Experimental comparison of 5G SDR platforms: srsRAN x OpenAirInterface

Ruan P. Alves, João Guilherme A. da S. Alves, Mikael R. Camelo,
Wilker O. de Feitosa, Victor F. Monteiro, Fco. Rodrigo P. Cavalcanti

***Abstract*— A Software-Defined Radio (SDR) platform is a communication system that implements as software functions that are typically implemented in dedicated hardware. One of its main advantages is the flexibility to test and deploy radio communication networks in a fast and cheap way. In the context of the Fifth Generation (5G) of wireless cellular networks, there are open source SDR platforms available online. Two of the most popular SDR platforms are srsRAN and OpenAirInterface. This paper presents these two platforms, the characteristics of the networks created by them, the possibilities of changes in their interfaces and configurations, and also their limits. Moreover, in this paper, we also evaluate and compare both platforms in an experimental setup deployed in a laboratory.**

***Keywords*— Software-defined radio, srsRAN, OpenAirInterface, 5G.**

I. Introduction

Aiming at reducing costs and the time required for tests, deployment and updates, telecommunication operators and equipment vendors have been considering the replacement of network components typically implemented in dedicated hardware by software running in programmable computers [1]. One of the main enablers of this "virtualization approach" is the use of the concept of Software-Defined Radio (SDR) [2]. The ultimate goal of such approach is to decouple hardware and software which is expected to bring reduced costs and increased flexibility in network operation.

A SDR is a radio communication system where traditional hardware components, such as mixers, filters, amplifiers, and modulators/demodulators, are replaced or augmented by software processing. In a SDR, the majority of the radio's functionality is implemented using software running on a general-purpose computer or embedded system.

In simpler terms a SDR-based network typically consists of: telecommunication radios, a software platform and computers. Regarding the telecommunication radios, they are responsible for transmission and reception of signals. Concerning the software platform, it implements as software the functions previously implemented as hardware, e.g., signal processing. Finally, the computers are connected to the telecommunication radios and are responsible for executing the software platform.

Regarding the telecommunication radios, they can be found in different configurations and price ranges. They usually differ according to their capabilities, e.g., supported bandwidth, supported central frequency for transmission and reception, etc. Some popular devices used for this purpose are bladeRF, LimeSDR and Universal Software Radio Peripheral (USRP).

Concerning SDR platforms that implement Fifth Generation (5G) networks, two prominent examples are Software Radio Systems Radio Access Networks (srsRAN) and OpenAirInterface (OAI). The srsRAN is an open source initiative, while the OAI is more closed. The OAI is older than the srsRAN and has a bigger and centralized team.

In this context, the present paper extends our previous work using SDRs. On the one hand, in [3], we have deployed a Fourth Generation (4G) Long Term Evolution (LTE) network testbed using SDRs to evaluate an algorithm that predicts the signal quality of a link between an User Equipment (UE) and its serving eNodeB (eNB), i.e., a 4G base station. On the other hand, the present paper presents the results related to the implementation of two distinct 5G networks, each one using a different SDR platform, i.e., srsRAN and OAI.

This paper is organized as follows. Section II presents a technical background related to both SDRs platform. The technical details related to our implementations are presented in Section III. The obtained results and related discussions are presented in Section IV. Finally, Section V presents the conclusions of the present study and its future perspectives.

II. Background on 5G SDR-based Networks

The srsRAN is an end-to-end solution for LTE and New Radio (NR) networks. Its NR implementation has three main components: **Software Radio Systems User Equipment (srsUE)**, **Software Radio Systems gNodeB (srsENB)** and **Software Radio Systems Evolved Packet Core (srsEPC)**, which implement, respectively, the three main elements of a NR network (Fig. 1):

- **User Equipment:** The UE is any device used directly by an end-user to communicate. For example, a notebook and a smartphone.
- **gNodeB:** it is the 5G Base Station (BS). It provides the connectivity between UE and the Core Network (CN).
- **Core Network:** it is the heart of a 5G-NR network, and ensures the efficient and reliable delivery of traffic between network nodes.

More specifically, the main characteristics of each srsRAN component are:

- **srsUE:** Support to: LTE and NR (Non-Standalone (NSA)/Standalone (SA)); Frequency Division Duplex

The authors are with the Wireless Telecommunications Research Group (GTEL), Federal University of Ceará (UFC), Fortaleza, Ceará, Brazil. The work of Victor F. Monteiro was supported by CNPq under Grant 308267/2022-2.

