialULBWPk2_0	2
initialULBWPmappingType_0	1
initialULBWPstartSymbolAndLength_0	0..127
initialULBWPk2_1	2
initialULBWPmappingType_1	1
initialULBWPstartSymbolAndLength_1	0..127
initialULBWPk2_2	7
initialULBWPmappingType_2	1
initialULBWPstartSymbolAndLength_2	0..127
msg3_DeltaPreamble	Range as per 3GPP TS
p0_NominalWithGrant	-202..24
pucch-ConfigCommon setup :	

pucchGroupHopping	0, neither 1, group hopping 2, sequence hopping
hoppingId	0..1024
p0_nominal	-202..24
ssb_PositionsInBurst_PR	<15 OR <255
ssb_PositionsInBurst_Bitmap	1
ssb_periodicityServingCell	5,10,20,40,80,160
dmrs_TypeA_Position	0, position 2 1, position 3
subcarrierSpacing	30 kHz
tdd-UL-DL-ConfigurationCommon	
referenceSubcarrierSpacing	30 kHz
dl_UL_TransmissionPeriodicity	ms0p5,ms0p625,ms1,ms1p25,ms2,ms2p5,ms5,ms10
nrofDownlinkSlots	0..20
nrofDownlinkSymbols	0..13
nrofUplinkSlots	0..20
nrofUplinkSymbols	0..13
ssPBCH_BlockPower	-60…50

7.3 EU Testbed RF Interfaces

This section reports the requirements for the interfaces I0601/I0606 and I0608/I0806, namely the RF interfaces between the SAT/TERR UE and gNB modems and the SAT/TERR channel emulator.

Figure 41 Front panel of the PROPSIM F64

The SAT channel emulator, i.e. PROPSIM F64 (see Figure 41), is equipped with RF input/output duplex ports and output only RF connectors that allow interconnection with third-party devices to be tested (e.g. RF transceiver), irrespective of the system technology or modulation (as the PROPSIM supports all major wireless standards and waveforms). The RF characteristics are reported in Table 32.

Table 32. PROPSIM RF characteristics

RF ports	N-female
TRX port configurations	24 (3 CUs)
Sync connections	C1-7: Controls for external units and accessories 10MHZ Ref IN/OUT 200MHz Sample clock IN/OUT
RF input	
RF input level	+25dBm RMS / +33dBm peak
Input signal level range at TRX port:	-27dBm… +23dBm rms. / +35dBm peak
Input signal level range at TRX port: BW up to 40/80/100/160 MHz BW up to 300 MHz BW up to 450 MHz BW up to 600 MHz BW up to 900 MHz BW up to 1200 MHz	All input level ranges specified give the maximum dynamic range for the user signal in RF domain. --27dBm… +23dBm rms. / +35dBm peak -21dBm… +23dBm rms. / +35dBm peak -16dBm… +23dBm rms. / +35dBm peak -16dBm… +23dBm rms. / +35dBm peak -11dBm… +23dBm rms. / +35dBm peak -11dBm… +23dBm rms. / +35dBm peak Input signal level refers to total wideband signal power.
Crest factor setting	0…24 dB (resolution 0.1 dB)
RF input/output resolution	0.1 dB
RF output	
RF output level	+5 dBm, peak
Output signal level range for full reference: BW up to 40/80/100/160 MHz BW up to 300 MHz BW up to 450 MHz BW up to 600 MHz BW up to 900 MHz BW up to 1200 MHz	Full reference refers to such emulator setting that gives the maximum dynamic range for the user signal in RF domain. -100dBm … -17dBm rms. / +5dBm peak -100dBm … -22dBm rms. / 0 dBm peak -100dBm … -26dBm rms. / -4 dBm peak -100dBm … -26dBm rms. / -4 dBm peak -100dBm … -31dBm rms. / -9 dBm peak -100dBm … -31dBm rms. / -9 dBm peak
RF output gain setting range	+5 … - 100 dB
RF Output level accuracy	< +/- 1.5dB < +/- 0.5dB typical Operation outside specified frequency range is not specified.
Level setting resolution	0.1 dB
Output noise floor output level ≤ –40 dBm	< –168 dBm/Hz
EVM	< –45 dB RMS, 5G NR 100/400 MHz, 256 QAM, 3.5 GHz < –45 dB RMS, 802.11ax 160MHz, 1024 QAM, 5.9 GHz
Crosstalk between TRX ports	< –100 dB

VSWR all RF ports	450 MHz to 700 MHz < 1.5 > 700 MHz to 2 GHz < 1.3 > 2 GHz to 6 GHz < 1.5

The RF characteristics of the SAT gNB and UE modems are reported in Table 33.

Table 33. USRP X300 RF characteristics

RF ports	Female SMA Connector for both the TX/RX and RX2 connectors
Maximum output power	○ 22 dBm @ (1.2GHz ~ 3GHz) ○ 12 ~ 22 dBm @ (3GHz ~ 6GHz)
Maximum input power	○ -15 dBm