Implementing OpenRAN: Democratizing 5G Networks

Sumit Chakravarty
Dept. of Electronic Engineering
Kennesaw State University
Marietta, Georgia, USA
schakra2@kennesaw.edu

Ying Xie*
Dept. of Information Technology
Kennesaw State University
Marietta, Georgia, USA
yxie2@kennesaw.edu
**Corresponding Author*

Shaoen Wu
Dept. of Information Technology
Kennesaw State University
Marietta, Georgia, USA
swu10@kennesaw.edu

2024 IEEE 12th International Conference on Information, Communication and Networks (ICICN) | 979-8-3503-5580-2/24/$31.00 ©2024 IEEE | DOI: 10.1109/ICICN62625.2024.10761700

Abstract—**This paper highlights the endeavor to democratize 5G networks using OpenRAN by making their deployment and ownership more accessible to a broader range of stakeholders. The purpose is to break down existing barriers and foster greater inclusivity in the 5G landscape. To achieve this goal, we proposed a method involving the decoupling of software and hardware, coupled with leveraging virtualization techniques to simulate User Equipment (UE) and gNodeB. This approach embraces open standards and interfaces, effectively lowering the barriers to entry for new vendors and promoting innovation in the 5G ecosystem. Through these efforts, the project underlines the transformative potential of making 5G networks more democratic, inclusive, and conducive to accelerated technological advancements.**

Keywords—5G, Access and Mobility Management Function (AMF), base station, Control Unit, Distributed Unit, FlexRIC, Radio Unit, gNodeB (gNB), MSG (message), OAI, OpenRAN, Ran Intelligent Controller (RIC), User Equipment (UE)

I. Introduction

Open RAN refers to a general movement towards a more diversified RAN architecture and the architecture proposed by the O-RAN Alliance (O-RAN) based on 3GPP standards [3, 2]. Using O-RAN's architecture, the OpenAirInterface Software Alliance (OAI) has developed open-source code for implementing this architecture [1]. This project aimed to simulate activity between a base station (gNB) and an end-user device (UE) using OAI open-source 5g software. Although the concept of Open RAN in radio communications is familiar [3], recent years have seen a surge in innovation. More and more individuals in the industry are motivated to embrace open-source technology to disseminate control over physical radio components. In the past, this control was limited to a handful of vendors. By transitioning away from proprietary hardware and opting for a more "open, flexible, and virtualized network" [1], a more diverse RAN architecture for mobile radio communications becomes a reality, as evidenced by OAI Open RAN.

A. Benefits of Open RAN

Open RAN technology offers various benefits, the most significant being an alternative to conventional equipment vendors like Ericsson and Nokia [3]. In the past, the interaction between the radio unit (RU) and distributed unit (DU) was a "black box" of proprietary interfaces. However, Open RAN is an open-source technology that ensures vendor diversity and promotes innovation without locking users into a single vendor [3,1]. Additionally, Open RAN virtualization empowers multiple operators to share a single RAN, eliminating the need for physical components for each operator. When combined with eliminating proprietary vendor lock-in, this significantly reduces the cost of deployment and maintenance for existing and new suppliers [3].

B. Purpose

This paper aims to create a framework that future students can utilize in their research. By demonstrating the implementation of OAI OpenRAN software, we aim to establish a foundation for integrating new components, supporting multiple UEs, and incorporating FlexRIC.

Table 1. shows the issues with current implementations of RAN.

Issues	**Details**
Vendor Lock-In	Physical components in RAN are created and controlled by a few vendors.
Closed Source	Proprietary software, hence less room for innovation.
High Cost	The cost of breaking into the industry is high due to proprietary software.
Black-Box	Unable to visualize and learn with "black box" technologies.

II. Architecture: 5G, OpenRan, OAI

This section will discuss overall 5G architecture, how O-RAN OpenRAN specifications have been built within this framework, and how the OAI software implements these standards.

