

5G and Beyond: Analytical Comparison of Real NG-RAN Implementations

Michael Amistà

Apeiroon

Vicenza, Italy

michael.amista@apeiroon.com

Claudio E. Palazzi

Department of Mathematics

University of Padua

Padua, Italy

claudio.palazzi@unipd.it

Lorenzo Perinello

Department of Mathematics

University of Padua

Padua, Italy

lorenzo.perinello@phd.unipd.it

Gianluca Verin

Apeiroon

Vicenza, Italy

gianluca@apeiroon.com

Abstract—As mobile networks adopt increasingly heterogeneous multi-vendor RAN solutions, the complexity of configuring gNBs increases due to diverse vendor-specific architectures and parameter sets. This work presents an analytical comparison of Next Generation Radio Access Network (NG-RAN) implementations. Our systematic comparative analysis of different NG-RAN implementations included examination of vendor documentation and preliminary evaluation of CPU usage through experimental tests on dedicated hardware. Key functional areas were identified and a structured evaluation methodology was applied to highlight similarities, differences, and configuration dependencies between implementations.

Index Terms—5G, NG-RAN, srsRAN, openAirInterface

I. INTRODUCTION

The ongoing advancement of mobile communications toward 5G and future generations has fundamentally reshaped the architecture of the Next Generation Radio Access Network (NG-RAN). By introducing functional disaggregation, virtualization, and software-driven frameworks, 3GPP 5G New Radio (NR) standards offer remarkable flexibility. Although these innovations foster scalability and rapid development, they simultaneously introduce significant architectural complexity and configuration diversity, particularly within academic and experimental testbeds. Consequently, open-source software platforms for NG-RAN elements have emerged as crucial assets for empirical research, hands-on training, and rapid prototyping. Within this landscape, srsRAN and OpenAirInterface (OAI) represent two of the most prominent and widely used open-source 5G Software Defined Radio (SDR) stacks.

Despite their widespread adoption, these two solutions exhibit substantial disparities in internal design, configuration frameworks, and support for functional splits. Nevertheless, a comprehensive study combining analytical breakdown and empirical evaluation of both stacks specifically contrasting monolithic and disaggregated topologies (DU-CU split) is still largely missing from existing research. Bridging this gap is essential for researchers and system engineers aiming to evaluate the real-world capabilities and behavior of open-source gNodeB implementations.

This study delivers a detailed analytical and experimental comparison of srsRAN and OAI 5G gNodeBs across both monolithic and DU-CU split architectures. The analytical phase examines protocol coverage and underlying software

architectures, whereas the experimental phase delivers quantitative performance metrics focused on CPU consumption across the investigated deployment configurations.

II. RELATED WORK

A growing body of research analyzes and benchmarks open-source NG-RAN implementations, with many studies providing practical guidance for researchers and practitioners building 5G testbeds. Over the past few years, this literature has expanded rapidly as projects such as srsRAN and OpenAirInterface matured enough to support real NR experiments, but the set of directly comparable, reproducible studies remains relatively small. This is mainly because these open-source NR stacks are relatively new, their fully NR-capable releases have only appeared in the last few years, so the ecosystem is still maturing and researchers are only now beginning to build stable and complete testbeds. As a result, many papers focus on deployment guidance, single-stack characterization, or monolithic benchmarks, while only a few attempt direct head-to-head evaluations.

Alves *et al.* [1] present a careful head-to-head experimental comparison of srsRAN and OpenAirInterface in a controlled laboratory setup, detailing architectural differences and measured throughput/latency under several radio conditions. Several works document the experience of building full 5G standalone testbeds: Mamushiane *et al.* [2] describe a srsRAN + Open5GS deployment and contrast the feature sets of srsRAN and OAI, while Bozis *et al.* [3] demonstrate a versatile OAI-based 5G testbed on commodity hardware and discuss the tuning and stability challenges encountered in practice.

Other studies focus on the characterization of single stacks or the methodological aspects of experimentation. Rouili *et al.* [4] compare SDR-based, emulated, and simulated RAN deployments (using OAI in several setups) and quantify how emulation/simulation can misrepresent real over-the-air performance. Practical deployment guides by Seidel *et al.* [5] and Chepkoech *et al.* [6] assist in building low-cost testbeds, with Seidel *et al.* explicitly noting that OAI's extensive physical-layer parameters require significant engineering expertise to tune.

