Received November 1, 2021, accepted December 15, 2021, date of publication December 16, 2021

Digital Object Identifier 10.46470/03d8ffbd.4ccb7950

A Comprehensive Tutorial on How to Practically Build and Deploy 5G Networks Using Open-Source Software and General-Purpose, Off-the-Shelf Hardware

SADIQ IQBAL[1], JEHAD M. HAMAMREH[2]

[1]WISLAB, Department of Electrical and Computer Engineering, Antalya Bilim University,Antalya, Turkey (e-mail: sadiq.iqbal@std.antalya.edu.tr)
[2]WISLAB, Department of Electrical and Electronics Engineering, Antalya Bilim University, Antalya, Turkey (e-mail: jehad.hamamreh@antalya.edu.tr)

Corresponding author: S. Iqbal and J. M. Hamamreh (emails: sadiq.iqbal@std.antalya.edu.tr, jehad.hamamreh@gmail.com, Web: https://wislab.researcherstore.com/solutions).

WISLAB (wislabi.com/solutions) offers solutions for building and deploying fully secure, cloud-based, and low-cost end-to-end 4G/5G networks along with providing consultations on helping companies reduce their networks CAPEX/OPEX cost and determine which solutions are best suited for their needs and use cases.

ABSTRACT 5G technology is the cornerstone for new services like eMBB, uRLLC, and mMTC, thus it should be globally available and affordable to make these services a reality. Yet most vendors responsible for the manufacturing of 5G hardware equipment and software still utilize and adopt the classical closed radio access network (RAN) concept. Consequently, this forces mobile operators, who want to deploy such equipment to build 5G cellular networks, to pay a hefty, unreasonable amount of money and still end up in vendor lock-in, which limits them from making personal changes in the network architecture or using different equipment and software from other vendors. This generally means that operators do not have complete control over the network, which they have paid for. To tackle this challenge, a new concept called Open RAN has recently emerged, which has attracted significant attention by industry leaders due to its very exciting features like the ability to use open-source software, general-purpose hardware, having the hardware separated from the software while maintaining full transparency, and interoperability. Motivated by the many features and benefits enabled by Open RAN, there is a huge need for an educational comprehensive step-by-step tutorial on how an individual or company can practically build and deploy a 5G network using open-source software and commercial off-the-shelf, general-purpose hardware. In this tutorial, we discuss almost everything related to the history and background of 5G companies and manufacturers, legacy RAN solutions and equipment, different ways and approaches to build a 5G network including RAN, CORE, and EDGE frameworks, open-source software, and generic hardware. After this, we illustrate in detail the steps of how to build and set up a 5G network using srsRAN with LimeSDR, and Raspberry Pi 4. In the end, we list the best possible PCs and software-defined radio (SDR) combinations that can be extremely helpful in building 5G networks.

INDEX TERMS 4G, 5G, 6G, radio access networks, RANs, Closed RAN, Open RAN, universal software radio peripheral, USRP, open-source software, software-defined radio, SDR, srsRAN, 5G Non stand alone.

I. INTRODUCTION

5G is a revolutionary mobile communication technology that has ushered in the 4th Industrial Revolution with the potential to change the society and economy of the future. From the beginning of 2010, the International Telecommunication Union (ITU) began researching 5G mobile communication technology to prepare for the expected data explosion after 2020 and to meet various service requirements for future mobile communications [1]. After several years of research, three service scenarios of

5G are available: 1) eMBB (enhanced mobile broadband), which provides data rates up to 20Gbps, much faster than traditional mobile technologies; 2) uRLLC (ultra-reliable low latency communication), which aims to minimize the latency to 1ms or lower; and 3) mMTC (massive machine type communication), which supports up to 1 million connections per 1 GHz [1]. The emergence of 5G created enhanced 5G-convergence services and integrated the existing industries more deeply with mobile telecommunication. Adopting 5G technology along with the ICT innovation services, such as self-driving cars, smart factories, personal drones, and remote healthcare, are expected to experience tremendous changes in service paradigms, leading to new markets [1].

KISDI, Korea's prestigious ICT state-run think tank, predicts that the annual growth in major-related industries of 5G will be 43.3 percent and that 1,161 trillion win in global markets in 26 years will be created. IHS Markit, a global market research firm, estimates that 5G will account for 4.6 percent of the world's total production in the future [1]. Previous generations of mobile networks 1G, 2G, 3G, and 4G all led to 5G, which is offering to provide more connectivity than what is available right now [2]. Through a landmark 5G Economy study, it is found that 5G's full economic effect will likely be realized across the globe by 2035, supporting a wide range of industries and potentially enabling up to 13.1 trillion US dollars worth of goods and services [2].

More importantly, 5G is designed to deliver peak data rates up to 20 Gbps based on IMT-2020 requirements. Qualcomm Technologies' flagship 5G solution, the Qualcomm Snapdragon X65 is designed to achieve up to 10 Gbps in downlink peak data rates. But 5G is about more than just how fast it is [2], where in addition to providing higher peak data rates, 5G is designed to provide much more network capacity by expanding into new spectrum bands, such as mmWave. 5G can also deliver much lower latency for a more immediate response and provide an overall more uniform user experience so that the data rates stay consistently high even when users are moving around [2]. This clearly shows the importance of 5G due to the significant change it will bring globally, which will lead us to think about how we can utilize the available resources to build 5G base stations and networks in an efficient and affordable manner, which can make users happy with the service and its cost, whereas companies satisfied with the revenue and return on investment (ROI).

Having understood the importance of 5G and the value it offers to the world, it is very crucial to know about the current big 5G companies, players, and what they are doing so that we can compare the limitations and challenges these companies are currently facing with the benefits and solutions we can get by building a 5G network. The companies leading 5G research and development (RD) are Samsung, Huawei, Nokia, LG, Ericsson, Qualcomm, ZTE, Orange, Verizon, ATT, NEC Corporation, and Cisco. **Samsung** started researching 5G technology in 2011. In 2013, Samsung had successfully developed the world's first adaptive array transceiver technology operating in the millimeter-wave Ka bands for cellular communications [3]. The new technology sits at the core of the 5G mobile communications system and provides data transmission several hundred times faster than the current 4G networks. The company achieved a lot in the next generation of technology and can now be considered one of the leaders in the 5G domain [3].

Huawei has been pouring money into research on 5G wireless networks and patenting key technologies. The company has hired many experts from abroad to decide the technical standards for the next-generation wireless communication technology [3]. **LG** has also been one of the top 5G players in research activities, products, and patent analytics. In 2019, Bloomberg cited the 5G era as the era of LG as the company managed to ship more than 100K 5G smartphones in the Korean market [3]. The Korean company has been researching 5G for quite some time and built a reputation by getting published in many 5G related reports. LG has been one of the few companies, like Samsung and Huawei, that does not just deploy 5G networks but also build products that utilize the 5G network [3]. **Ericsson** on the other hand claims to be the only vendor currently working on all continents to make 5G a global standard for the next generation of wireless technology. Their 5G radio prototypes are the first products designed to enable operators to conduct live field trials in their network which helps operators to get a greater understanding of the potential of 5G in their networks and environments [3]. **Qualcomm**, one of the leading 5G chip makers, is also leading in the overall 5G race. While other companies are talking about 5G, Qualcomm is building the technologies. Qualcomm also disclosed their royalties' price for every 5G phone that could be up to 16.25 US dollars, threefold than Ericsson's price [3].

Nokia has also joined the race of 5G as the company is developing, researching, and partnering with other entities to render 5G communication as fast as possible. The company uses an 8000-hectare site to carry out 5G tests collaborating with Deutsche Telekom and Hamburg Port Authority for their project 5G MoNArch. The project's goal is to gain knowledge and experience from 5G networks in the real-world environment. Its industrial uses could be traffic light management, data processing from mobile sensors, and VR applications [3].

ZTE Corporation, regarded as one of the leaders in the 4G LTE, also maintains its position in 5G research and tests. Because of its performance and capabilities, the 4G network was confronted by various bottlenecks on the Internet of things. At that time, ZTE was the first company to propose its Pre5G concept and series solution [3]. **NEC** also rolled its sleeves for 5G and introduced a new business concept, "5G. A Future Beyond Imagination" to make drastic changes in the industry. According to SVP Toshimitsu Shimuzu, NEC plans to "collaboratively create new

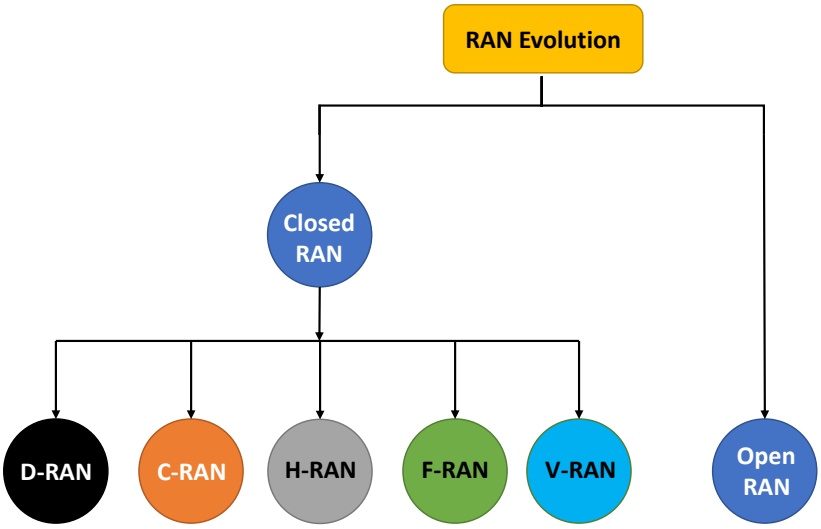

FIGURE 1. **Evolution of RAN.**

business models and services that connect information from different industries and companies by utilizing advanced information and communications technologies (ICT) that combine 5G with NEC's proprietary AI, IoT, and other digital technologies" [3].

The US telecom company **Verizon** deployed policies to render 5G for US consumers. Verizon positioned itself at the forefront of 5G technology as they are building modern infrastructure all over the country. According to them, the world needs policies for 5G deployment which they are ready to provide [3]. Mobile network operator **Orange** is also participating in the build-out of a more connected planet. The company is exploring different complementary areas such as improving mobile broadband up to 10 times faster than 4G, employing high-performance fixed Internet access to complete the fiber network where it's not available, and deploying new applications to support digital transformation across business sectors. Orange also claims that it will be a genuine multi-service network designed to adapt to a whole host of devices: smartphones mainly, but also enhanced 360° content, augmented reality, connected objects, refrigerators, and driverless cars [3].

ATT is going to be the first company to provide 5G services in the US. For achieving this feat, they deployed 5G connection in three cities: Waco TX, Kalamazoo MI, and South Bend IN [3]. **Cisco Systems** launched 5G now portfolio in MWC 2018 to support 5G automation and infrastructure, which the company will support in three primary ways: Services – enable 5G services so service providers (SPs) can make more money; Infrastructure – help build the 5G infrastructure, and Automation – Make a mass scale simpler to operate [3].The company knows the potential of 5G and plans to connect more than 30 billion devices in the next three years. From the previous discussions, we can see the hype, attention, and importance of 5G. The World Economic Forum describes 5G as the Fourth Industrial Revolution, so 5G will generate billions of dollars through unrealized revenue streams.

All the telecom giants are motivated to get a hold of the upcoming wave [3]. Recently, even **Amazon** one of the big five companies in America, lauched AWS Private 5G service through its own Amazon Web Services (AWS) adopted cloud platform. AWS Private 5G is a managed service that makes it easy to deploy, operate, and scale your own private cellular network, with all required hardware and software provided by AWS. Some use cases of AWS Private 5G are running a smart manufacturing facility, enabling business-critical applications such as augmented and virtual reality (AR/VR) applications for design engineering, image analysis during medical procedures, and autonomous guided vehicles at fulfillment centers and deliver reliable campus connectivity [4].

Now we move onto the actual question of **why do we need to build a 5G base station**. Earlier, we mentioned, the companies, vendors, and operators, working on different internal and external areas of the 5G network are building their own devices, architectures, and networks. The cost of these products is hundreds of thousands of dollars, for example, China Mobile's 2020 capital expenditure budget was 179.8 billion yuan (about 25 billion US dollars). Of this sum, the 5G-related investment plan was about 100 billion yuan (about 14 billion US dollars). This means that on average, each 5G base station costs nearly 400,000 yuan (about 57,000 US dollars) [5].