Open Air Interface has developed the O-RAN alliance, which is an open technology that avoids the black box nature of vendors like Ericson and Nokia. It ensures vendor diversity and does not lock in a vendor. Additionally open virtualization empowers multiple operators to share a single RAN, eliminating the need to buy components for each operator. This significantly reduces the Capex and the Opex costs.

The project aims to clearly document and elucidate the installation and operation of the O-RAN, so that it can be further developed by future users to incorporate concepts like Network

This material is based upon the work partially supported by the National Science Foundation under Grant No. 2300955

Function Virtualizations(NFVs) and Software Defined Networks (SDNs).

A. 5G Architecture

There are three components to a basic 5G architecture: user equipment (UE), radio access network (RAN), and core network (5GC) [1]. The interaction of these components is illustrated below:

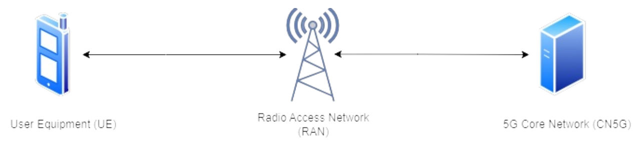


Figure 1
5G Architecture diagram, derived from 3GPP illustration [2]

Each of these components can be summarized as follows:

- **UE**: The device a user uses to connect to the 5G network. This is composed of the mobile station (MS) and SIM card (USIM).
- **gNB**: 5G radio transmitter or base station. This is composed of the gNB-CU (Central Unit) and gNB-DU (Distributed Unit)
- **5G**: 5G Core Network [1].

The RAN is equipped with protocols that facilitate communication between the gNB and UE for both user and control data functions. This is illustrated in the diagram, which depicts the "User plane" on the right, responsible for managing user data, and the "Control plane" on the left, responsible for overseeing control signals.

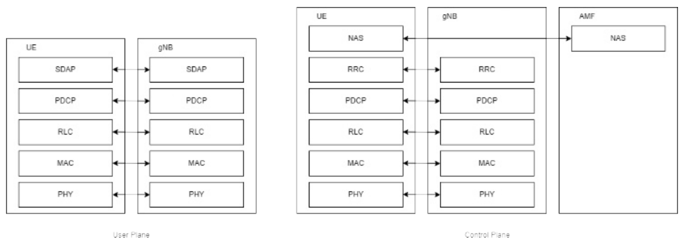


Figure 2
5G RAN protocol stacks [1]

The layers of the RAN protocol stack can be described as follows:

- 5G Layer 1: Physical (PHY) layer
- 5G Layer 2: Medium Access Control (MAC), Radio Link Control (RLC), Packet Data Convergence Protocol (PDCP) layers
- 5G Layer 3 (Control Plane only): Radio Resource Control (RRC) layer

Wireless networks rely on multiple layers, each with its own protocol, for communication to occur. Radio signals are converted into PDUs and transferred to higher logical layers [1].

To connect to a 5G network, a device first sends an Attach message to the Access and Mobility Management Function (AMF) through the gNB, which is a component of the 5G CoreNetwork [1]. However, for our project, we are only interested in the interactions between the gNB and UE, not the entire 5G Core Network, as we are focused on the attachment procedure between the UE and the node B and documenting the communication protocols for this procedure.

B. O-RAN Architecture

Using 3GPP-defined 5G architecture, O-RAN developed the architecture for Open RAN. Open RAN takes the specifications of 5G and seeks to offer a way to democratize the service via open-source hardware.
Open RAN can be broken down from top to bottom into the following logical layers:

- Radio Unit (RU) – part where radio frequency signals are transmitted, can be hardware or software
- Distributed Unit (DU) – part that digitizes radio signal and sends to the CU
- Central Unit (CU) – part that sends the digitized signal from the DU into the network
- RIC (RAN Intelligent Controller) – part that contains API, allowing new radio services to be added to system. Allows new software to communicate with other parts of RAN [3, 1].