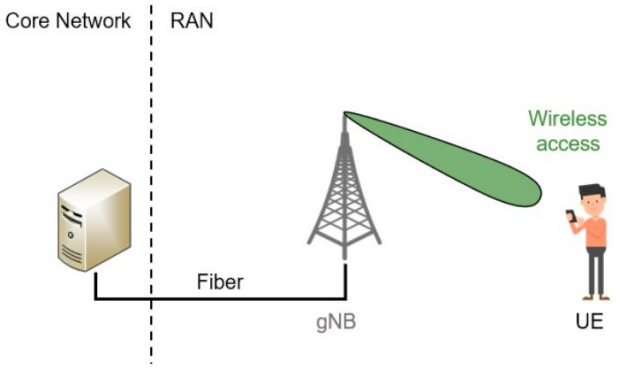


Fig. 1. Simplified architecture of a 5G/NR network.

Fig. 2. Environment of the deployed SDR network. UE's on the left, CN + gNB on the right.

(FDD) and Time Division Duplex (TDD) schemes; bandwidths: 1.4, 3, 5, 10, 15 and 20 MHz; up to 2x2 Multiple Input Multiple Output (MIMO); up to 256 Quadrature Amplitude Modulation (QAM) modulation (only downlink); and physical Subscriber Identity Module (SIM) cards using a PC/SC reader.

- **srsgnb:** Support to: TDD and FDD schemes; subcarrier spacings: 15 and 30 kHz (FR1); bandwidths: 5, 10, 15, 20, 25, 30, 40, 50, 60, 70, 80, 90, 100 MHz (273 Physical Resource Blocks); MIMO up to 2x2; 64 QAM modulation in downlink.
- **srsEPC:** It is compatible only with LTE, and, moreover, supports only a small set of LTE features. For example, it does not support some features such as Inter-eNB (S1) roaming and VoLTE. To run these applications, it is necessary to use another CN software.

We highlight that with these components, srsRAN is able to run an end-to-end LTE and 5G NSA network. For 5G SA, it is necessary a separate CN, since **srsEPC** only acts as an Evolved Packet Core (EPC), i.e., the 4G CN. The srsRAN Project documentation [4] suggests that if someone wants to deploy a complete 5G SA network, he/she can replace its CN by the CN provided by the Open5GS, where the Open5GS is a project that provides only the CN.

The second platform, OAI, is a mobile network solution, which, as the srsRAN, is designed to build end-to-end LTE and NR networks. Despite the similarities, there are differences between these two platforms. For example, the nomenclature of the components. In OAI we have the UE split into **nr-uesoftmodem** for NR and **lte-uesoftmodem** for LTE; moreover we have **nr-softmodem** as gNodeB (gNB) for NR and **lte-softmodem** as eNB for LTE; concerning the CN, it has **OpenAirInterface 5G Core Network (OAI-5GCN)** and **OpenAirInterface-Core Network (OAI-CN)**, which are the 5G and 4G CN, respectively. Regarding the OAI features, some of them are:

- **UE for LTE (lte-uesoftmodem):** Support to: FDD and TDD schemes; bandwidths: 5, 10, and 20 MHz; and MIMO up to 2x2.

- **eNB (lte-softmodem):** Support to: FDD and TDD schemes; bandwidths: 5, 10, and 20 MHz; MIMO up to 2x2; 256 QAM modulation in downlink ; and roaming (experimental).
- **UE for NR (nr-uesoftmodem):** Support to: TDD and FDD schemes; subcarrier spacings: 15 and 30 kHz (FR1), 120 kHz (FR2); bandwidths: 10, 20, 40, 80, 100 MHz (273 Physical Resource Blocks); MIMO up to 4x2; 256 QAM modulation in downlink.
- **gNB (nr-softmodem):** Support to: TDD and FDD; subcarrier spacings: 15 and 30 kHz (FR1), 120 kHz (FR2); bandwidths: 10, 20, 40, 80, 100 MHz (273 Physical Resource Blocks); MIMO support up to 4x2; 256 QAM modulation.

In terms of deployment, on the one hand, srsRAN offers its documentation with several tutorials, use cases and configuration examples [5]. On the other hand, the OAI has a Gitlab page available at [6] containing some tutorials in the doc folder. Concerning this aspect, srsRAN is more user friendly.