From this example alone, we can see how much the vendors and operators charge for their products. The products

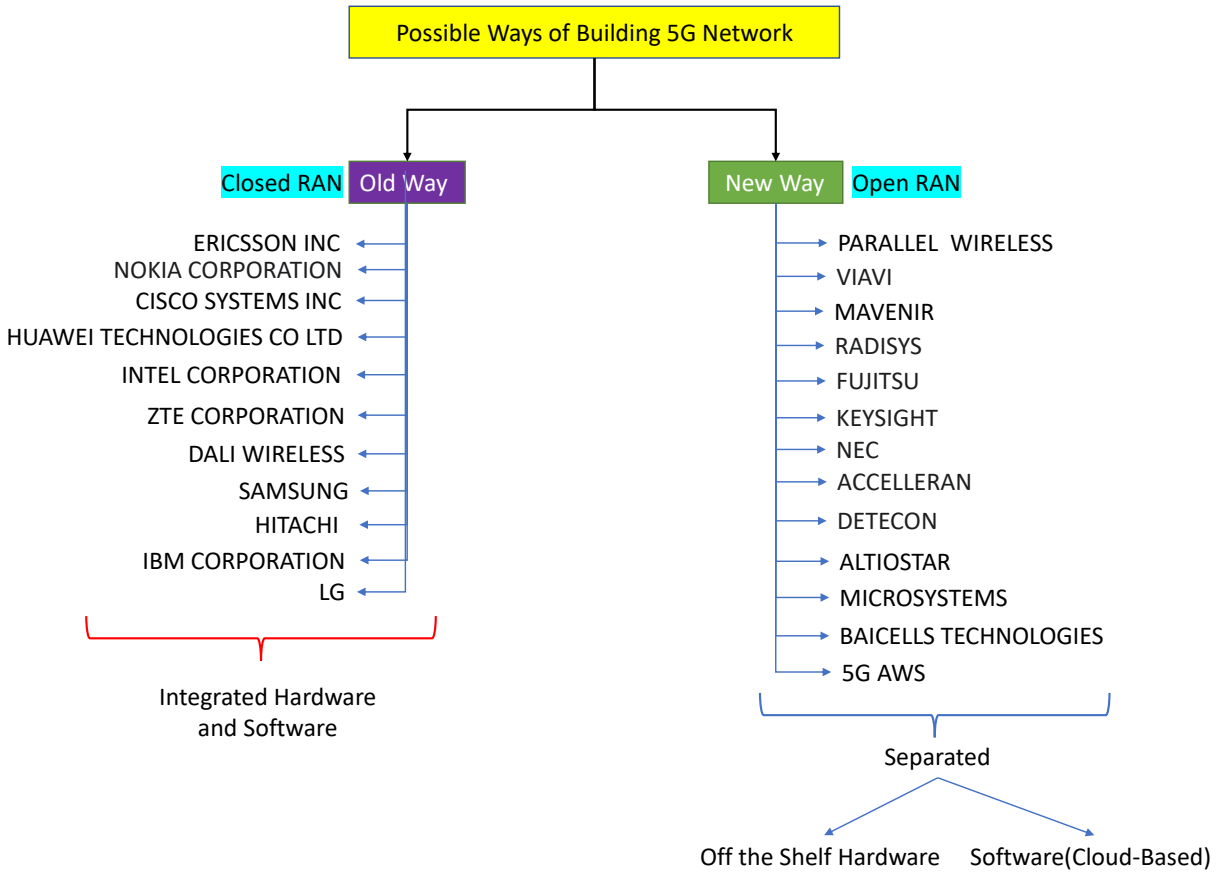

FIGURE 2. **Closed RAN vs. Open RAN with some examples on companies from each category (noting that the lists are not inclusive).**

are not available in the local market and cannot work with third-party hardware. Even the software is the proprietary product of these organizations. This indicates the limitations enforced on local consumers from the tech giants. We can ignore these limitations and build an end-to-end 5G networks using open-source software and state-of-the-art off-the-shelf SDR-RF hardware along with generic compute devices, which costs a fraction of the previously mentioned products.

II. PREVIOUS RADIO ACCESS NETWORKS (RANS) AND 5G OPEN RAN

In this section, we will highlight the evolution of RANs from Distributed RANs (D-RANs) to Cloud RANs (C-RANs), Heterogeneous Cloud RANs (H-CRANs), Fog Computing RANs (F-RANs) and virtual RAN (V-RAN) as shown in Fig. 1, to know about the previous technologies so that we have the base knowledge about how networks work and then we can focus on building a 5G network. Previous generations of cellular systems used to have a baseband unit (BBU) and remote radio head (RRH) components physically integrated and located at the bottom of a Base Station (BS) connected to a Radio Frequency (RF) antenna at the top of the tower through heavy electrical cables. This architecture presented significant RF signal propagation loss in the electrical cable feed resulting in degraded signal transmission/reception power and quality. Therefore, telecommunication operators began to adopt a separated BBU and RRH architecture based on D-RAN or just RAN [6].

What is a D-RAN? In a traditional D-RAN system, each BS is composed of two collocated components: (1) a digital unit (DU) or BBU, and (2) a radio unit (RU) or RRH. These two components connect through a Common Public Radio Interface (CPRI). The BBU is the component responsible for baseband processing that processes multiple calls and forward's traffic. The RRH is responsible for digital radio signal processing by transmitting, receiving, and converting signals. Each BS connects to the core network through a backhaul [6]. C-RANs have emerged as a centralized solution, moving the BS functionalities to the cloud to optimize the resources and improve energy efficiency. **What is a C-RAN?** The principle behind C-RAN architecture

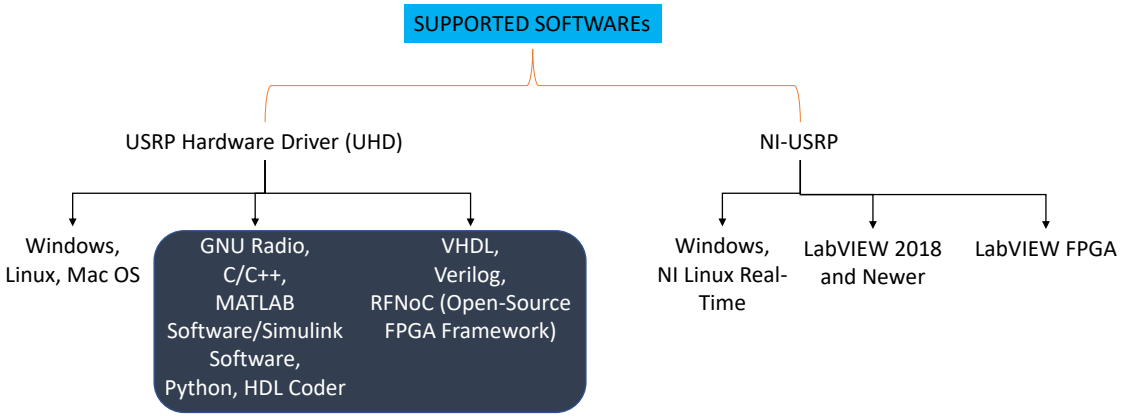

FIGURE 3. **Supported software generally used to build 5G Networks.**

is to relocate some of the cellular network functions to the cloud infrastructure. Network operators understand that the main cost of 5G is related to RAN and decided to invest in new types of open and low-cost architectures [6]. The traditional RANs have limitations, so to overcome them the RRH, and BBU functions decouple in C-RAN architectures where RRH is at the BS, the BBU in the cloud infrastructure, and we have a front-haul in between RRH and BBU. The most intensive computational tasks get performed in BBUs allocated in the cloud. In a traditional RAN architecture, the deployment and the commissioning of a new BS are expensive and time-consuming. But in C-RAN architecture, the infrastructure is easily deployed as only an RRH is installed and an associated BBU service deployed in the cloud [6]. However, C-RANs experience drawbacks such as primary user emulation attack, spectrum sensing data falsification, centralized signal processing in the cloud which can introduce the risk of higher latency, and these issues can be solved by H-CRAN and F-RAN [6].

What is a H-RAN? H-CRAN is an architecture that takes advantage of two approaches: C-RAN and Heterogeneous Networks (HetNets). In H-CRAN architectures, RRHs assume the role of low power nodes (LPNs) by performing simple functions (such as radiofrequency management and simple symbol processing). The BBU is responsible for coordination between high power nodes (HPNs) and RRHs to mitigate inter-tier interference [6].

What is a F-RAN? The C-RAN and H-CRAN architectures centralize their software process at the cloud resulting in a heavy load on the front-haul link. To mitigate this problem the F-RAN architecture based on the H-CRAN architecture is used [6]. F-RAN aims to minimize the disadvantages of C-RAN and H-CRAN. 5G networks need ultra-densified networks, device-centric architecture, specialized hardware, need to coexist with legacy infrastructures, e.g., 2G, 3G, and 4G, which increases management cost and complexity. A solution to address these factors is to implement the 5G network functions as software components [6].

What is a v-RAN? A virtual radio access network (vRAN) is a type of RAN with its networking functions separated from the hardware it runs on. The control and data planes of the vRAN are also separated as part of the virtualization. Network functions virtualization (NFV) is the practice of turning hardware-based functions into software. In an NFV architecture, the hardware is typically commercial off-the-shelf (COTS) standard hardware. A vRAN is more flexible because it also allows change without having to replace hardware throughout the entire infrastructure and all it needs is to update its software [8].

What is a Open RAN? It includes developing interoperable open hardware, software, and interfaces for cellular wireless networks that use white box servers and other standard equipment, rather than the custom-made hardware typically used in base stations [9]. Now we know about the use of conventional RAN and its types. Therefore, we mention the companies that utilize these architectures, their limits, and the disadvantages. Then compare them with the Open RAN. Afterward, we will discuss the usage of Open RAN in building a 5G Network and BS.

A. DIFFERENT RAN SOLUTIONS AND EQUIPMENT

Many custom RAN solutions and equipment are available from vendors such as ASOCS, Airspan, Altiostar, Casa Systems, Ericsson, Intel, Nokia, Parallel Wireless, Radisys, and Samsung [11]. Out of these ASOCS focuses on vRAN solutions, Airspan deals in OpenRANGE, RRUs, and DUs, Altiostar rolls out open virtualised RAN software solutions, Casa Systems manage Casa's Axyom Software platform with 5G equipment, Ericsson's RAN portfolio consists of basebands, radio processors and a radio access processing platform, Ericsson Radio System, Intel's portfolio for wireless 5G deployment solutions is centered around its Intel FGPA suite, which consists of a wide variety of configurable

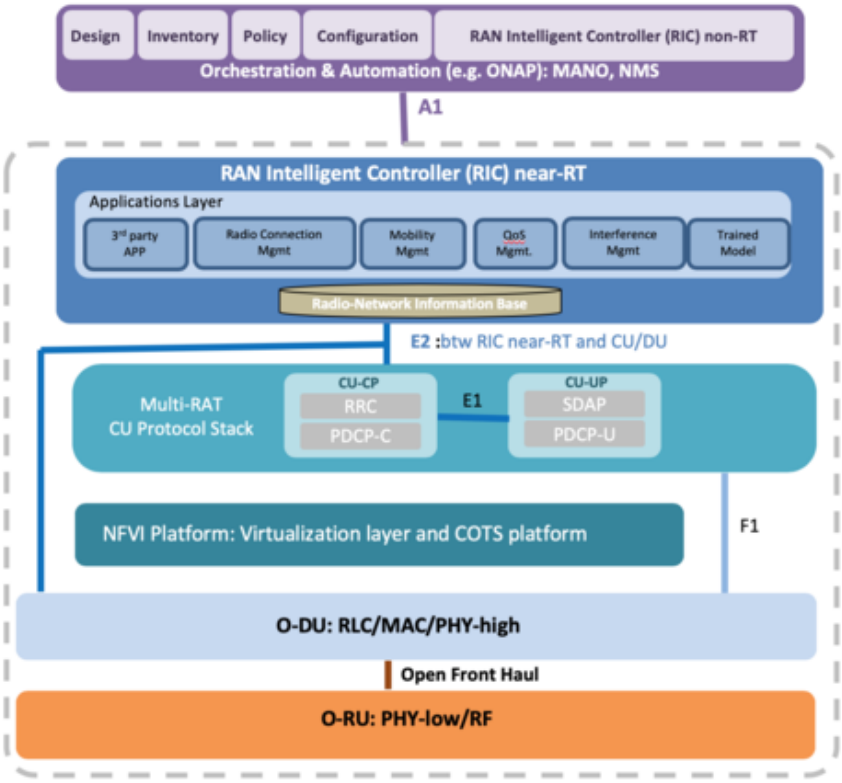

FIGURE 4. **O-RAN Architecture [7].**

embedded SRAM, high-speed transceivers, logic blocks and routing, Nokia's AirScale Radio Access portfolio supports all radio access technologies including both NSA and SA 5G networks, Nokia is also a contributor in ORAN alliance, Parallel Wireless offer Open RAN solutions like end-to-end virtualized network deployment services combined with software-defined hardware, Radisys is also a contributor in Open RAN alliance because of sharing its Open 5G Software seed code and Samsung is an active member of the Open RAN community, participating in large-scale deployment tests, conferences and most recently, commercializing a new carrier-grade, fully-virtualized 5G RAN solution [11].

However, the companies that use closed or traditional RAN will face challenges like declined sales, and equipment companies will also face challenges from new rivals that support Open RAN. The orange tech company even said that any equipment it buys in Europe will have to be O-RAN-compliant starting in 2025 [12]. Omdia, a telecommunication, media, and technology consultant, anticipates a 13% decline in 4G and 5G closed RAN products between 2020 and 2024. Ericsson and Nokia both have joined the O-RAN Alliance, a group developing specifications, and promised O-RAN products. Omdia expects Open RAN sales to soar from 252 million US dollars last year to around 3.2 billion US dollars in 2024. European operators also want authorities to cultivate a local Open RAN supply chain, ensuring Asian and US vendors are not the only alternatives to Ericsson and Nokia. All this could make for an extremely crowded 3.2 billion US dollar market [12].