Below (Figure III) is a diagram of Open RAN architecture. This architecture can be broken down (from top to bottom) into the following logical layers: ONAP (Orchestration and Automation), Near-Real Time RIC (n-RT RIC), CU, DU, and RU. It also contains various interfaces that connect these layers [3]. The gNB is composed of the n-RT RIC, CU, DU, and RU, as denoted in the diagram above [1].

- *ONAP*

The ONAP is outside the RAN and contains several important elements. Service Management and Orchestration (SMO) controls the non-real-time RIC (non-RT RIC) and manages services connection, traffic, and performance monitoring. The non-RT RIC is separate from the n-RT RIC, as it is used to control less time-sensitive control functions [3].

- n-RT RIC

Using information from the non-RT RIC, the n-RT RIC optimizes radio resources using data from the non-RT RIC. This function manages the more time-sensitive operations.

- *CU, DU, and RU*

The CU comprises of the User Plan and Control Plane, as discussed previously. The CP portion handles RRC (Radio Resource Control) and part of PDCP (Packet Data Convergence Protocol). The UP-portion handles SDAP (Service Data Adaption Protocol), as well as PDCP as well. Both the Central Unit (CU) and Distributed Unit (DU) are software-based and can be deployed on a virtualized platform. The DU is responsible for managing the Radio Link Control (RLC) and Medium Access Control (MAC) layers while being supervised by the CU, which controls the overall network functions. In contrast, the Radio Unit (RU) manages the Physical Layer (PHY) and Radio Frequency (RF) processing [3].

- *OAI Software*

The OpenAirInterface Software Alliance (OSA) has developed OAI, a software that enables virtualization of specific components in the RAN. OAI comprises two primary projects: OAI 5G RAN and OAI 5G Core Network [4], with our focus being on the RAN project [5].

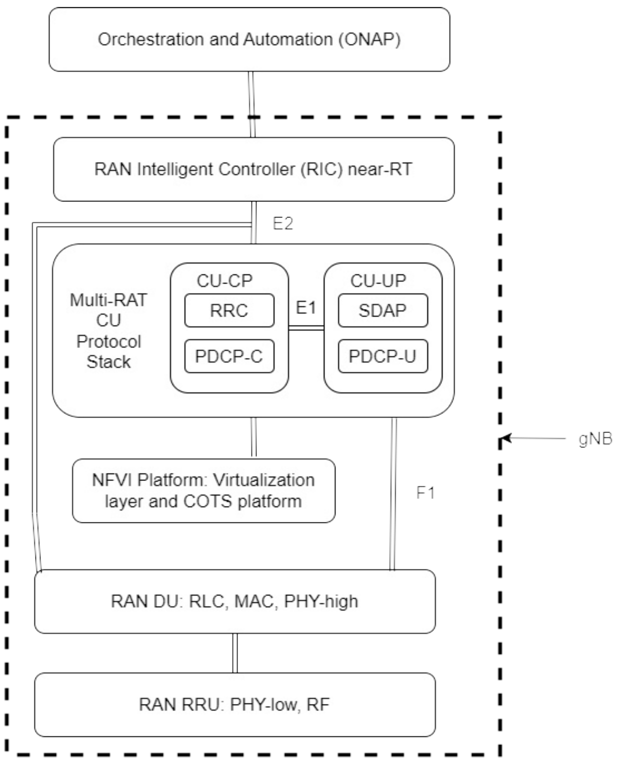

Orchestration and Automation (ONAP)
RAN Intelligent Controller (RIC) near-RT
E2
Multi-RAT CU Protocol Stack
CU-CP
RRC
PDCP-C
E1
CU-UP
SDAP
PDCP-U
gNB
NFVI Platform: Virtualization layer and COTS platform
F1
RAN DU: RLC, MAC, PHY-high
RAN RRU: PHY-low, RF

Figure 3
O-RAN ARCHITECTURE DIAGRAM [4]

III. Deployment

The methodology followed in the paper includes installing and configuring the necessary components, testing the components as demonstrated by the interaction between each layer of the model using Wireshark outputs, and finally, documenting for future researchers.