Regarding platform compatibility, both run in a linux environment, with srsRAN having some precompiled binaries for some distributions such as Ubuntu, openSUSE and Debian. It is also possible to compile srsRAN in other linux distributions. As for OAI, it has some docker images that can be used, but to install it, it is necessary to compile the platform through a script available at OAI's Gitlab page.

Regarding the compatibility of SDR devices that can be used together with these platforms, the OAI is compatible with Eurecom EXMIMO II, USRP (B210/X300), BladeRF and LimeSDR for LTE. For NR SA, it is compatible with USRP (B210/X300). Concerning the srsRAN in LTE and NR NSA, it is compatible with USRP (all models), BladeRF, LimeSDR and also there is compatibility with the ZeroMQ (ZMQ) driver, which simulates a radio virtually inside a machine.

III. Implementation Methodology

To create a 5G open-source fully-functional network testbed, firstly the environment shown in Fig. 2 was set up. It was composed by 2 mini-PC's Dell OptiPlex 3070 (black boxes in Fig. 2), 2 USRPs B210 (white boxes in Fig. 2) and a **Commercial Off-The-Shelf User Equipment (COTS UE)** [Moto G200 5G] (the smartphone in Fig. 2).

The deployed network testbed had the three main components shown in Fig. 1. One of the mini-PCs acted as CN

and gNB. It had access to Internet to allow the CN to act as a gateway between the Internet and the private 5G network created using the platform. The other mini-PC was configured as an UE. Each mini-PC was connected to an USRP with two antennas VERT2450. Even though not illustrated in Fig. 2, a notebook Dell Inspiron 3501 was used to access the mini-PCs through Secure Shell Protocol (SSH).

Regarding the considered softwares for CN, gNB and UE, some combinations were evaluated.

First, the srsRAN platform was evaluated. More precisely, in this implementation the srsRAN softwares of gNB and UE, i.e., **srsgnb** and **srsUE**, were considered together with the **Open5GS CN**. Besides, in this implementation, the **COTS UE** was also tested with the implemented 5G private network.

After that, the OAI platform was evaluated. In this case, the OAI 5G/NR softwares, e.i., **OAI-5GCN**, **nr-softmodem** and **nr-uesoftmodem**, were used. As in the previous case, it was also evaluated the compatibility of a **COTS UE** with the deployed OAI network.

Finally, after having completed the platforms individual evaluations, an interoperability test between them was performed. First, it was tested the interoperability between the CN from Open5GS with the **OAI gNB**, the **OAI UE** and the **COTS UE**. In the second case, it was tested the interoperability between the **OAI-CN** with the **srsENB**, **srsUE** and the **COTS UE**.

Important to highlight that, while the mini-PC implementing a UE used either the **srsUE** or the **OAI UE** softwares, the **COTS UE** [Moto G200 5G] did not used those softwares, since it is already a complete 5G UE.

After the connection between UE and gNB was established, four tests were performed. First, the SpeedTest app was used to evaluate the download rate. After, the iPerf app was also used for similar purposes. Also, a ping was performed from the UE to the CN to evaluate the latency of the link. Finally, a video call was made with Google Meet to qualitatively evaluate the connection.

For both platforms the following configurations were considered:

- Configuration 1: 40 MHz bandwidth, 46.08 MHz sample rate using 78 band with 3.5 GHz frequency, 30 kHz subcarrier spacing and TDD scheme.
- Configuration 2: 20 MHz bandwidth, 23.04 MHz sample rate using 78 band with 3.5 GHz frequency, 30 kHz subcarrier spacing and TDD scheme.

We remark that Configuration 1 was not used to evaluate the platforms UEs. It was used to evaluate only the platforms CN and gNB with the **COTS UE**, since, as already mentioned, the **srsUE** only supports bandwidths up to 20 MHz.

IV. Results and discussions

The installation procedure of the **srsgnb** was simple. An installation guide [7] explaining the installation process was used. It describes in a beginner-friendly way how to proceed. The **srsgnb** can be installed on 3 linux system environments, i.e., Ubuntu, Arch Linux and Fedora. It is possible to adapt it for other linux environments too.

As already mentioned, we needed a third-party CN, i.e., the **Open5GS CN**. However, its installation was harder than the installation of the **srsgnb**. One of the main reasons for this is that the **Open5GS CN** documentation is harder to follow/understand. Furthermore, the documentation focuses on Ubuntu, lacking information regarding the installation in other operational systems.