So far, we have discussed 5G, its importance, tech companies that utilize Closed RAN, custom equipment, their limitations, challenges, costs, and why we need to build a 5G network. In the subsequent sections, we will mention the possible ways of building a 5G network using Closed RAN and Open RAN. Afterward, we will elaborate on different approaches of how a 5G network can be built using Open RAN and open-source software?

III. POSSIBLE WAYS OF BUILDING A 5G NETWORK

Building a 5G network usually can be categorized into two ways the classical old way and the new way, as shown in Fig. 2. The old way utilizes Closed RAN, which has coupled and integrated the hardware and software as a joint RAN solution, and the new way uses the Open RAN concept, which has separated and disaggregated hardware from software. One way of implementing Open RAN is through the disaggregation of software from hardware which allows RAN software to run on any common hardware platform such as those based on Intel x86 and ARM architectures [15]. This disaggregation also applies to other hardware components such as Field Programmable Gate Arrays (FPGAs) and Graphics Processing Units (GPUs), which opens the abstraction layer. From Fig. 2, on the left-hand side, we can see the vendors who use closed

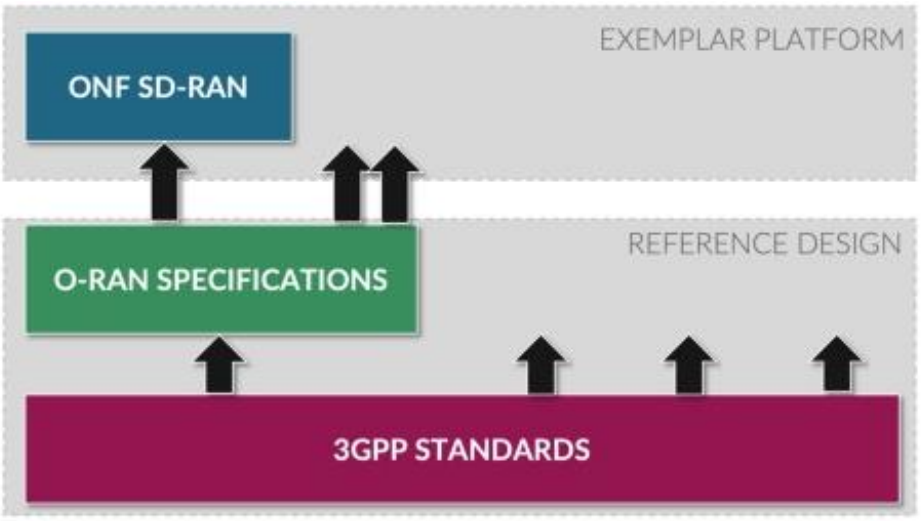

FIGURE 5. SD-RAN Architecture [10].

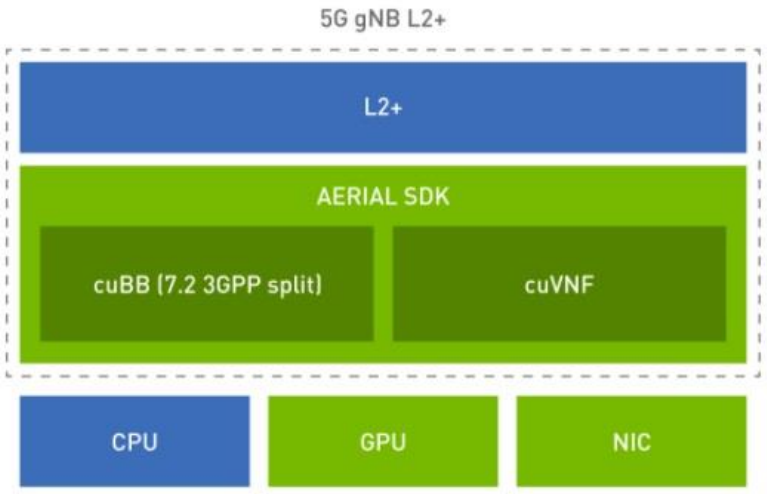

FIGURE 6. NVIDIA Aerial Architecture [13].

RAN architecture to build products that include hardware and software from the same vendor. In other words, if an operator uses vendor A for the radio, typically, it must use the baseband from vendor A as well. In addition, the software that runs on the baseband hardware does not run on another vendor's hardware. This creates a vendor lock-in due to the proprietary vendor-specific product realization of the interface specification [15]. It is a serious issue faced by individuals who do not have the resources to buy these products from the same vendor as they are costly. So, for this reason along with the need to have flexibility in scaling when compared with integrated platforms (Closed RAN), the vendors mentioned on the right-hand side of Fig. 2 support Open RAN.

IV. APPROACH TO BUILDING A 5G NETWORK USING OPEN RAN HARDWARE AND OPEN-SOURCE SOFTWARE

From Fig. 3, we can see the supported software that can generally be used with the compatible hardware to build a 5G network, such as the NI USRP LabVIEW and the USRP hardware driver (UHD). LabVIEW software adopted

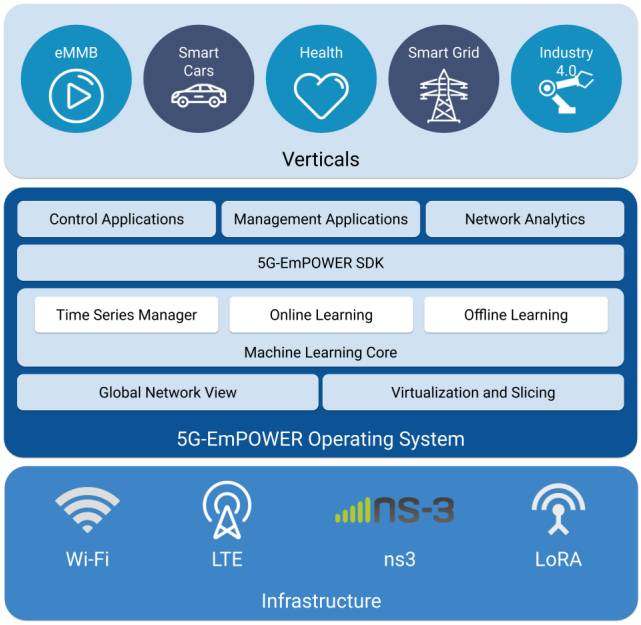

FIGURE 7. 5G-EmPOWER Operating System [14].

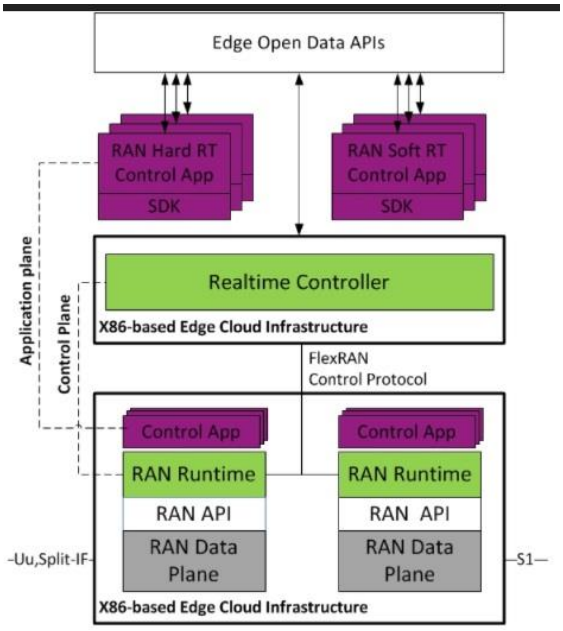

FIGURE 8. FlexRAN Infrastructure [16].

by LabVIEW users with the NI-USRP LabVIEW driver is a paid program [17]. In this way, users can buy LabVIEW software, the supported hardware to build a 5G network which is costly at the same time. On the other hand, users can build a 5G network, using USRP UHD open-source software that is a product of Ettus Research published by NI under open-source licenses [17]. The UHD driver facilitates application development on USRP hardware in C/C++, offers cross-platform support for industry development environments and frameworks, such as RF Network-on-Chip (RFNoC), GNU Radio, HDL Coder, and MathWorks MATLAB and Simulink software [17]. Now moving on to other open-source solutions and implementations for the RAN, Core and Edge frameworks such as:

A. RAN FRAMEWORKS

1) O-RAN

A key principle of the O-RAN architecture is to extend the SDN concept of decoupling the control plane (CP) from the user-plane (UP) into RAN while bringing in embedded intelligence. The first benefit decoupling offers

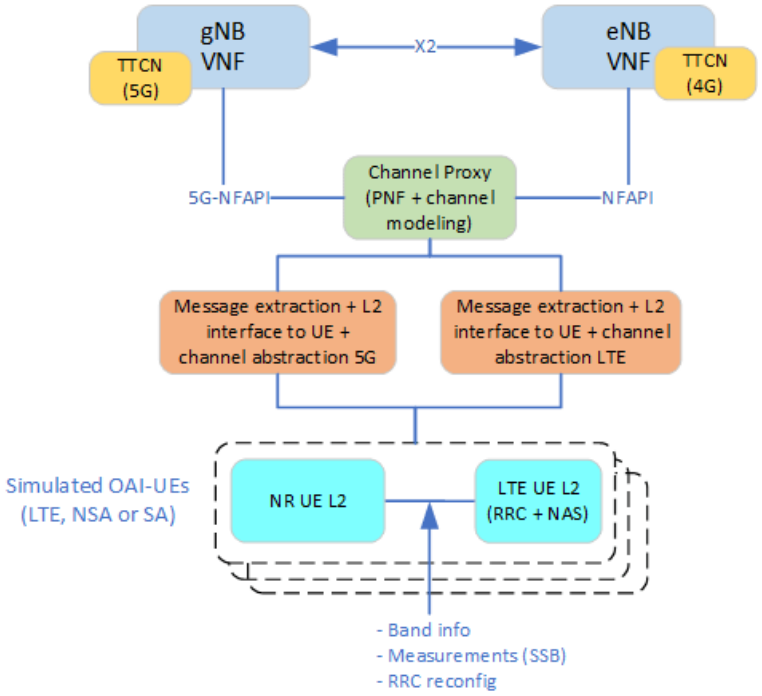

FIGURE 9. **Open-Air-Interface 5G Stack [18].**

is to allow the UP to get more standardized since most of the variability is in the CP. This allows easy-scaling and cost-effective solutions for the UP. RAN cloudification is one of the fundamental tenets of the O-RAN architecture [7]. Operators are delivering NFVI/VIM requirements to enhance virtualization platforms in support of various splits. For example, high layer split between PDCP and RLC, low layer split within PHY. The O-RAN reference architecture is built on a set of key interfaces between multiple decoupled RAN components. These include enhanced 3GPP interfaces (F1, W1, E1, X2, Xn) for true multi-vendor interoperability. To take full advantage of the economies of scale offered by an open computing platform approach, O-RAN Alliance reference designs will specify high performance, spectral, and energy-efficient white-box base station hardware. Reference platforms support a decoupled approach and offer detailed schematics for hardware and software architecture to enable both the BBU and RRU. Many components of the O-RAN architecture will be delivered as open-source, through existing communities. These components include: the RAN intelligent controller, protocol stack, PHY layer processing, and virtualization platform [7].

The O-RAN reference architecture shown in Fig. 4, is designed to enable next-generation RAN infrastructures. Empowered by principles of intelligence and openness, the O-RAN architecture is the foundation for building the virtualized RAN on open hardware, with embedded AI-powered radio control, that has been envisioned by operators around the globe. The architecture is based on well-defined, standardized interfaces to enable an open, interoperable supply chain ecosystem in full support of and complementary to standards promoted by 3GPP and other industry standards organizations [7].