To install our components, we followed instructions from the OAI 5G RAN Github repository [5]. Here are the final installation steps:

The project sed the following server components

Minimum VM/Device specs:

- *Ubuntu 22.04.3 LTS*
- *256 GB storage*
- *16 GB RAM*
- *2 core CPU*

1) Install dependencies

```
sudo apt update
sudo apt-get update
sudo apt-get install ca-certificates gpg wget
sudo apt-get install kitware-archive-keyring
sudo apt-get install cmake
sudo apt  install cmake-curses-gui # restart
sudo apt install asn1c libsctp-dev poppler-utils
python3.10 libreoffice
sudo apt-get install swig
sudo apt install gcc-10
sudo update-alternatives --install /usr/bin/gcc gcc
/usr/bin/gcc-10 10
```

2) Retrieve OpenAirInterface5G source code

```
git clone
https://gitlab.eurecom.fr/oai/openairinterface5g.git
~/openairinterface5g
cd ~/openairinterface5g
git checkout develop
```

3) Install NRScope dependencies

```
sudo apt install -y libforms-dev libforms-bin
```

4) Build OAI gNB and UE

```
cd ~/openairinterface5g
source oaienv
cd cmake_targets
./build_oai -I # Install OAI dependencies
./build_oai -w USRP --ninja --nrUE --gNB --build-lib
```

5) Launch gNB and UE

Navigate to path:

1. cd openairinterface5g/cmake_targets/ran_build/build
2. #start UE
   sudo ./nr-uesoftmodem -r 106 --numerology 1 --band 78 -C 3619200000 --rfsim --sa --uicc0.imsi 001010000000001 --rfsimulator.serveraddr 127.0.0.1
3. #Start gNB
   sudo ./nr-softmodem -O ../../../targets/PROJECTS/GENERIC-NR-5GC/CONF/gnb.sa.band78.fr1.106PRB.usrpb210.conf --gNBs.[0].min_rxtxtime 6 --rfsim –sa

A. *Initial* Access and Attachment

Our project did not include a simulated core network, as we focused on the UE-gNB initial connection. The UE, gNB, and AMF would proceed with the process outlined below in a real-world scenario.

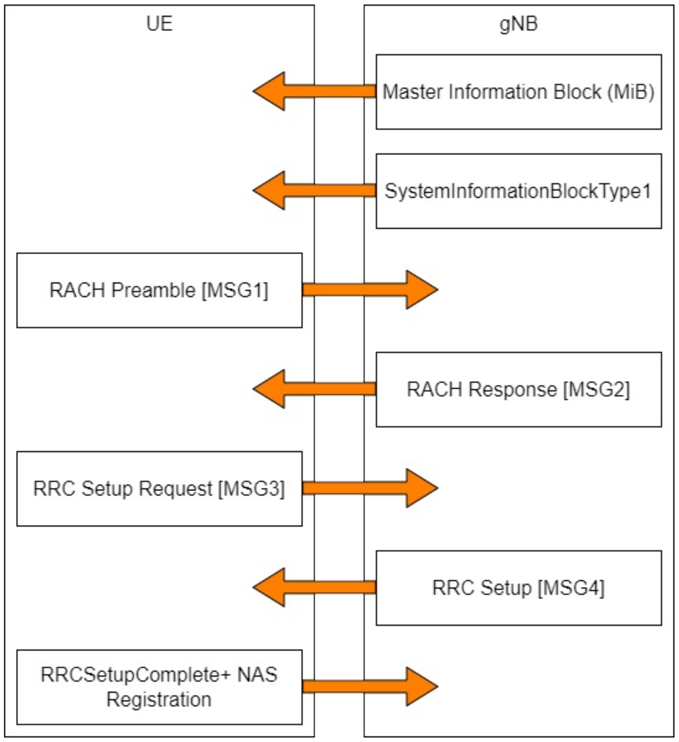

UE
gNB
Master Information Block (MiB)
SystemInformationBlockType1
RACH Preamble [MSG1]
RACH Response [MSG2]
RRC Setup Request [MSG3]
RRC Setup [MSG4]
RRCSetupComplete+ NAS Registration

Figure 4
Initial Attachment Procedure [1].