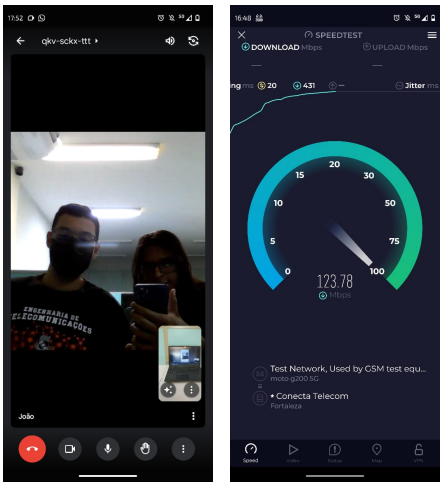
The adopted operational system was an Arch Linux, so parts of the installation needed to be changed. The most important adjustment made was to create a script file with the processes of the core in Arch Linux. Differently from the **srsgnb**, the **Open5GS CN** installation guide is confusing for beginners.

After having installed the gNB and the CN, the network was ready to be tested by using a **COTS UE** or a third-party UE. To run the **srsENB** a configuration file was created, based on the configuration file already given by the software. This file was dependent of the adopted USRP. For editing this file, a guide for each of the configurations was available at [8]. Moreover the lines of the file had comments explaining each parameter, which made it easier to edit it.

Regarding OAI platform installation, some difficulties were faced during the installation process. For **OAI gNB** installation, adjustments were made for the installation of the main dependencies, for the same reason as for the srsRAN platform (the adaptation for the installation on Arch Linux). An example of problem that we faced was during the installation of the libraries Linear Algebra Package (LAPACK) and Basic Linear Algebra Subprograms (BLAS). Both of them could not be properly installed. To overcome this issue, a solution was to use another library called Open Basic Algebra Subprograms (OpenBLAS). This library is an open-source optimized implementation of BLAS and LAPACK [9]. However, the library complete installation (including compilation process) took a long time, i.e., 8 hours. For time optimization, the compiled files were recorded in a folder to be reused in possible future re-installations.

In OAI, installing the CN was easier than installing the gNB. No documentation was found regarding the gNB installation procedure. We faced similar issues during the **OAI UE** setup. The documentation is a little confusing sometimes, since they are all placed in a repository [6], and they require more time to assemble and understand. Also, some configuration files are coded in C, which means that, to change them, you have to recompile the platform, which takes time.

Regarding the tests with the platforms UEs and the **COTS UE**, both platforms performed worse with their own UEs compared to using **COTS UE**. More specifically, configuring the **COTS UE** was easier since we only needed to configure a SIM card and add its International Mobile Subscriber Identity (IMSI) number to the CN. Moreover, the connection between the **COTS UE** and the gNB was also more stable than the connection between an UE of a given platform and its own gNB. This can be partially explained by the fact that the **COTS UE** hardware was superior to the one used with the **srsUE** and **OAI UE** softwares. Thus, even though we tested the platforms UEs, the **COTS UE** was used to obtain the network Key Performance Indicators (KPIs) presented in the following.



(a) VideoCall with Google Meet. (b) SpeedTest.

Fig. 3. Screenshots of the tests with Google Meet and SpeedTest, both conducted with OAI configuration 1.

TABLE I

AVERAGE DOWNLOAD RATE ACHIEVED BY THE **COTS UE** IN EACH EVALUATED SETUP, MEASURED BY SPEEDTEST.

Configuration	CN	OAI gNB	srsRAN gNB
Configuration 1	OAI-CN	123.78 Mbps	59.61 Mbps
Configuration 1	Open5GS	116.19 Mbps	62.69 Mbps
Configuration 2	OAI-CN	52.04 Mbps	17.70 Mbps
Configuration 2	Open5GS	52.63 Mbps	17.24 Mbps

Fig. 3 presents screenshots of Google Meet and SpeedTest during the tests. Regarding the video call quality, illustrated in Fig. 3a, on the one hand, with OAI, for a distance between UE and gNB of up to 10 meters, the video call had a good quality with a sharp image and without lags or freezing. On the other hand, with srsRAN, the video call did not have a good quality. The connection was unstable with the image frequently freezing.

Concerning the tests with SpeedTest, Table I presents the average download rate achieved by the **COTS UE** in each evaluated setup presented in Section III. These download rates are average of the download rates measured by the SpeedTest.