2) SD-RAN

SD-RAN is building open-source components for the mobile RAN space, complementing O-RAN's focus on architecture and interfaces by building and trialing O-RAN compliant open-source components. SD-RAN is developing a near-real-time RIC (nRT-RIC) and a set of exemplar xApps for controlling the RAN. This RIC is cloud-native and builds on several of ONF's well-established platforms including the ONOS SDN Controller. The architecture for the SD-RAN nRT-RIC will leverage the O-RAN architecture and vision [10]. As illustrated in Fig. 5, ONF has started to develop an Exemplar Platform consistent with the O-RAN architecture using a specific set of implementation choices:

1) The solution will include open-source implementations of O-DU, O-CU-UP, and OCU-CP.
2) The solution will implement O-CU-UP using P4.
3) The solution will include an open-source Near-Real-Time RIC Controller implementation that is based on ONF's ONOS.
4) The solution will likely expand on the E2 interface to

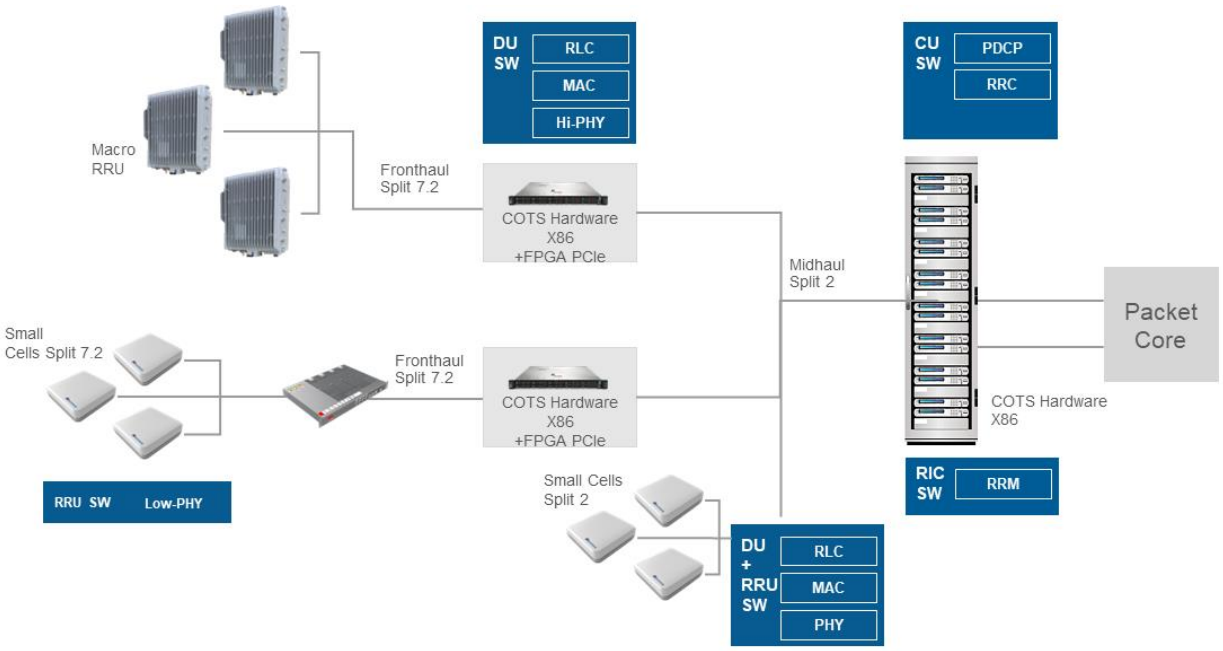

FIGURE 10. **Virtualized Open RAN Split 7.2 architecture [19].**

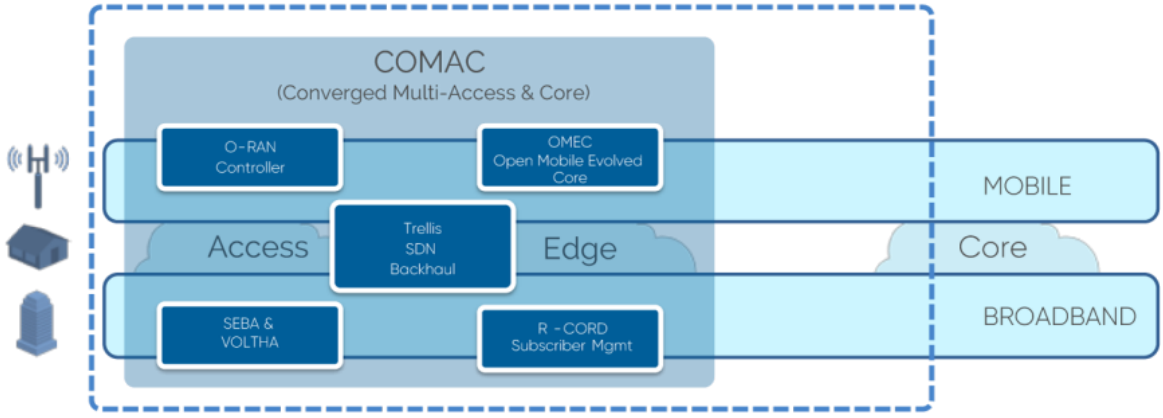

FIGURE 11. **COMAC Exemplar Platform [20].**

allow for scheduler control and network slicing and contribute this expansion back to O-RAN for inclusion in the specifications.

5) The solution will be inter-operable with third-party Rus.
6) The solution will leverage COTS and white box P4-programmable switches.
7) The solution will use Aether5 as the Virtualization Layer, VIM, and Infrastructure Management Framework.

3) NVIDIA Aerial

NVIDIA Aerial is an Application framework for building high-performance, software-defined, cloud-native 5G applications to address increasing consumer demand. Optimize your results with parallel processing on GPU for baseband signals and data flow [13]. The NVIDIA Aerial SDK package as shown in Fig. 6, simplifies building programmable and scalable software-defined 5G RAN. The Aerial SDK supports:

1) CUDA Baseband (cuBB): The NVIDIA cuBB SDK

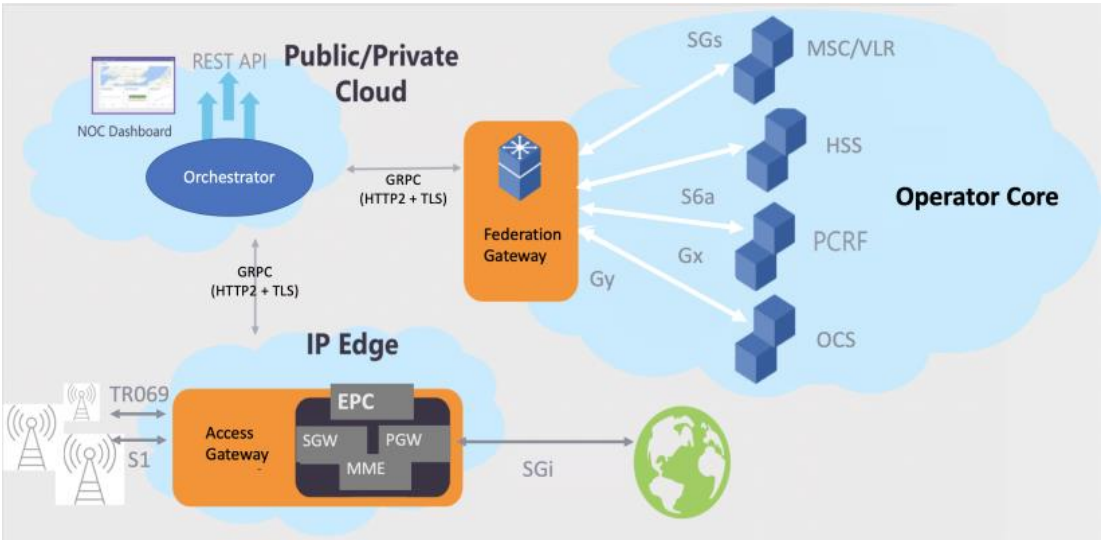

FIGURE 12. Magma Architecture [21].

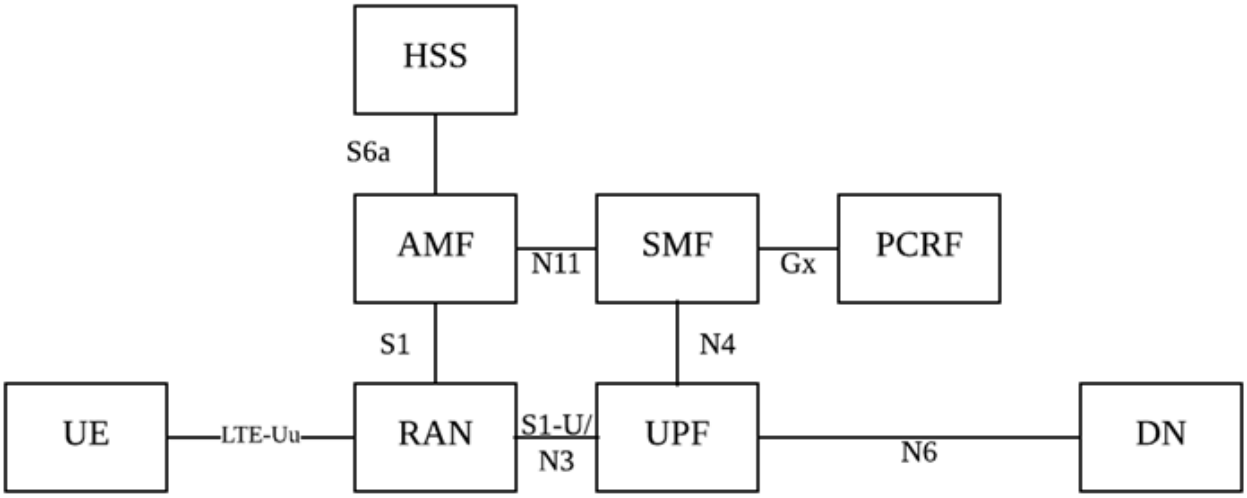

FIGURE 13. free5GC Architecture [22].

provides a GPU-accelerated 5G signal processing pipeline, including cuPHY for Layer 1 5G PHY. It delivers unprecedented throughput and efficiency by keeping all physical layer processing within the high-performance GPU memory [13].

2) CUDA Virtual Network Functions (cuVNF): The NVIDIA cuVNF SDK provides optimized input/output and packet processing, exchanging packets directly between GPU memory and GPUDirect-capable NVIDIA ConnectX-6 DX network interface cards [13].

The Aerial SDK works on COTS hardware and cloud-native platforms such as NVIDIA EGX. It's the single software programming model that goes from pico cells to centralized RAN (CRAN) to distributed RAN (DRAN). It's Kubernetes based and provides container orchestration for ease of deployment and management [13].

4) 5G-EmPOWER

5G-EmPOWER is a Software-Defined Networking Platform for 5G Radio Access Networks. Its flexible architecture provides an open ecosystem where new 5G services can be tested in realistic conditions [14]. The 5G-EmPOWER Operating System as shown in Fig. 7, consists of the following components:

1) Empower-core, the core library used to develop the 5G-EmPOWER controller.
2) Empower-runtime, the Python-based 5G-EmPOWER Controller. This allows network apps to control Wi-Fi APs and LTE eNBs using either a REST API or a Python API.
3) Empower-lvap-agent, the 5G-EmPOWER Wi-Fi

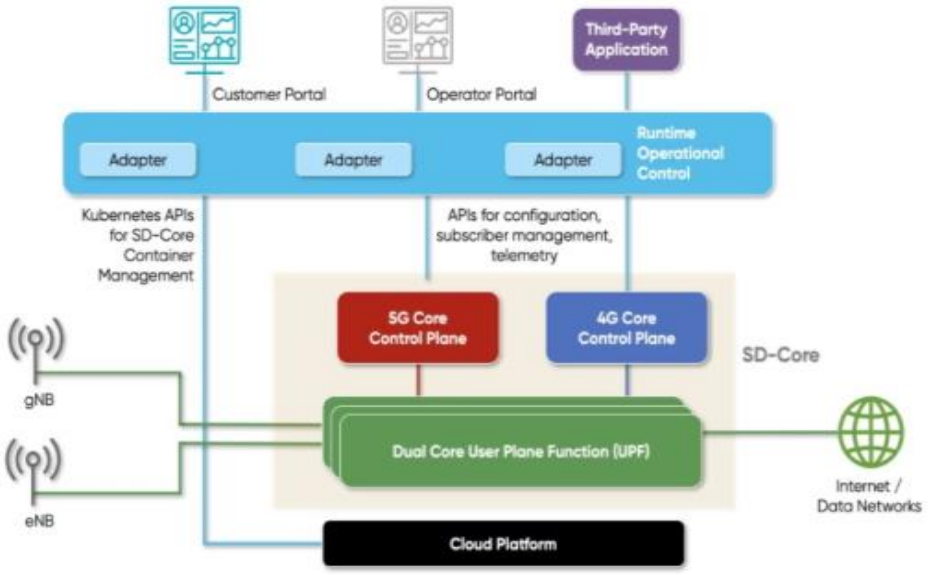

FIGURE 14. **SD-Core Architecture [23].**

agent. This agent allows controlling Wi-Fi access points using the empower-runtime. While in principle it is possible to install this agent on any Linux box you will avoid a lot of pain if you use our OpenWRT branch which includes all the necessary Kernel patches.

4) Empower-enb-agent, the 5G-EmPOWER LTE agent library. This agent allows controlling LTE eNBs using the empower-runtime. The agent can be integrated with any LTE stack however for the moment we officially support only srsLTE.
5) Empower-vbs-emulator, a basic dummy eNB implementing part of the 5G-EmPOWER southbound interface and meant to help in the development of the controller when a real eNB is not available.
6) srsLTE, a branch of srsLTE with the 5G-EmPOWER eNB agent.
7) Empower-openwrt-packages, the 'empower-lvap-agent' package for OpenWRT 19.07.
8) Empower-openwrt, a branch of OpenWRT 19.07 including some Kernel patches necessary for the correct operation of the 'empower-lvap-agent'.
9) Empower-config, the configuration files for the Wi-Fi WTPs.
10) Docker, some docker files of the main 5G-EmPOWER components.