The demonstration of the UE to gNB connectively is performed by the following procedure.

1. Start up Wireshark capture
2. Move to build folder for oai for both gNB and UE screens,
3. Start UE,
4. Start gNB
5. Record and present Wireshark data.

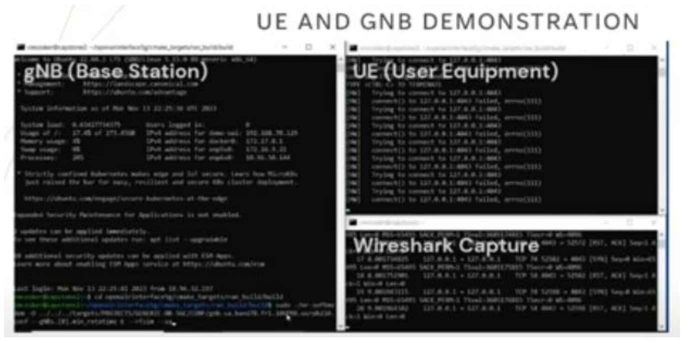


Figure 5 UE and GNB

The screenshot (Figure V) shows that the UE starts looking for connection, once the gNB is started the connection is made and the pack capture is initiated. Once the base station gNB is stopped, the packet capture is stopped. The breakdown of the attachment procedure is shown in figure IV and is detailed below.

The initial communications for a UE to connect to a gNB are as follows:

- DL Sync - the gNB is broadcasting the MiB and SiB Type 1, which the UE receives and decodes
- UL Sync - the UE starts the Random Access (RACH) procedure to start a connection with RRC. This is the RACH MSG1. gNB responds with MSG2.
- SRB0 – the UE sends the RRC set up request (MSG3) and the gNB responds (MSG4)
- SRB1 – The UE communicates that it has completed the RRC setup and requests registration with the core AMF component [1].

Our deployment observations are presented in the table below, with Output columns sourced directly from UE and gNB.

TABLE 2: Observations gathered via the deployment
Initial attachment procedure with UE and gNB output