Taken together, these contributions establish that srsRAN and OAI are mature platforms for 5G experimentation and

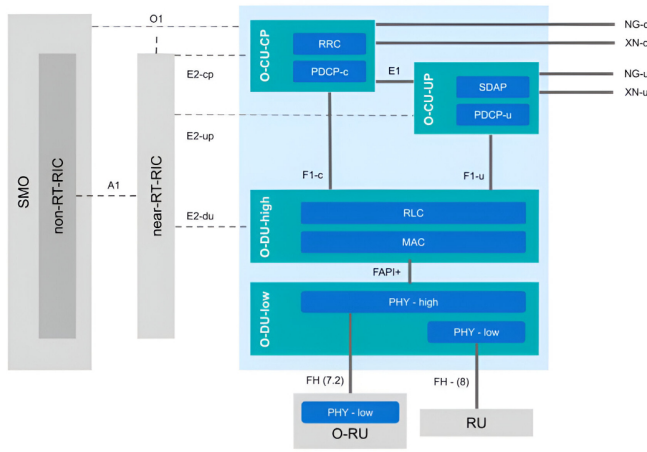


Fig. 1. gNB architecture following O-RAN standard

expose differences in feature support, performance profile, and ease of deployment. However, a key gap remains. Most comparative and experimental studies consider single-stack deployments or compare stacks only in monolithic gNB setups (i.e., where the DU-CU functions are co-located), or else they focus on one stack in multiple deployment modes. None of the works above presents a systematic implementation-level comparison that evaluates both srsRAN and OAI in both monolithic and DU-CU split topologies under the same hardware/testbed conditions. In other words, there is no published, reproducible study that isolates how architectural split (monolithic vs disaggregated) interacts with implementation choices in each stack in terms of CPU usage.

III. BACKGROUND

The 5G system can be viewed as a composition of four main elements: the User Equipment (UE), the Next Generation Radio Access Network (RAN for simplicity), the 5G Core (5GC), and the external data network [7]. The UE provides the wireless interface, the 5GC handles control and user-plane procedures such as registration, mobility, and session management, and the data network offers connectivity to external services. In the center of the RAN sits the gNodeB (gNB), which is responsible for enabling UEs to access the 5G system over the radio interface and for relaying traffic between the wireless link and the core network [8]. Additionally, gNB determines how radio and protocol functions are executed and how UEs can wirelessly reach 5G services.

A gNB can be implemented as a monolithic node containing the entire radio protocol stack, or as disaggregated functional blocks. The commonly adopted decomposition separates the Radio Unit (RU), the Distributed Unit (DU), and the Central Unit (CU). The RU hosts the radio front end and the lowest PHY components. The DU, in turn, is typically divided into DU-high and DU-low: DU-high carries out functions such as MAC and RLC, and DU-low implements time-critical portions of the PHY. The CU is composed of CU-CP (control plane) and CU-UP (user plane), which handle higher-layer

functions such as RRC, PDCP-c, PDCP-u, and SDAP. O-RAN decomposition, its functional blocks and its connecting logical interfaces are illustrated in Fig. 1.

The O-RAN representation also positions the non-RT RIC (non-Real-Time RAN Intelligent Controller) and the near-RT RIC alongside the gNB functions. These RICs operate over standard O-RAN interfaces (A1 and E2) to provide policy, optimization, and near-real-time control services to the RAN. At the same time, the RU-DU-CU chain manages actual radio access. Furthermore, multiple functional split options exist within the RU/DU/CU architecture, each reflecting different trade-offs in fronthaul bandwidth, timing constraints, and implementation complexity. In practical research setups, the most commonly adopted configurations are the monolithic gNB or the DU-CU split. This work focuses precisely on these two deployment models: in the DU-CU split, the DU hosts RLC/MAC and upper-PHY components, while the CU retains PDCP and RRC, thereby relocating latency-sensitive operations closer to the radio and leaving higher-layer tasks to the CU. This arrangement requires consistent parameterization and a sufficiently stable fronthaul, but provides a realistic structure for evaluating disaggregated RAN behavior.

IV. ANALYTICAL COMPARISON

The analytical comparison examines the internal organization, configuration schemas and protocol-procedure support of the two open-source gNodeB implementations considered in this study: srsRAN (release 24.10.1) and OpenAirInterface (OAI, release 2.3.0).

srsRAN's codebase is organized around modular components structured to facilitate future granularity in functional splits [9]. The implementation exposes a hierarchical, YAML-based configuration schema with top-level keys grouped by subsystem (e.g., *cu_cp*, *ru_sdr*, *cell_cfg*). srsRAN supports a DU-CU functional split, but, at the analyzed release, finer-grained sub-process separation (CU-CP / CU-UP, DU-High / DU-Low) is not fully realized as distinct running processes—rather, modular components exist within the same runtime organization.