5) FlexRAN

The FlexRAN platform is made up of two main components: the FlexRAN Service and Control Plane and FlexRAN Application plane. The FlexRAN service and control plane follows a hierarchical design and is composed of a Real-time Controller (RTC) that is connected to several underlying RAN runtime, one for each RAN module (e.g., one for monolithic 4G eNB, or multiple for a disaggregated 4G and 5G) [16]. The control and data plane separation as shown in Fig. 8, are provided by the RAN runtime environment which acts as an abstraction layer with the RAN module on one side and RTC and control apps on the other side. RAN control applications can be developed both on the top of the RAN runtime and RTC SDK allowing to monitor, control, and coordinate the state of RAN infrastructure [16].

1) RAN Control Data Plane Separation: FlexRAN decouples the RAN control and data plane with several benefits, including reducing the complexity of developing new control solutions; promoting openness and innovation by allowing operators to open their RAN service environment to authorized third parties to rapidly deploy innovative applications and services for mobile subscribers, enterprises, and vertical segments [16].
2) Centralized Real-time Control: FlexRAN consolidates the control plane into a single logically centralized controller, enabling easier coordination among base stations, effectively simplifying the development of more sophisticated control applications [16].
3) Abstraction and Virtualized Control Functions: FlexRAN allows the flexible and programmable control of the underlying RAN infrastructure through the introduction of RAN API and virtualized control functions that have a modular structure and well-defined interfaces and are responsible for performing the various control operations of the base station [16].
4) Control Delegation Policy Reconfiguration: The virtualized control functions of FlexRAN are exploited through a set of mechanisms designed to allow the

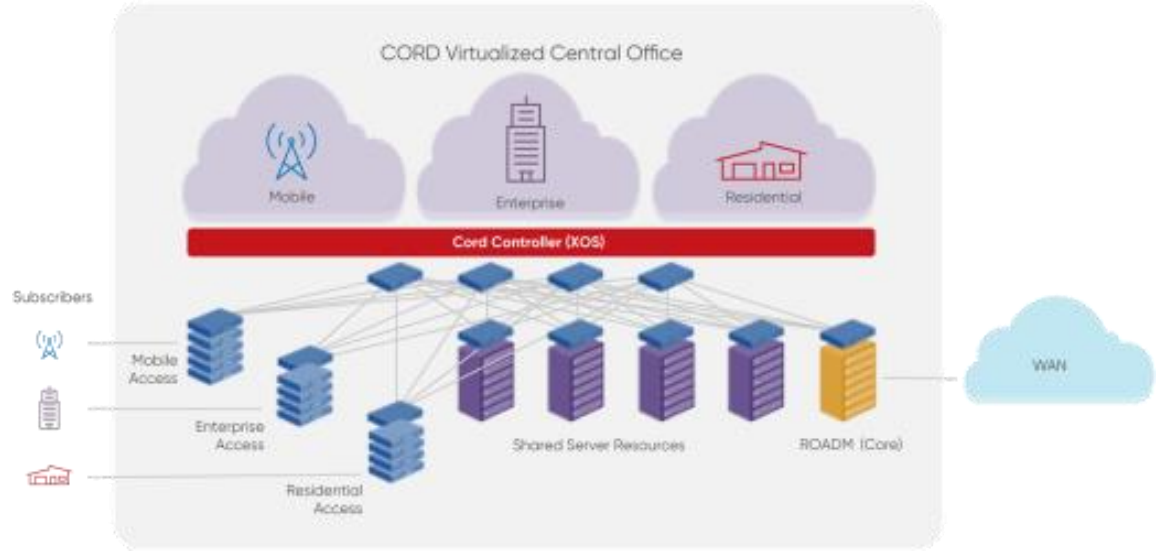

FIGURE 15. **CORD Architecture [24].**

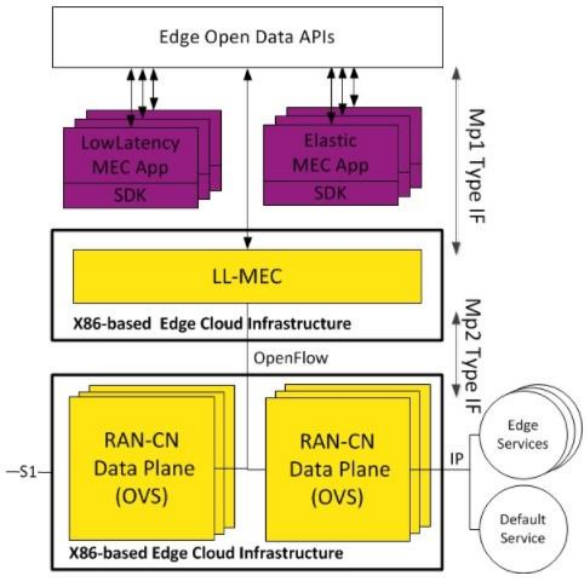

FIGURE 16. **LL-MEC Platform [25].**

delegation of control functions, like schedulers and mobility managers, from the master controller to the base stations at runtime and the reconfiguration of their behavior and parameters on the fly simply and seamlessly [16].

6) Open-Air-Interface 5G Radio Access Network Project

The OAI 5G stack supports the following Non-Stand-Alone (NSA) gNB, Stand-Alone (SA) gNB, and 5G NSA SA UE.

1) L1-simulation framework: The RF simulator replaces the radio board with software (TCP/IP) communication to make possible all functional tests without an RF board. The OAI gNB and the OAI UE communicate as if there were an RF interface between them, but without any real-time clock constraints. The I/Q samples can be transmitted over a radio channel simulator. The RF simulator also supports MIMO [18].
2) L2-simulation framework: Using actual radios or even RF simulators does not allow testing a large number of UEs. Therefore, as shown in Fig. 9, the L2 simulator offers the possibility of connecting the OAI UE with the OAI xNB (eNB in LTE and gNB in 5G) through the nFAPI interface defined by the Small Cells Forum

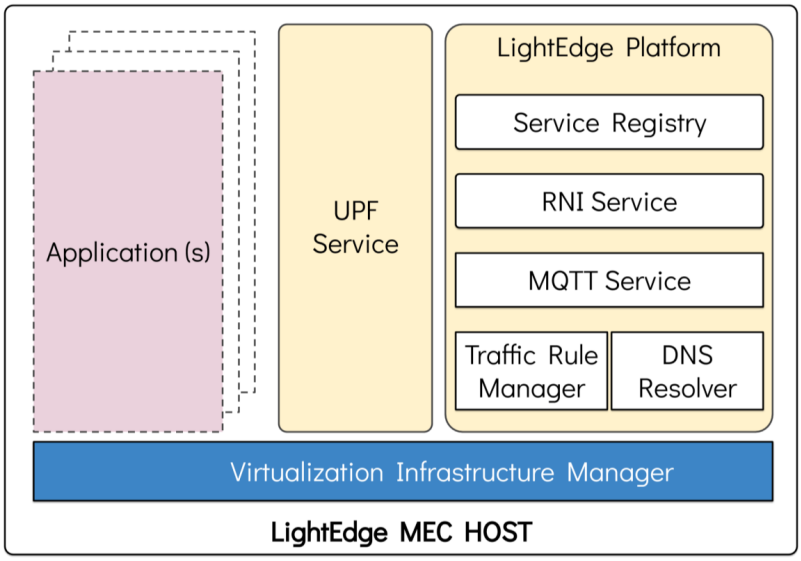

FIGURE 17. **LightEdge-MEC Platform [26].**

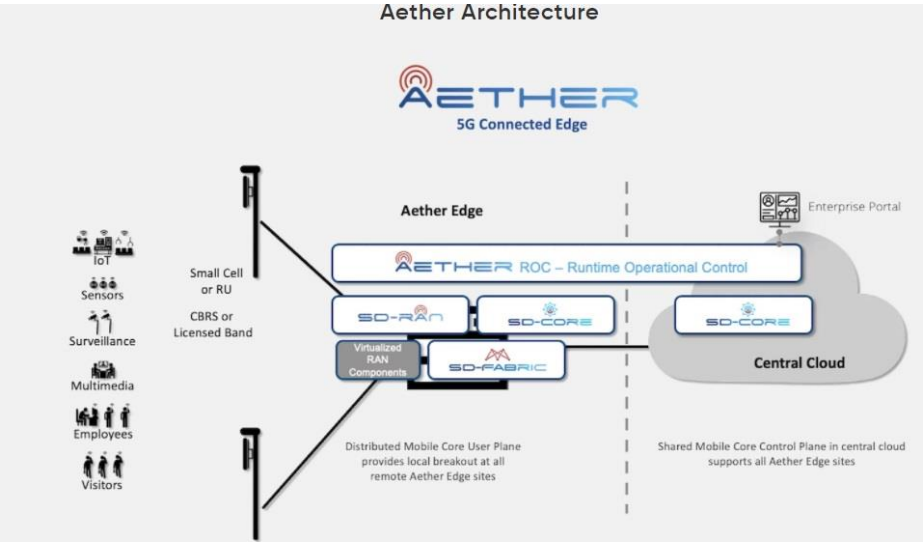

FIGURE 18. **Aether Architecture [27].**

(SCF). nFAPI splits the xNB into a MAC entity and a PHY entity. In OAI, the xNB MAC connects through the nFAPI interface to a channel proxy that simulates the channel and allows the connection of many UEs to the MAC stub. Each UE is the simulated OAI UE that connects to the proxy [18].

7) openLTE

OpenLTE is an open-source implementation of the 3GPP LTE specifications. The focus is on the transmission and reception of the downlink. Currently, octave code is available for the test and simulation of downlink transmit and receive functionality. In addition, GNU Radio applications are available for downlink to transmit and receive to and from a file. The current focus is on extending the capabilities of the GNU Radio applications [28].

8) Open vRAN

Mavenir Open vRAN solution brings increased business agility with network elasticity and flexibility in radio access networks with the world's first fully containerized, virtualized Open RAN Split 7.2 architecture [19]. This allows

the Mavenir customers to break free of vendor lock-in with the evolved Open RAN architecture, designed with cloud-native virtualization techniques, which enables RAN to flex and adapt based on usage and coverage. Mavenir's Open vRAN O-RAN compliant, fully containerized true Open RAN solution works on open interfaces supporting O-RAN Split 7.2x and Split 2 as shown in Fig. 10,. It further disaggregates into Distributed unit (DU) and centralized unit (CU). These entities work as a containerized network function (vDU, vCU CNF) running on Commercial Off the Shelf (COTS) hardware. Designed to support multiple Fronthaul splits simultaneously [19].

B. CORE FRAMEWORKS

1) COMAC

The Converged Multi-Access and Core (COMAC) open-source project, paired with the COMAC RD, has been launched to bring convergence to Operators' mobile and broadband access and core networks. It will build upon a suite of ONF projects that are part of the CORD project umbrella. By leveraging and unifying both access and core projects, COMAC will enable greater infrastructure efficiencies, as well as common subscriber authentication and service delivery capabilities [20].

Optimized for 5G deployments, the COMAC Exemplar Platform as shown in Fig. 11, will leverage SDN and cloud principles to disaggregate elements and will be built on a microservice architecture, so operators can dynamically place elements where they best serve their needs. Access, edge, core, or public clouds work together in a centralized and coordinated way to deliver exceptional user experience while leveraging common infrastructure and public cloud economics [20].

2) Magma

Magma is an open-source software platform that gives network operators an open, flexible, and extendable mobile core network solution. The high-level Magma architecture Ais shown in Fig. 12. Magma is designed to be 3GPP generation and access network (cellular or WiFi) agnostic. It can flexibly support a radio access network with minimal development and deployment effort [21]. Magma has three major components:

1) Access Gateway: The Access Gateway (AGW) provides network services and policy enforcement. In an LTE network, the AGW implements an evolved packet core (EPC), and a combination of an AAA and a PGW. It works with existing, unmodified commercial radio hardware [21].
2) Orchestrator: Orchestrator is a cloud service that provides a simple and consistent way to configure and monitor the wireless network securely. The Orchestrator can be hosted on a public/private cloud. The metrics acquired through the platform allow you to see the analytics and traffic flows of the wireless users through the Magma web UI [21].
3) Federation Gateway: The Federation Gateway integrates the MNO core network with Magma by using standard 3GPP interfaces to existing MNO components. It acts as a proxy between the Magma AGW and the operator's network and facilitates core functions, such as authentication, data plans, policy enforcement, and charging to stay uniform between an existing MNO network and the expanded network with Magma [21].

3) free5GC

The free5GC is an open-source project for 5th generation (5G) mobile core networks. The goal of this project is to implement the 5G core network (5GC) defined in 3GPP Release 15 (R15) and beyond. The implementation is based on nextEPC, implementation of 4G EPC R13. That is, the MME, SGW, and PGW are migrated into 5GC. Because commercial 5G User Equipment (UE) and base station (gNB) are not on the market yet, the free5GC uses 4G protocols to communicate with 4G UE and 4G base station (eNB) as shown in Fig. 13. Thus, the authentication protocol is still based on 4G [22].