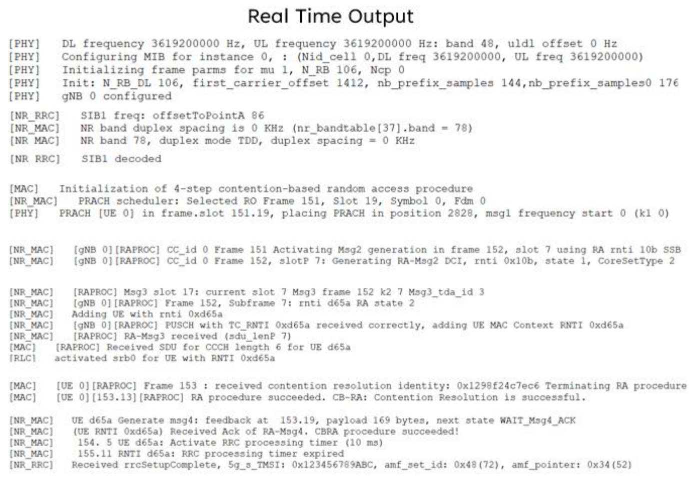
Protocol			Real Time Output			
Procedure	UE	gNB	Line	UE Output	Line	gNB Output
		<= Master Information Block (MiB)	256 ?	[NR_RRC] Configuring MAC for first MIB reception	192	[PHY] Configuring MIB for instance 0, : (Nid_cell 0,DL freq 3619200000, UL freq 3619200000)
DL Sync		<= SystemInformationBlockType1	264	[NR_RRC] SIB1 decoded	201	[NR_RRC] SIB1 freq: offsetToPointA 86
	RACH Preamble [MSG1] =>		278	[PHY] PRACH [UE 0] in frame.slot 151.19, placing PRACH in position 2828, msg1 frequency start 0 (k1 0), preamble_offset 13, first_nonzero_root_idx 0	416	[NR_PHY] [gNB 0][RAPROC] Frame 151, slot 19 Initiating RA procedure with preamble 52, energy 54.0 dB (I0 0, thres 120), delay 0 start symbol 0 freq index 0
					418	[NR_MAC] [gNB 0][RAPROC] CC_id 0 Frame 151 Activating Msg2 generation in frame 152, slot 7 using RA rnti 10b SSB, new rnti d65a index 0 RA index 0
UL Sync		<= RACH Response [MSG2]	283	[PHY] RAR-Msg2 decoded	419	[NR_MAC] [gNB 0][RAPROC] CC_id 0 Frame 152, slotP 7: Generating RA-Msg2 DCI, rnti 0x10b, state 1, CoreSetType 2
	RRC Setup Request [MSG3] =>		284	[NR_MAC] [RAPROC][152.17] RA-Msg3 transmitted	420	[NR_MAC] [RAPROC] Msg3 slot 17: current slot 7 Msg3 frame 152 k2 7 Msg3_tda_id 3
					424	[NR_MAC] [RAPROC] RA-Msg3 received (sdu_lenP 7)
					428	[MAC] [UE 0][RAPROC] Frame 153 : received contention resolution identity: 0x1298f24c7ec6 Terminating RA procedure
					437	[NR_MAC] UE d65a Generate msg4: feedback at 153.19, payload 169 bytes, next state WAIT_Msg4_ACK
SRB0		<= RRC Setup [MSG4]	285	[MAC] [UE 0][RAPROC] Frame 153 : received contention resolution identity: 0x1298f24c7ec6 Terminating RA procedure	438	[NR_MAC] (UE RNTI 0xd65a) Received Ack of RA-Msg4. CBRA procedure succeeded!
					441	[NR_RRC] Received rrcSetupComplete, 5g_s_TMSI: 0x123456789ABC, amf_set_id: 0x48(72), amf_pointer: 0x34(52), 5g TMSI: 0x56789ABC
					442	[NR_RRC] [FRAME 00000][gNB][MOD 00][RNTI 1] [RAPROC] Logical Channel UL-DCCH, processing NR_RRCSetupComplete from UE (SRB1 Active)
SRB1	RRCSetupComplete+ NAS Registration =>		286	[MAC] [UE 0][153.13][RAPROC] RA procedure succeeded. CB-RA: Contention Resolution is successful.	443	[NR_RRC] [FRAME 00000][gNB][MOD 00][RNTI 1] UE State = NR_RRC_CONNECTED

The snippets from the deployment's real time output is shown in figure VI, while the summary of the process is below. The initial communications for a UE to connect to a gNB are as follows:

First is the down link sync - the gNB is broadcasting the MiB and SiB Type 1, which the UE receives and decodes
Next is the up link sync – here, the UE starts the random access procedure to start a connection with the RRC. This is the random-access message 1. The gNB then responds with message 2

Next is the SRB 0 – the UE sends the RRC set up request in message 3 and the gNB responds, which is message 4
Finally, the last step is SRB 1 – The UE communicates that it has completed the RRC setup and requests registration with the core AMF component.

In a real-world scenario, the UE would then be able to connect to the AMF in the core 5G network.