OAI's architecture emphasizes a standards-aligned and feature-rich code layout, with distinct directories reflecting PHY (*openair1*), MAC/RLC/PDCP (*openair2*), and higher layers/interfaces (*openair3*) [10], [11]. OAI provides explicit support for functional splits and is implemented as a set of integrated modules designed to closely mirror 3GPP procedures; this results in an extensive set of low-level PHY and protocol parameters exposed to the operator.

The native configuration schemas differ substantially and are a primary source of operational friction. In fact, srsRAN's YAML-based format groups parameters by logical subsystem and encodes radio settings using a compact, human-readable style (examples include *dl_arfcn*, *band*, *channel_bandwidth_MHz*, and *common_scs*). By contrast, OAI adopts a structured configuration syntax in which equivalent concepts are expressed with different parameter names and encodings (e.g., *dl_arfcn*, *band*, *channel_bandwidth_MHz*, and

common_scs parameters in srsRAN correspond to OAI fields such as *dl_absoluteFrequencyPointA*, *dl_carrierBandwidth*, *dl_subcarrierSpacing*, and *initialDLBWPlocationAndBandwidth*.

Protocol and procedure-level coverage was catalogued for both F1AP and NGAP following the 3GPP definitions standardized by the latest stable releases at the time when this work was performed (F1AP/NGAP release 18.5.0 [12], [13]). Each implementation provides a subset of the procedures needed for UE lifecycle management and inter-node control, but differences in 3GPP release targets affect apparent completeness: srsRAN in the analyzed release aligns with F1AP release 17.4.1 and NGAP release 17.4.0; OAI's analyzed release shows compliance traces with earlier F1AP NGAP releases (notably release 16.3.1 for F1AP and release 15.8.0 for NGAP). Despite not fully conforming to the latest 3GPP F1AP/NGAP standards, both protocol stacks are sufficiently mature for production deployments. The absent procedures and information elements are defined as optional fields by the standard and relate only to secondary use cases and do not impact core network functionality.

Overall, the analytical comparison shows that the two stacks differ mainly in how their architectures and configuration schemas expose radio and protocol parameters, which directly affects cross-stack consistency and reproducibility. OAI adopts a broad and fine-grained set of configuration fields across PHY, MAC, and higher layers, resulting in a larger number of parameters that must be considered during setup and mapping. In contrast, srsRAN employs a more compact and hierarchical YAML-based configuration that groups related parameters and offers a smaller, more consolidated set of tunable fields. These structural differences make parameter translation a necessary step, as equivalent concepts are often encoded differently in each stack. Regarding protocol procedures, both implementations provide sufficiently mature and stable support for the F1AP and NGAP functionalities required for the deployments evaluated in this work, even though neither targets full alignment with the most recent 3GPP releases. Together, these factors define the practical challenges involved in aligning configurations and ensuring fair and reproducible comparisons between srsRAN and OAI.

V. PRELIMINARY TESTS

The experiments aim to compare srsRAN and OAI gNB CPU utilization under different network load conditions (NLC) in an ad-hoc testbed that reproduces essential components of a 5G system and emulates the behavior of real 5G networks. The testbed consists of a UE device acting as the end-user terminal, responsible for generating traffic and initiating procedures. The endpoint interaction with the other parts of the system allows us to observe latency, connectivity, and protocol behavior under supervised conditions. Proxmox, a virtual environment server solution, is used for deploying the NG-RAN stack under evaluation. Proxmox communicates with the UE via a USRP B210, which serves solely as an RF front-end. The USRP manages the analog transmission and

reception of NR signals, while the servers handle PHY, MAC, RLC, PDCP, RRC, and higher-layer procedures. Proxmox host runs the gNB and the 5G Core (5GC). Open5GS is deployed inside a virtual machine (VM1), where AMF, SMF, UPF, NRF, and PCF operate as independent SBA services communicating via standard HTTP/2 interfaces. External data network access is provided via a commercial router, which also serves as the endpoint for user-plane flows exiting the UPF, enabling realistic end-to-end data-flow evaluation. Regarding the hardware used to run Proxmox and the test, we used a server equipped with an Intel Core i3-N305 (8 physical cores at 3.80 GHz) and 16 GB of RAM, which uses CPU isolation to host the Proxmox server. Only four CPU cores are used: one is dedicated to CU processing, while the other three run DU operation, which is computationally heavier. Please note that no virtualization is used for running the gNB stack. All experiments use the following parameters: FR1 band n78 at 3.5 GHz, 20 MHz channel bandwidth, 30 kHz subcarrier spacing, and a single-UE configuration. Moreover, the 3GPP TDD frame pattern is used to balance uplink and downlink capacity. Note that the constant for the latter parameters ensures that observable performance differences arise only from the RAN implementation.