4) SD-Core

The SD-Core project is a 4G/5G disaggregated mobile core optimized for public cloud deployment in concert with distributed edge clouds and is ideally suited for carrier and private enterprise 5G networks. It exposes standard 3GPP interfaces enabling the use of SD-Core as a conventional mobile core [23]. SD-Core is a flexible, agile, scalable, and configurable dual-mode 4G/5G core network platform that builds upon and enhances ONF's OMEC and free 5GC core network platforms to support LTE, 5G NSA and 5G SA services as shown in Fig. 14. The SD-Core control plane provides the flexibility of simultaneous supports for 5G standalone, 5G non-standalone, and 4G/LTE deployments. SD-Core provides a rich set of APIs to Runtime Operation Control (ROC). Operators can use these APIs to provision the subscribers in the mobile core; control runtime configuration of network functions, and provide telemetry data to third party applications [23].

C. EDGE FRAMEWORKS

1) CORD

The edge of the operator network (such as the central office for telcos and the head-end for cable operators) is where operators connect to their customers. CORD is a project intent on transforming this edge into an agile service delivery platform enabling the operator to deliver the best end-user experience along with innovative next-generation services [24]. The CORD (Central Office Re-architected as a Datacenter) platform as shown in Fig. 15, leverages SDN, NFV, and Cloud technologies to build agile datacenters for the network edge. Integrating multiple open-source projects, CORD delivers a cloud-native, open, programmable, agile platform for network operators to create innovative services

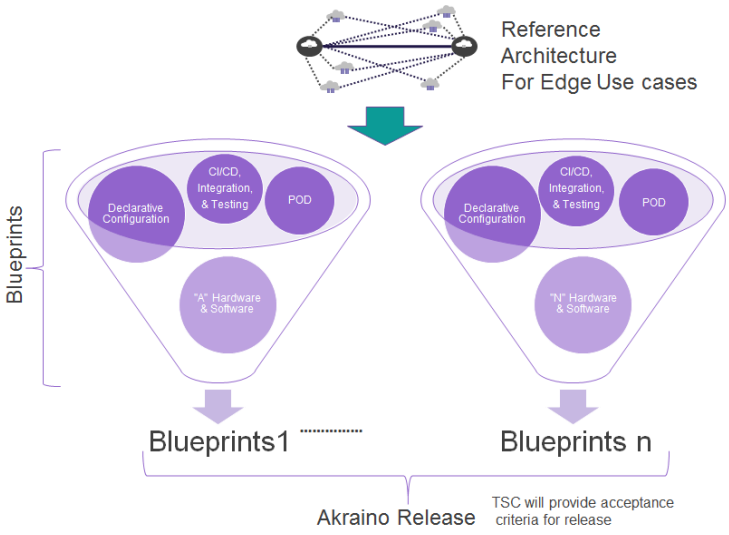

FIGURE 19. **Akraino Architecture [29].**

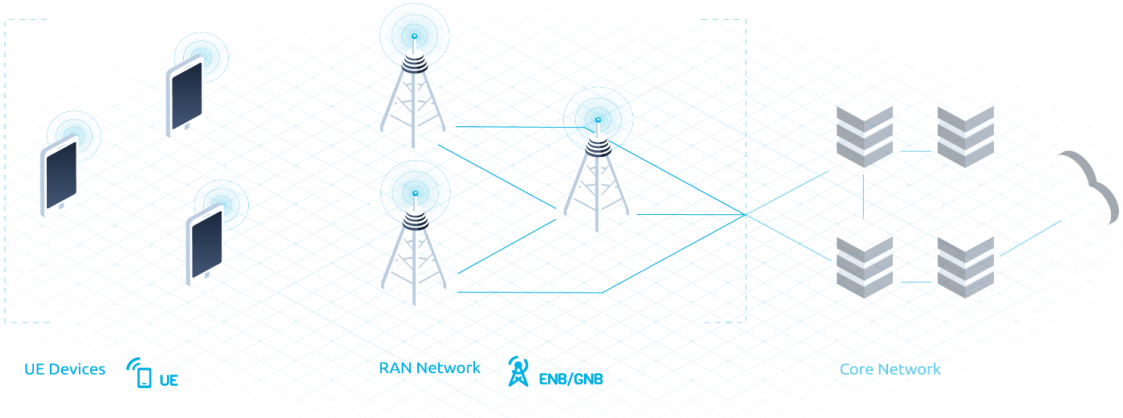

FIGURE 20. **srsRAN Architecture [30].**

[24]. Commodity servers interconnected by a fabric of White-box Switches, switching fabric in a Spine-Leaf topology for optimized East-to-West traffic and specialized access hardware for connecting subscribers (residential, mobile, and/or enterprise).

2) LL-MEC

The LL-MEC platform is made up of two main components: the LL-MEC platform and data-control APIs as shown in Fig. 16. The LL-MEC provides two main services: native IP-service endpoint and real-time radio network information to MEC applications on per user and service basis and can be connected to several underlying RANs and CN gateways. The data plane APIs acts as an abstraction layer between RAN and CN data plane and the LL-MEC platform. The OpenFlow and FlexRAN protocols facilitate the communication between the LL-MEC and underlying RAN and CN. With LL-MEC, coordinated RAN and CN network applications can be developed by leveraging both LL-MEC and FlexRAN SDKs allowing to monitor and control not only the traffic but also the state of network infrastructure [25].

3) LIGHTeDGe

Lightedge is a lightweight, ETSI-compliant MEC solution for 4G and 5G networks. Lightedge can provide Mobile Network Operators (MNOs) with a MEC platform that can immediately bring the advantages of edge computing to current 4G users as shown in Fig. 17 while enabling a seamless transition from the 4G towards a full 5G architecture [26]. Lightedge follows the bump in the wiring architecture proposed by ETSI, thereby placing the MEC host between

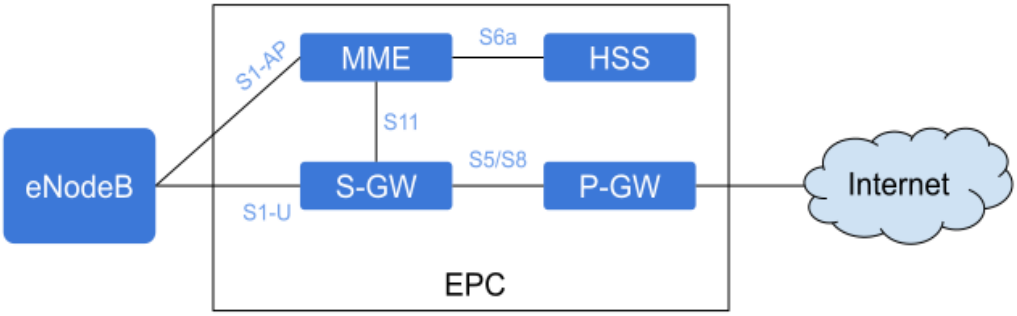

FIGURE 21. **srsEPC Architecture [31].**

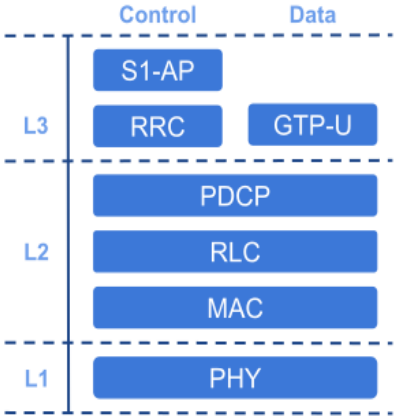

FIGURE 22. **srsENB Architecture [31].**

the RAN and the EPC of the 4G system to enable the interception of UE requests.

4) Aether

Aether is the first open-source 5G Connected Edge platform for enabling enterprise digital transformation. It provides mobile connectivity and edge cloud services for distributed enterprise networks as a cloud-managed offering as shown in Fig. 18. Aether is an open-source platform optimized for multi-cloud deployments, and simultaneous support for wireless connectivity over licensed, unlicensed, and lightly licensed (CBRS) spectrum [27]. The Aether network is operational and connects Aether Edges at project collaborator locations around the world. The network is used for Aether and Pronto development.

5) Akraino

Akraino is a set of open infrastructures and application blueprints for the Edge, spanning a broad variety of use cases, including 5G, AI, Edge IaaS/PaaS, IoT, for both provider and enterprise edge domains. These Blueprints have been created by the Akraino community and focus exclusively on the edge in all its different forms as shown in Fig. 19. What unites all these blueprints is that they have been tested by the community and are ready for adoption as-is or used as a starting point for customizing a new edge blueprint [29].

Akraino follows a holistic design focused on availability, capacity, security, and continuity such as:

1) Finite set of configurations – To reduce complexity, the design will follow a finite set of configurations.
2) May support Multiple workloads types such as VMs, Containers, microservices, etc.
3) Security – The design needs to validate the security of the blueprint.
4) Autonomous, turn-key solution for service enablement.
5) Platform, VNF, and application assessment and gating – assess whether the application is fit to run at the edge (e.g., latency sensitiveness, code quality).

These above-mentioned open-source software are non-conventional 5G software suites that can be used to build a 5G mobile networks. It depends on the user's choice, the resources available, and software platforms. **In our use case study, we will focus on srsRAN as the open-source software** [30] and possible software-defined radio (SDR) devices that can be deployed with srsRAN such as **Lime**

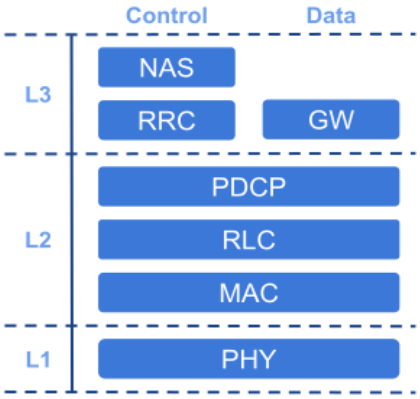

FIGURE 23. srsUE Architecture [31].

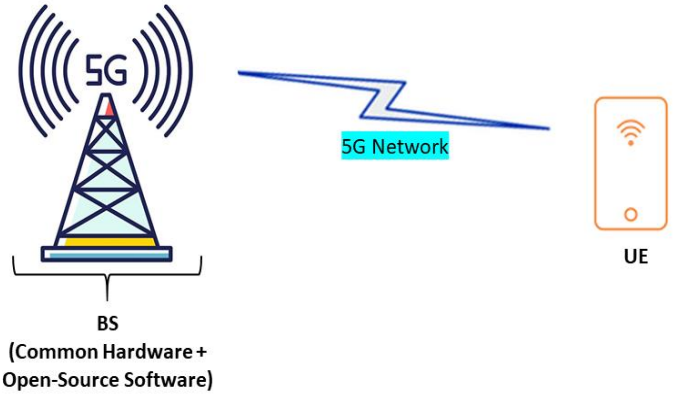

FIGURE 24. Basic 5G Network.

SDR mini, **BladeRF micro 2.0 xA4**, and **Ettus x310** that are listed in Tables 1,2, and 3 respectively [32]. Tables 1,2, and 3 give the description of each mentioned SDR, their advantages and limitations. It is up to the reader's choice which SDR to use.

TABLE 1.

Lime SDR mini	Description
1	Price: 200 US Dollars
2	Driver: SoapySDR
3	Frequency Range: 10 Mhz – 3.5 GHz
4	RF Bandwidth: 30.72 Mhz
5	Clock: 30.72 MHz onboard VCTCXO
6	Channels: 1x1

A brief description of what is srsRAN, use case scenarios of how to set up a 5G network using the **Zero MQ** with virtual radios, srsRAN on **Raspberry Pi 4**, and **5G Non-Standalone Access (NSA)** end to end system respectively will be given in the subsequent section. After this, we will mention the best possible SDR and PC hardware combinations that can be used with the srsRAN software suite in the following subsequent sections.

TABLE 2.

BladeRF micro 2.0 xA4	Description
1	Price: 500 US Dollars
2	Driver: SoapySDR
3	Frequency Range: 47 Mhz – 6 Ghz
4	RF Bandwidth: 56 Mhz
5	Clock: 38.4 MHz onboard VCTCXO
6	Channels: 2x2

TABLE 3.

Ettus x310	Description
1	Price: 7050 US Dollars
2	Driver: UHD
3	Frequency Range: DC - 6GHz (w/ Daughter Cards)
4	RF Bandwidth: 160 MHz (w/ Daughter Cards)
5	Clock: Configurable
6	Channels: 2x2

V. WHAT IS SRSRAN AND ITS FEATURES

srsRAN is an open-source 4G and 5G software radio suite as shown in Fig. 20. that provides UE and RAN solutions [30], [33]. It runs on off-the-shelf computer, RF hardware,

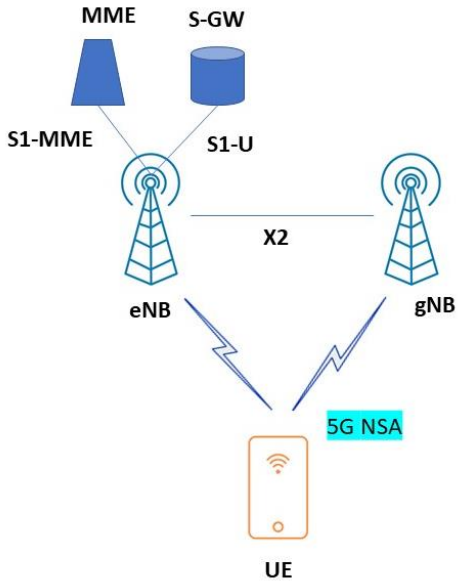

FIGURE 25. 5G NSA Network.

featuring both the UE and eNodeB/gNodeB applications [34].