Real Time Output

```
[PHY]    DL frequency 3619200000 Hz, UL frequency 3619200000 Hz: band 48, uldl offset 0 Hz
[PHY]    Configuring MIB for instance 0, : (Nid_cell 0,DL freq 3619200000, UL freq 3619200000)
[PHY]    Initializing frame parms for mu 1, N_RB 106, Ncp 0
[PHY]    Init: N_RB_DL 106, first_carrier_offset 1412, nb_prefix_samples 144,nb_prefix_samples0 176
[PHY]    gNB 0 configured

[NR_RRC]   SIB1 freq: offsetToPointA 86
[NR_MAC]   NR band duplex spacing is 0 KHz (nr_bandtable[37].band = 78)
[NR MAC]   NR band 78, duplex mode TDD, duplex spacing = 0 KHz

[NR RRC]   SIB1 decoded


[MAC]   Initialization of 4-step contention-based random access procedure
[NR_MAC]   PRACH scheduler: Selected RO Frame 151, Slot 19, Symbol 0, Fdm 0
[PHY]   PRACH [UE 0] in frame.slot 151.19, placing PRACH in position 2828, msg1 frequency start 0 (k1 0)


[NR_MAC]   [gNB 0][RAPROC] CC_id 0 Frame 151 Activating Msg2 generation in frame 152, slot 7 using RA rnti 10b SSB
[NR_MAC]   [gNB 0][RAPROC] CC_id 0 Frame 152, slotP 7: Generating RA-Msg2 DCI, rnti 0x10b, state 1, CoreSetType 2


[NR_MAC]   [RAPROC] Msg3 slot 17: current slot 7 Msg3 frame 152 k2 7 Msg3_tda_id 3
[NR_MAC]   [gNB 0][RAPROC] Frame 152, Subframe 7: rnti d65a RA state 2
[NR_MAC]   Adding UE with rnti 0xd65a
[NR_MAC]   [gNB 0][RAPROC] PUSCH with TC_RNTI 0xd65a received correctly, adding UE MAC Context RNTI 0xd65a
[NR_MAC]   [RAPROC] RA-Msg3 received (sdu_lenP 7)
[MAC]   [RAPROC] Received SDU for CCCH length 6 for UE d65a
[RLC]   activated srb0 for UE with RNTI 0xd65a

[MAC]   [UE 0][RAPROC] Frame 153 : received contention resolution identity: 0x1298f24c7ec6 Terminating RA procedure
[MAC]   [UE 0][153.13][RAPROC] RA procedure succeeded. CB-RA: Contention Resolution is successful.

[NR_MAC]   UE d65a Generate msg4: feedback at  153.19, payload 169 bytes, next state WAIT_Msg4_ACK
[NR_MAC]   (UE RNTI 0xd65a) Received Ack of RA-Msg4. CBRA procedure succeeded!
[NR_MAC]    154. 5 UE d65a: Activate RRC processing timer (10 ms)
[NR_MAC]    155.11 RNTI d65a: RRC processing timer expired
[NR_RRC]   Received rrcSetupComplete, 5g_s_TMSI: 0x123456789ABC, amf_set_id: 0x48(72), amf_pointer: 0x34(52)
```

Figure 6
Code snippets from real-time deployment of UE-gNB connectivity

IV. Conclusion

Our project achieved successful results by implementing OpenRAN components on a single virtual server. We aimed to provide a 5G Standalone CN implementation compliant with 3GPP through OSA. By relying solely on software, we were able to effectively enable communication between the gNodeB, and UE in line with OSA's vision. This success is a testament to the enormous potential of OpenRAN technology and its capacity to transform wireless communication.

Based on our initial findings, we can confirm that OpenRAN can be easily installed on a single Ubuntu server using the recommended resources. With clear and concise instructions, even non-experts can configure the essential components of an OpenRAN system. This work has the potential to establish a solid foundation for further development of these components, including adding features like FlexRIC for testing and research. Ultimately, democratizing 5G networks will promote healthy competition and encourage innovation, breaking the grip of the current two-party vendor network.