The performance evaluation is based on an end-to-end 5G SA topology in which: gNB runs directly on the Proxmox host, VM1 executes the 5GC, and VM2 runs a traffic generator. The N1, N3, and N6 interfaces are emulated via Proxmox virtual bridges, allowing us to connect all the components of the end-to-end pipeline. To isolate user-plane throughput generation from the core network VM and ensure that any computational bottleneck observed during tests reflects only the behavior of the gNB stack, the iPerf3 server runs on VM2. It is attached only to the N6 bridge. Once the UE completes registration and PDU session establishment, it initiates TCP or UDP iPerf3 sessions to VM2, generating controlled downlink and uplink traffic flows. iPerf3 provides application-layer throughput and can operate in both TCP and UDP modes in parallel, a feature that allows us to stress gNB with incremental loads. Bidirectional throughput is assessed by executing separate downlink and uplink sessions, and parallel pings from the UE to VM2 provide round-trip time samples at each test. In conclusion, the traffic follows the entire path (i.e., UE → gNB → 5GC → iPerf3) and ensures that all RLC, PDCP, GTP-U, and virtualization layers are exercised. The experiments focus on a key metric that together characterizes the usage of RAN resources. In this case CPU utilization on the reserved real-time cores is continuously monitored using htop.

A. Monolithic gNodeBs

Table I reveals significant differences in CPU utilization patterns between the two 5G implementations. OAI demonstrates substantially lower baseline CPU consumption, requiring only a span of 0.52-0.58 cores when idle compared to srsRAN's 0.89-0.92, suggesting more efficient resource management in idle states. The most striking difference appears in the TCP downlink traffic, where OAI maintains relatively modest CPU

TABLE I
MONOLITHIC GNB CPU UTILIZATION EXPRESSED AS NUMBER OF
CONSUMED CORE(S)

Network load condition (NLC)	srsRAN	OAI
No UE attached	0.89-0.92	0.52-0.58
UE attached (no traffic)	0.89-0.95	0.58-0.61
UL TCP traffic	1.50-1.57	1.42-1.48
DL TCP traffic	1.44-1.58	0.86-0.93
UL UDP traffic	1.26-1.33	1.25-1.35
DL UDP traffic	2.15-2.31	1.70-1.86

TABLE II
DU-CU SPLIT GNB CPU UTILIZATION EXPRESSED AS NUMBER OF
CONSUMED CORE(S)

NLC	srsRAN (DU+CU)	OAI (DU+CU)
No UE attached	0.71-0.77 + 0.06-0.07	0.55-0.58 + 0.014-0.016
UE attached no tr.	0.75-0.83 + 0.07-0.08	0.63-0.67 + 0.007-0.014
UL TCP traffic	1.40-1.50 + 0.12-0.17	1.40-1.44 + 0.04-0.06
DL TCP traffic	1.65-1.67 + 0.14-0.16	0.66-0.73 + 0.006-0.009
UL UDP traffic	0.85-0.88 + 0.07-0.08	0.70-0.77 + 0.012-0.014
DL UDP traffic	1.40-1.44 + 0.50-0.53	1.60-1.70 + 0.70-0.80

core usage at 0.86-0.93 span while srsRAN reaches 1.44-1.58, indicating the superior optimization of OAI for the processing of TCP downlinks. However, both implementations show similar CPU demands for uplink TCP and UDP traffic, with OAI slightly outperforming srsRAN in downlink UDP scenarios despite both reaching high utilization levels above 1.70 CPU cores. Although OAI provides better overall efficiency, particularly for baseline operations and TCP downlink traffic, both implementations face significant computational challenges when handling UDP traffic, especially in the downlink direction.

B. Disaggregated gNodeBs: DU-CU Split

As shown in Table II, the DU-CU split deployment reveals distinct architectural advantages for OAI in most scenarios as it demonstrates significantly lower CPU consumption of DU in idle states (0.55-0.58 vs 0.71-0.77 CPU cores) and maintains this efficiency advantage in most traffic conditions. The most dramatic difference occurs in downlink TCP processing, where OAI's DU utilization remains moderate with a span of 0.66-0.78 compared to srsRAN's 1.65-1.67, suggesting superior TCP downlink optimization in the split architecture. However, both implementations face considerable challenges with downlink UDP traffic, where the CU component becomes heavily loaded (0.50-0.53 used cores for srsRAN, 0.70-0.80 for OAI), indicating that UDP downlink processing places substantial demands on the centralized unit. Interestingly, while OAI's DU efficiency generally surpasses srsRAN's, its CU shows higher utilization in UDP downlink scenarios, suggesting different load distribution strategies between the two implementations' split architectures.