A. SRSRAN FEATURES

The srsRAN includes features like srsEPC, srsENB and srsUE which are discussed below:

1) srsEPC

srsRAN uses srsEPC as a lightweight implementation of a complete LTE core network (EPC) as shown in Fig. 21. However, srsEPC runs as a single binary but still provides the key EPC components such as Home Subscriber Service (HSS), Mobility Management Entity (MME), Service Gateway (S-GW), and Packet Data Network Gateway (P-GW) [31].

2) srsENB

srsRAN uses srsENB in conjunction with srsEPC to implement functionalities of a complete 4G/5G BS [31] and includes the following features:

1) srsENB is aligned with LTE Release 10. It supports FDD configuration, has been tested with bandwidths 1.4, 3, 5, 10, 15, and 20 MHz and it also includes transmission modes such as 1 (single antenna), 2 (transmit diversity), 3 (CCD), and 4 (closed-loop spatial multiplexing) [31]. srsENB also offers command-line trace metrics, detailed input configuration files, and supports 5G NSA [31].
2) Frequency-based ZF and MMSE equalizers and highly optimized Turbo Decoder are also included in the srsENB. Detailed log system can be obtained with per-layer log levels and hex dumps and MAC layer Wireshark packet capture is also possible [31]. It also has a channel simulator for EPA, EVA, and ETU 3GPP channels, a ZeroMQ-based fake RF driver for I/Q over IPC/network, mobility support for Intra-ENB, and Inter-ENB (S1) and proportional-fair and round-robin MAC scheduler with FAPI-like C++ API [31].
3) srsENB also provides SR, periodic, aperiodic CQI feedback support, and standard S1AP and GTP-U interfaces to the core network as shown in Fig. 22. It can reach up to 150 Mbps during DL in 20 MHz MIMO TM3/TM4 with commercial UEs (195 Mbps with QAM256), 75 Mbps during DL in SISO configuration with commercial UEs, 50 Mbps during UL in 20 MHz with commercial UEs, and user-plane encryption [31].

3) srsUE

It is an application running on a Linux-based operating system, the excellent advantage of srsUE is that it is implemented entirely in software as a 4G LTE and 5G NR NSA UE modem [31]. It can connect to an LTE network to provide a standard network interface and includes the following features:

1) srsUE is LTE release 10 aligned with features up to release 15, is configured for TDD and FDD operations. It is tested with bandwidths like 1.4, 3, 5, 10, 15, and 20 MHz and includes transmission modes 1 (single antenna), 2 (transmit diversity), 3 (CCD), and 4 (closed-loop spatial multiplexing). It can manually configure DL/UL carrier frequencies and simulate 3GPP channels such as EPA, EVA, and ETU [31].
2) It offers TUN virtual network kernel interface integration for Linux OS, a detailed log system with per-layer log levels and hex dumps. It comes with MAC and NAS layer Wireshark packet captures, command-line trace metrics, and detailed input configuration files as shown in Fig. 23. srsUE also provides services like evolved multimedia broadcast and multicast service (eMBMS), frequency-based ZF and MMSE equalizers, and highly optimized Turbo Decoder for Intel SSE4.1/AVX2 (with +150 Mbps) [31].
3) For support purposes, srsUE includes 5G NSA support, soft USIM supporting XOR/Milenage authentication, hard USIM support via PC/SC, snow3G and AES integrity/ciphering support, QoS support, 150 Mbps during DL in 20 MHz MIMO TM3/TM4 or 2xCA configuration (195 Mbps with QAM256), 75 Mbps during DL in 20 MHz SISO configuration (98 Mbps with QAM256), 36 Mbps during DL in 10 MHz SISO configuration and supports Ettus USRP B2x0/X3x0 families, BladeRF, LimeSDR [31].

In the above paragraphs, we briefly discussed what is srsRAN and its features, now in the subsequent sections, we will talk about how to implement srsRAN with ZMQ application to simulate a virtual 5G Network, an end-to-end

5G NSA Network and after that, we will also list the steps of how practically implement srsRAN with an ultra-low-cost SDR device like limeSDR and low power Raspberry Pi 4 so that the users can practically build extremely low budget 5G Networks.

VI. SIMULATING AND SETTING UP A 5G NETWORK

The main goal of this setup is to help the readers learn how to build and deploy 4G/5G base stations. From Fig. 24, we can see a 5G network consisting of a 5G base station (which utilizes commonly available hardware and open-source software) and a user equipment (UE) device. As it depends on the reader's choice about what type of hardware device one can use, on the other hand, we illustrate the steps of how one can virtually simulate and practically build a 4G/5G Network using srsRAN with ZMQ and srsRAN with LimeSDR and Raspberry Pi 4.

1) Virtually building a full end-to-end LTE network on a single computer by using ZMQ Virtual Radios

In this particular approach, we can use virtual radios with a ZMQ networking library to transfer radio samples between applications and build an end-to-end network [35]. Why this approach is useful because we do not need to have any physical radios for performing development, testing, CI/CD, teaching, and demonstration. Before this, the user needs to make sure that they have installed ubuntu on their PC [36], on the virtual box which is a free and open-source hosted hypervisor for x86 virtualization [37]. After this, the user needs to install ZMQ and build srsRAN and for installing ZMQ development libraries on ubuntu the user can use:

1) ***sudo apt-get install libzmq3-dev***

Then, the user must install two important library files such as libzmq and czmq [35].

For libzmq:

1) ***git clone https://github.com/zeromq/libzmq.git***
2) ***cd libzmq***
3) ***./autogen.sh***
4) ***./configure***
5) ***make***
6) ***sudo make install***
7) ***sudo ldconfig***

For czmq:

1) ***git clone https://github.com/zeromq/czmq.git***
2) ***cd czmq***
3) ***./autogen.sh***
4) ***./configure***
5) ***make***
6) ***sudo make install***
7) ***sudo ldconfig***

Now, to build and compile srsRAN so that it recognizes the addition of ZMQ:

1) ***git clone https://github.com/srsRAN/srsRAN.git***
2) ***cd srsRAN***
3) ***mkdir build***
4) ***cd build***
5) ***cmake ../***
6) ***make***

After this, the cmake console output should read the following lines:

1) ***FINDING ZEROMQ.***
2) ***Checking for module 'ZeroMQ'***
3) ***No package 'ZeroMQ' found***
4) ***Found libZEROMQ: /usr/local/include,***
5) ***/usr/local/lib/libzmq.so***

The ZMQ application has been installed.

Now, to start the LTE network on a single PC, we need to make sure that UE and EPC both are in different namespaces because they will share the same network configuration, UE receives the IP address from the EPC subnet and the Linux will bypass the TUN interfaces when routing traffic between both ends. To communicate over the TCP/IP stack, we require TUN interfaces for the UE and EPC. Each srsRAN application will be launched in a separate terminal as well as ping and iperf [35].

Basically, there are six steps in order to launch the network and at the end terminate it, following the **step 1**, creating network namespace ue1 for UE as:

1) ***sudo ip netns add ue1***

In **step 2**, running the EPC, will create a TUN device in the default network namespace so it will need root permissions as:

1) ***sudo ./srsepc/src/srsepc***

In **step 3**, running the eNodeB, with the default configurations and pass all the parameters needed to be tweaked for ZMQ as command line programs:

1) ***./srsenb/src/srsenb - -rf.device_name=zmq - -rf.device_args="fail_on_disconnect=true, tx_port=tcp://*:2000,rx_port=tcp://localhost:2001, id=enb,base_srate=23.04e6"***

In **step 4**, running the UE with root permission to create the TUN device as:

1) ***sudo ./srsue/src/srsue - -rf.device_name=zmq - -rf.device_args="tx_port=tcp://*:2001, rx_port=tcp://localhost:2000,id=ue, base_srate=23.04e6" - -gw.netns=ue1***

This command will run the UE and attach it to the core network, then the UE will be assigned a new IP address 172.16.0.2.

In **step 5**, generating traffic in the downlink direction from EPC to UE, by just entering ping in command line as:

1) ***ping 172.16.0.2***

and in order to generate traffic in the uplink direction from UE to EPC, running the ping command in the UE network name space as:

1) ***sudo ip netns exec ue1 ping 172.16.0.1***

After **step 5** and finishing the setup, the user also needs to delete the netns as: s

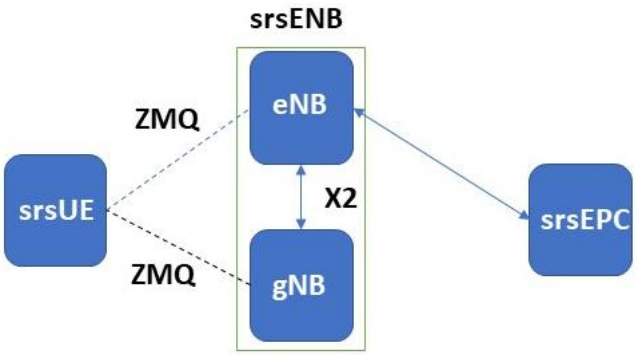

FIGURE 26. **Overview of NSA Network Architecture.**

1) ***sudo ip netns delete ue1***

By following the above directions and the five steps, the users can run an end-to-end LTE network [35] which can be highly desirable for an individual with a single PC. To end the simulation, first, the UE needs to be terminated then the eNB, they can only run one time, to run them again the UE needs to detach and after that, the eNB can be restarted. For now, only one eNB and UE are supported [35].

2) Virtually building a 5G NSA End-to-End Network using ZMQ

The 21.10 release of srsRAN brings 5G NSA features to srsENB and can be enabled via the srsENB configuration files. Therefore, by using the ZMQ virtual radios in place of actual RF hardware we can create a 5G network as shown in the following steps:

From Fig. 27. we can see the 5G Non-Standalone mode, which uses the pre-existing 4G infrastructure to build upon, two carriers are sent to UE, the primary carrier being the 4G and the secondary carrier being the 5G, the UE first connects with the 4G carrier and then with the 5G carrier. The 4G carrier anchor is used for control plane signaling while the 5G carrier is used for high-speed data plane traffic [39].

To run this setup, we need three things, srsUE must run in a separate network namespace, srsENB should be configured so that both the LTE eNB, and the NSA gNB cells are created at run time and srsEPC should include UE in the list of subscribers [39]. To enable ZMQ and 5G NSA carrier changes must be made in the configuration files of srsUE and srsENB, no changes are made in the srsEPC and also the modified config files can be downloaded from here [39].

For srsUE, changes must be made in ue.conf file to enable ZMQ and 5G NR features. To enable ZMQ, we need to change the device names in [rf] section of UE as:

1) ***device_name = zmq***
2) ***device_args = tx_port0=tcp://*:2001, rx_port0=tcp://localhost:2000,tx_port1=tcp://*:2101, rx_port1=tcp://localhost:2100,id=ue, base_srate=23.04e6***

In the above command, two TX and two RX channels are added to support the primary and secondary carrier as shown in Fig. 26. So, to complete the ZMQ setup, the user needs to set the network namespace netns under the [gw] settings as:

1) ***[gw]***
2) ***netns = ue1***

Now to setup the NR RAT, which provides the 5G NR capabilities of UE, this has to enabled in the config under [rat.nr] as:

1) ***[rat.nr]***
2) ***bands = 3, 78***
3) ***nof_carriers = 1***

In the above command, the enable bands are 3 and 78 which reflect the FDD and TDD respectively, and the user can test each duplex mode simply by configuring the network. The number of carriers must be set to 1, if not then the UE will not be able to connect to gNB and will only have an LTE connection [39]. The NSA mode is a part of 3GPP release 15, so it must be included in the config entry under the [rrc] field, by default the release included is 8.