Analyzing the results presented in Table I, we can see that for a given configuration and gNB, the download rate is similar for both CNs. The main difference appears when comparing gNBs of different platforms. This can be explained by the fact that **OAI gNB** supports 256 QAM, while the **srsENB** only supports up to 64 QAM.

Regarding the tests with iPerf, Fig. 4 presents an example of the temporal evolution of the throughput for both SDR platforms with configuration 1, measured by iPerf. The throughput was measured at each second. The difference between the throughput achieved with each one of the platforms is highlighted. As already mentioned, it was mainly due to the difference of the QAM order used. The downward cycle of the throughput of the OAI platform, shown in Fig. 4, can be explained by the difference in the data capture threshold and the GNB output threshold, as the GNB output being lower than the data capture.

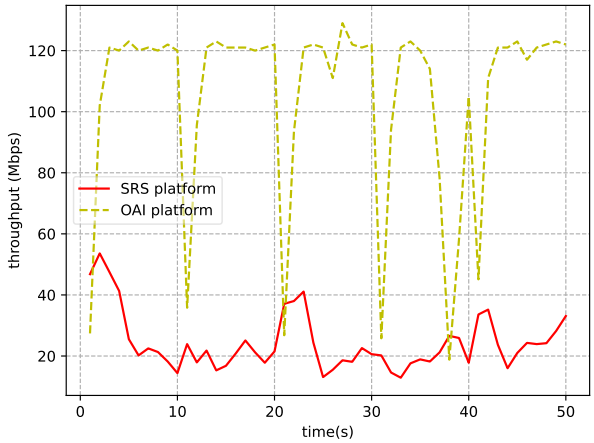
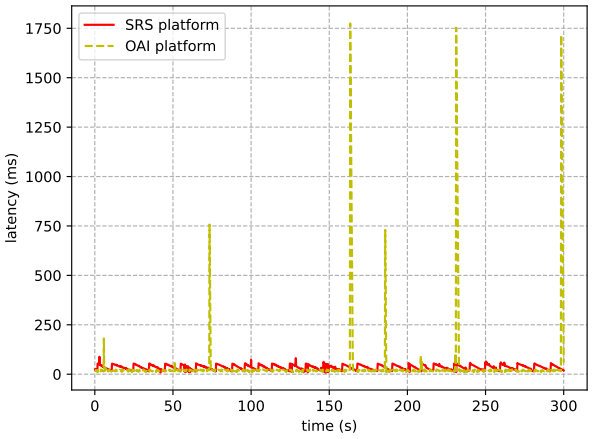
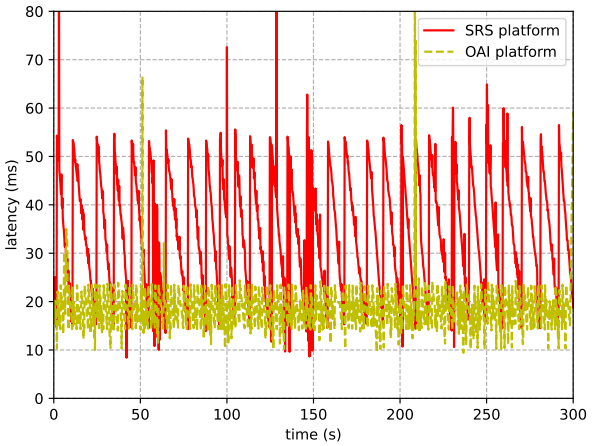


Fig. 4. Example of throughput temporal evolution measured by iPerf.



(a) Complete Y-axis.



(b) Zoom in Y-axis.

Fig. 5. Example of latency temporal evolution for both platforms with configuration 1.

Similarly, Fig. 5 presents the temporal evolution of the latency for both SDR platforms with configuration 1. Figs. 5a and 5b refer to the same case, the difference is that Fig. 5b presents a zoom in the Y-axis. The measurements were performed at each 0.2 seconds. Comparing srsRAN andOAI in Fig. 5b, we can see the the OAI average value is lower

than the srsRAN average value. However, the OAI presented spikes in Fig. 5a. The spikes were due to a refresh of the network, instructed by the gNB to occur approximately at each 75 seconds. These spikes can be explained by a configuration made by the developers of the OAI platform, so that the GNB can have its memory cache cleared and store new network information and perform operations, since the platform is coded in order to perform recompilations while running.