VI. CONCLUSION

This work compared srsRAN and OpenAirInterface through analytical inspection and preliminary experiments. Both implementations were found to support the main procedures required for a functional gNodeB, with differences that largely reflect project-specific design choices.

Future extensions include evaluating multi-UE and multi-cell scenarios, testing on more powerful hardware, and exploring additional SDR platforms and FR2 bands. Thorough experiments, also including other O-RAN splits, would broaden the understanding of disaggregated deployments.

ACKNOWLEDGMENT

Partial financial support for this work is provided by the Department of Mathematics of the University of Padua, through its SID funding program. This research also falls within the "Sustainable Mobility Center - Centro Nazionale per la Mobilità Sostenibile – CNMS" Spoke 8 "MaaS & Innovative Services" funded by the European Union under the PNRR – MISSIONE 4 Istruzione e Ricerca, project code CN00000023, thematic area: Sustainable Mobility (Mobilità Sostenibile).

REFERENCES

- [1] R. P. Alves, J. G. A. da S. Alves, M. R. Camelo, W. O. de Feitosa, V. F. Monteiro, and F. R. P. Cavalcanti, "Experimental comparison of 5g sdr platforms: srsran x openairinterface," 2024.
- [2] L. Mamushiane, A. Lysko, H. Kobo, and J. Mwangama, "Deploying a stable 5g sa testbed using srsran and open5gs: Ue integration and troubleshooting towards network slicing," in *2023 International Conference on Artificial Intelligence, Big Data, Computing and Data Communication Systems (icABCD)*, pp. 1–10, 2023.
- [3] E.-Z. G. Bozis, N. C. Sagias, M. C. Batistatos, M.-A. Kourtis, G. K. Xilouris, and A. Kourtis, "A versatile 5g standalone testbed based on commodity hardware," in *2024 Panhellenic Conference on Electronics & Telecommunications (PACET)*, pp. 1–4, 2024.
- [4] M. Rouili, N. Saha, M. Golkarifard, M. Zangoeei, R. Boutaba, E. Onur, and A. Saleh, "Evaluating open-source 5g sa testbeds: Unveiling performance disparities in ran scenarios," in *NOMS 2024-2024 IEEE Network Operations and Management Symposium*, pp. 1–6, 2024.
- [5] M. Seidel, A. I. Grohmann, P. Sossalla, F. Kaltenberger, and F. H. Fitzek, "How to get away with openairinterface: A practical guide to 5g ran configuration," in *2023 3rd International Conference on Electrical, Computer, Communications and Mechatronics Engineering (ICECCME)*, pp. 1–6, 2023.
- [6] M. Chepkoech, N. Mombeshora, B. Malila, and J. Mwangama, "Evaluation of open-source mobile network software stacks: A guide to low-cost deployment of 5g testbeds," in *2023 18th Wireless On-Demand Network Systems and Services Conference (WONS)*, pp. 56–63, 2023.
- [7] 3GPP, "5GS System Architecture for the 5G System (5GS); Stage 2," Technical Specification TS 23.501 V19.3.0, 3rd Generation Partnership Project, Mar. 2025.
- [8] 3GPP, "NR and NG-RAN Overall Description; Stage 2," Technical Specification TS 38.300 V18.5.0, 3rd Generation Partnership Project, Mar. 2025.
- [9] srsRAN Project, "srsRAN Project Release 24.10.1," 2025. Accessed: Jun. 20, 2025.
- [10] OpenAirInterface Software Alliance, "OpenAirInterface5G RAN Release 2.3.0." GitLab repository, 2025. Accessed: Jul. 8, 2025.
- [11] OpenAirInterface Software Alliance, "OpenAirInterface Software Architecture." GitLab repository, 2025. Accessed: Jul. 8, 2025.
- [12] 3GPP, "NG-RAN; F1 Application Protocol (F1AP)," Technical Specification TS 38.473 V18.5.0, 3rd Generation Partnership Project, Mar. 2025.
- [13] 3GPP, "NG-RAN; NG Application Protocol (NGAP)," Technical Specification TS 38.413 V18.5.0, 3rd Generation Partnership Project, Mar. 2025.