References

[1] S. Mallasen Quintana, “Deployment and analysis of a 5G NR radio access network based on Open RAN, using USRPs and OpenAirInterface,” End of Degree Project, Universitat Politècnica de València, 2022. Accessed: Nov. 09, 2023. [Online]. Available: https://riunet.upv.es/handle/10251/187834

[2] A. Sultan, “5G System Overview,” *www.3gpp.org*, Aug. 08, 2022. https://www.3gpp.org/technologies/5g-system-overview (accessed Nov. 09, 2023).

[3] X. Krasniqi, E. Hajrizi, and B. Qehaja, “Challenges and Lessons Learned During Private 5G Open RAN Deployments,” in *IEEE Xplore*, IEEE, Sep. 2023.

[4] OpenAirInterface Software Alliance, “5G RAN – OpenAirInterface,” *OpenAirInterface.org*, 2023. https://openairinterface.org/oai-5g-ran-project/ (accessed Nov. 29, 2023).

[5] OpenAirInterface Software Alliance, “oai / openairinterface5G · GitLab,” *GitLab*. https://gitlab.eurecom.fr/oai/openairinterface5g/ (accessed Nov. 29, 2023).

[6] O-RAN Alliance (2018). O-RAN alliance. Available at: https://www.o-ran.org/about (Accessed October 10, 2022).

[7] O-RAN Alliance (2019). O-RAN alliance. Available at: https://docs.o-ran-sc.org/projects/o-ran-sc-ric-app-kpimon/en/latest/(Accessed December 01, 2022).

[8] O-RAN Alliance (2020). O-RAN alliance. Available at: https://static1.squarespace.com/static/5ad774cce74940d7115044b0/t/5e95a0a306c6ab2d1cbca4d3/1586864301196

[9] [O-RAN Working Group 1 (2021). O-RAN architecture description v6.0. O-RAN Alliance. Available at: https://oranalliance.atlassian.net/wiki/spaces/OAH/pages/2331377807/O-RAN+Architecture+Description+v6.0

[10] O-RAN Working Group 4 (2022). O-RAN fronthaul control, user, and synchronization plane specification –v10.0. Technical Report, O-RAN Alliance, 2022. Available at: https://oranalliance.atlassian.net/wiki/spaces/OAH/pages/2587525385/O-RAN+Control+User+and+Synchronization+Plane+Specification+v10.00 (Accessed on November 2022).

[11] O-RAN Working Group 6 (2020). Cloud architecture and deployment scenarios for RAN virtualized RAN v02.02. O-RAN Alliance. Available at: https://oranalliance.atlassian.net/wiki/spaces/OAH/pages/854262284/WG6+O-RAN+Cloud+Architecture+and+Deployment+Scenarios+for+O-RAN+Virtualized+RAN+v2.1+ORAN.WG6.CAD-v02.01+-+July+2020 (Accessed on October 2022).

[12] Parallel Wireless (2020). Everything you need to know about Open RAN. Available at: https://www.parallelwireless.com/wp-content/uploads/Parallel-Wireless-e-BookEverything-You-Need-to-Know-about-Open-RAN.pdf (Accessed June 15, 2022).

[13] Polese, M., Bonati, L., D’Oro, S., Basagni, S., and Melodia, T. (2022). Understanding - RAN: Architecture, interfaces, algorithms, security, and research challenges. arXiv preprint arXiv:2202.01032.

[14] Qiu, J., Tian, Z., Du, C., Zuo, Q., Su, S., and Fang, B. (2020). A survey on access control in the age of internet of things. IEEE Internet Things J. 7, 4682–4696. doi:10.1109/jiot.2020.2969326

[15] Rimedo Labs (2021). O-RAN use cases: Traffic steering. Available at: https://rimedolabs.com/blog/o-ran-use-cases-traffic-steering/(Accessed December 01, 2022).