Also, notice in Fig. 5, the sawtooth pattern of the srsRAN curve. This can be explained by the fact that, in srsRAN Project v.23.3, some Radio Resource Control (RRC) procedures are not implemented, such as connection reestablishment [10]. The lack of these procedures have as consequence the sawtooth pattern observed in the latency curve. More precisely, after a certain period without activity, the **COTS UE** entered in idle mode. When traffic arrived to it, it needed to initiate a new connection, which takes more time than just performing a connection reestablishment procedure, which was not available.

V. Conclusions and future perspectives

The present paper presented our main findings related to our experience in deploying and using an experimental 5G testbed. Two different SDR platforms were evaluated, i.e., the srsRAN and the OAI.

Overall, we were able to use both of them and they worked with a **COTS UE**, which means that researchers and developers can use both of them to deploy their own 5G testbeds that can be used for different purposes.

More specifically, based on our evaluations, the srsRAN fits better for beginners. It is easier to use, more intuitive and have a better documentation. However, for researchers and developers seeking for a more complete platform, OAI is probably the best choice. Its learning curve takes more time, but rewards the developer seeking for a more flexible network.

As a future perspective of this work, one can use the 5G testbed that was deployed to implement applications that can explore the advantages of a private 5G network. For example, connecting 5G modules to security cameras in order to transmit in real time to a central unit the images that will be used as input of a Machine Learning (ML) solution to detect undesired events. Another possibility is using the 5G testbed to evaluate solutions that improves the network itself, for example to test radio resource management algorithms.

References

[1] K. Kaur, V. Mangat and K. Kumar, "Architectural Framework, Research Issues and Challenges of Network Function Virtualization", *2020 8th International Conference on Reliability, Infocom Technologies and Optimization (Trends and Future Directions) (ICRITO)*, Noida, India, 2020, pp. 474-478, doi: 10.1109/ICRITO48877.2020.9197802.

[2] S. Katz and J. Flynn, "Using software defined radio (SDR) to demonstrate concepts in communications and signal processing courses," *2009 39th IEEE Frontiers in Education Conference*, San Antonio, TX, USA, 2009, pp. 1-6, doi: 10.1109/FIE.2009.5350716.

[3] Wilker de O. Feitosa, Ruan A. da Silva, Victor F. Monteiro and Francisco R. P. Cavalcanti, "RSRP Prediction on LTE Network Testbed Using a Software Defined Radio (SDR) Platform", *XL Simpósio Brasileiro de Telecomunicações e Processamento de Sinais (SBrT2022)*, Santa Rita do Sapucai, Brazil, 2022, doi: 10.14209/sbrt.2022.1570814230.

[4] SRSRAN PROJECT, "Running srsRAN Project", Software Defined Radio Radio Acess Network Project(srsRAN Project). Endereço: `https://docs.srsran.com/projects/project/en/latest/user_manuals/source/running.html#manual-running` (Acesso em: 15/05/2023).

[5] SRS, "Feature List", Software Radio System(SRS). Endereço: `https://docs.srsran.com/projects/project/en/latest/general/source/2\_feature\_list.html` (Acesso em 05/06/2023).

[6] OAI, "OpenAirInterface documentation overview", Open Air Interface(OAI). Endereço: `https://gitlab.eurecom.fr/oai/openairinterface5g/-/tree/develop/doc` (Acesso em 05/06/2023).

[7] SRSRAN PROJECT, "Installation Guide", Software Defined Radio Radio Acess Network Project(srsRAN Project). Endereço: `https://docs.srsran.com/projects/project/en/latest/user_manuals/source/installation.html#manual-installation` (Acesso em: 15/05/2023).

[8] SRSRAN PROJECT, "Configuration Reference", Software Defined Radio Radio Acess Network Project(srsRAN Project). Endereço: `https://docs.srsran.com/projects/project/en/latest/user_manuals/source/config_ref.html` (Acesso em 15/05/2023).

[9] OPENBLAS, "OpenBLAS Wiki", Open Basic Linear Algebra Subprograms(OpenBLAS). Endereço: `https://github.com/xianyi/OpenBLAS/wiki` (Acesso em 15/05/23).

[10] SRSRAN PROJECT, "Release Notes", Software Defined Radio Radio Acess Network Project(srsRAN Project). Endereço: `https://docs.srsran.com/projects/project/en/latest/general/source/5_release_notes.html` (Acesso em 15/05/2023).