1) ***[rrc]***
2) ***release = 15***

Until now we have only configured UE, now moving on the srsENB, to enable 5G NSA, the user needs to make changes in both enb.conf and rr.conf as shown below:

For configuring the eNB, the user needs to make the required changes to enable ZMQ in the [rf] section as:

1) ***device_name = zmq***
2) ***device_args = fail_on_disconnect=true, tx_port0=tcp://*:2000,rx_port0=tcp://localhost:2001, tx_port1=tcp://*:2100,rx_port1=tcp://localhost:2101, id=enb,base_srate=23.04e6***

Just like UE, here we have two TX and two RX channels that are mapped to the relevant ports configured on the UE

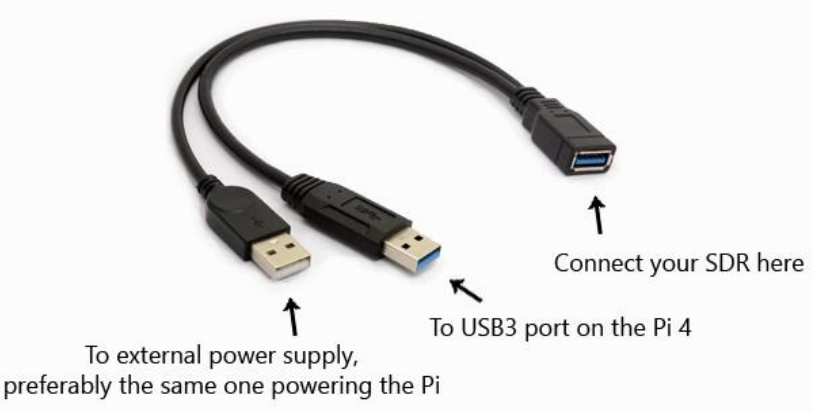

FIGURE 27. Y Cable [38].

[39]. For RRC config, the main change is to add the NR cell to the cell list in the rr.conf file to the end of the file as:

1) ***nr_cell_list =***
   (
   {
   rf_port = 1;
   cell_id = 0x02;
   tac = 0x0007;
   pci = 500;
   root_seq_idx = 204;

   // TDD:
   //dl_arfcn = 634240;
   //band = 78;

   // FDD:
   dl_arfcn = 368500;
   band = 3;
   }
   );

As shown in the above command, the user can see the FDD and TDD configs, however, for the RRC config, only the FDD is used as shown above. For utilizing TDD, the user needs to the opposite change and restart the srsENB [39]. Till now, the necessary changes have been made to operate the 5G NSA mode. Now the user can start the network as shown below in three steps:

Step 1, run the EPC,

1) ***sudo srsepc***

This command will give the output as:

1) ***HSS Initialized.***
2) ***MME S11 Initialized.***
3) ***MME GTP-C Initialized.***
4) ***MME Initialized. MCC: 0xf001, MNC: 0xff01***
5) ***SPGW GTP-U Initialized.***
6) ***SPGW S11 Initialized.***
7) ***SP-GW Initialized.***

Step 2, run the eNB/ gNB,

1) ***sudo srsenb***

The output obtained will look like this:

1) ***Opening 2 channels in RF device=zmq with args=fail_on_disconnect=true, tx_port0=tcp://*:2000,rx_port0=tcp://localhost:2001, tx_port1=tcp://*:2100, rx_port1=tcp://localhost:2101,id=enb,base_srate=23.04e6***
2) ***CHx base_srate=23.04e6***
3) ***CHx id=enb***
4) ***Current sample rate is 1.92 MHz with a base rate of 23.04 MHz (x12 decimation)***
5) ***CH0 rx_port=tcp://localhost:2001***
6) ***CH0 tx_port=tcp://*:2000***
7) ***CH0 fail_on_disconnect=true***
8) ***CH1 rx_port=tcp://localhost:2101***
9) ***CH1 tx_port=tcp://*:2100***
10) ***==== eNodeB started ===***
11) ***Type <t> to view trace***
12) ***Current sample rate is 11.52 MHz with a base rate of 23.04 MHz (x2 decimation)***
13) ***Current sample rate is 11.52 MHz with a base rate of 23.04 MHz (x2 decimation)***
14) ***Setting frequency: DL=2680.0 Mhz, UL=2560.0 MHz for cc_idx=0 nof_prb=50***
15) ***Setting frequency: DL=1842.5 Mhz, UL=1747.5 MHz for cc_idx=1 nof_prb=52***

From the output, the IDs can be seen as 0 and 1, where 0 represents LTE cell and 1 represents NR Cell [39]. Next step is to run the UE as:

1) ***sudo srsue***

When the UE runs successfully then attaches with the eNB/gNB, the user will see the following output:

1) ***Opening 2 channels in RF device=zmq with args=tx_port0=tcp://*:2001, rx_port0=tcp://localhost:2000,***

tx_port1=tcp://:2101,rx_port1=tcp://localhost:2100,id=ue,base_srate=23.04e6*
2) *CHx base_srate=23.04e6*
3) *CHx id=ue*
4) *Current sample rate is 1.92 MHz with a base rate of 23.04 MHz (x12 decimation)*
5) *CH0 rx_port=tcp://localhost:2000*
6) *CH0 tx_port=tcp://*:2001*
7) *CH1 rx_port=tcp://localhost:2100*
8) *CH1 tx_port=tcp://*:2101*
9) *Waiting PHY to initialize ... done!*
10) *Attaching UE...*
11) *Current sample rate is 1.92 MHz with a base rate of 23.04 MHz (x12 decimation)*
12) *Current sample rate is 1.92 MHz with a base rate of 23.04 MHz (x12 decimation)*
13) *Found Cell: Mode=FDD, PCI=1, PRB=50, Ports=1, CP=Normal, CFO=-0.2 KHz*
14) *Current sample rate is 11.52 MHz with a base rate of 23.04 MHz (x2 decimation)*
15) *Current sample rate is 11.52 MHz with a base rate of 23.04 MHz (x2 decimation)*
16) *Found PLMN: Id=00101, TAC=7*
17) *Random Access Transmission: seq=33, tti=981, ra-rnti=0x2*
18) *RRC Connected*
19) *Random Access Complete. c-rnti=0x46, ta=0*
20) *Network attach successful. IP: 172.16.0.3*
21) *Software Radio Systems RAN (srsRAN) 13/10/2021 15:29:9 TZ:0*
22) *RRC NR reconfiguration successful.*
23) *Random Access Transmission: prach_occasion=0, preamble_index=0, ra-rnti=0xf, tti=1611*
24) *Random Access Complete. c-rnti=0x4601, ta=0*

In the output, **RRC NR reconfiguration successful** points out the successful connection of UE with the NR cell, this will be used for data link while the LTE cell will be used for control messaging [39]. From here, we move towards the testing of the NSA connection, to generate traffic from UE to gNB, iperf3 client will be used on the UE side, UDP traffic will be generated at 10 Mbps and 60 seconds [39], here it is important to note to start the server first and then the client.

To listen to traffic coming from UE, the user must start the iPerf server on network side:

1) *iperf3 -s -i 1*

With the network and iPerf up and running, it's time to run the client on UE side from the command:

1) *sudo ip netns exec ue1 iperf3 -c 172.16.0.1 -b 10M -i 1 -t 60*

Traffic will now be sent from UE to eNB, it will be shown in both server and iperf consoles.

For client iPerf output:

1) *Connecting to host 172.16.0.1, port 5201*
2) *[5] local 172.16.0.2 port 52484 connected to 172.16.0.1 port 5201*
3) **[ID] Interval Transfer Bitrate Retr Cwnd**
4) *[5] 0.00-1.00 sec 954 KBytes 7.81 Mbits/sec 0 79.2 KBytes*
5) *[5] 1.00-2.00 sec 1.12 MBytes 9.44 Mbits/sec 0 126 KBytes*
6) *[5] 2.00-3.00 sec 1.00 MBytes 8.39 Mbits/sec 12 49.5 KBytes*
7) *[5] 3.00-4.00 sec 640 KBytes 5.24 Mbits/sec 2 42.4 KBytes*
8) *[5] 4.00-5.00 sec 512 KBytes 4.19 Mbits/sec 2 39.6 KBytes*
9) *[5] 5.00-6.00 sec 512 KBytes 4.19 Mbits/sec 2 33.9 KBytes*

For server iPerf output:

1) *- -*
2) *Server listening on 5201*
3) *- -*
4) *Accepted connection from 172.16.0.2, port 52482*
5) *[5] local 172.16.0.1 port 5201 connected to 172.16.0.2 port 52484*
6) *[ID] Interval Transfer Bitrate*
7) *[5] 0.00-1.00 sec 634 KBytes 5.19 Mbits/sec*
8) *[5] 1.00-2.00 sec 950 KBytes 7.78 Mbits/sec*
9) *[5] 2.00-3.00 sec 977 KBytes 8.00 Mbits/sec*
10) *[5] 3.00-4.00 sec 533 KBytes 4.36 Mbits/sec*
11) *[5] 4.00-5.00 sec 553 KBytes 4.53 Mbits/sec*
12) *[5] 5.00-6.00 sec 537 KBytes 4.40 Mbits/sec*

Traffic can be traced from UE and eNB/gNB consoles by typing t in each console and the result obtained will be as shown below: For UE console,

1) *- - - - - -Signal- - - - - - - - - -|*
   - - - - - - -DL- - - - - - - - - - - -|
   - - - - - - -UL- - - - - - - - - - - -
2) *rat pci rsrp pl cfo |*
   mcs snr iter brate bler ta_us |
   mcs buff brate bler
3) *lte 1 -11 11 -1.4u |*
   0 142 0.0 0.0 0% 0.0 |
   0 0.0 0.0 0%
4) *nr 500 1 0 23u |*
   27 70 1.0 8.5M 0% 0.0 |
   28 36k 8.3M 0%
5) *lte 1 -11 11 -1.4u |*
   0 142 0.0 0.0 0% 0.0 |
   0 0.0 0.0 0%
6) *nr 500 1 0 23u |*
   27 70 1.0 9.2M 0% 0.0 |
   28 24k 8.1M 0%

7) ***lte 1 -11 11 -1.4u | 0 142 0.0 0.0 0% 0.0 | 0 0.0 0.0 0%***
8) ***nr 500 2 0 23u | 27 69 1.0 4.6M 0% 0.0 | 28 19k 4.2M 0%***
9) ***lte 1 -11 11 -1.3u | 0 142 0.0 0.0 0% 0.0 | 0 0.0 0.0 0%***
10) ***nr 500 2 0 23u | 27 69 1.0 5.0M 0% 0.0 | 28 26k 4.8M 0%***
11) ***lte 1 -11 11 -1.4u | 0 142 0.0 0.0 0% 0.0 | 0 0.0 0.0 0%***
12) ***nr 500 2 0 23u | | 27 69 1.0 4.7M 0% 0.0 | 28 28k 4.7M 0%***

For eNB/ gNB console,

1) ***- - - - - - - - - - - - - -DL- - - - - - - - - - - - - -| - - - - - - - - - - - - - -UL- - - - - - - - - - - - - -***
2) ***rat rnti cqi ri mcs brate ok nok (%) | pusch pucch phr mcs brate ok nok (%) bsr***
3) ***lte 46 15 0 0 0 0 0 0% | n/a n/a 0 0 0 0 0 0% 0.0***
4) ***nr 4601 n/a 0 27 6.9M 124 0 0% | n/a n/a 0 0 6.1M 95 0 0% 0.0***
5) ***lte 46 15 0 0 0 0 0 0% | n/a n/a 0 0 0 0 0 0% 0.0***
6) ***nr 4601 n/a 0 27 4.4M 92 0 0% | n/a n/a 0 0 4.2M 76 0 0% 0.0***
7) ***lte 46 15 0 0 0 0 0 0% | n/a n/a 0 0 0 0 0 0% 0.0***
8) ***nr 4601 n/a 0 27 5.2M 113 0 0% | n/a n/a 0 0 5.0M 94 0 0% 0.0***
9) ***lte 46 15 0 0 0 0 0 0% | n/a n/a 0 0 0 0 0 0% 0.0***
10) ***nr 4601 n/a 0 27 5.4M 118 0 0% | n/a n/a 0 0 5.3M 99 0 0% 0.0***
11) ***lte 46 15 0 0 0 0 0 0% | n/a n/a 0 0 0 0 0 0% 0.0***
12) ***nr 4601 n/a 0 27 7.6M 156 0 0% | n/a n/a 0 0 7.2M 129 0 0% 0.0***

In both outputs obtained from UE and eNB/gNB, the rat column represents the metrics associated with the NSA 5G link (nr), or with the LTE link (lte). In the above section (ii), we talked about the steps related to simulating 5G NSA network using srsRAN with ZMQ application. In the (iii) section, we will enumerate the steps the user can take to build a practical 4G LTE network.

3) Building a 4G LTE Network using srsRAN on Raspberry Pi 4 as a PC and limeSDR USRP

A user can easily build a 4G LTE network by using ultra-low-cost computing hardware like the low power Raspberry Pi 4, since srsRAN consists of a core network and an eNB, the eNB can be run on the hardware device such as the **Raspberry Pi 4B /4GB rev 1.2** running with Ubuntu Server 20.04 LTS aarch64 image [38]. The Pi4 hardware revision can be checked from [40]. The whole LTE setup can be run with a USRP B210, a LimeSDR-USB, and a LimeSDR-Mini, but keeping in mind that when using USRP B210 the user can create a 2x2 MIMO cell with srsenb and run the srsepc core network on the Pi too, however, when using either of the LimeSDRs, the user can only create a 1x1 SISO cell with srsenb and the core network must be run on a separate device [38]. It should be noted that when using the SDRs due to their power requirements the user must use an external power source this can be achieved through the Y cable as shown in Fig. 27. The LTE setup consists of two steps, the first is a software set up and the second step is the hardware setup. In the first step, the user needs to set up the required software as shown below and then proceed towards setting up the hardware and running the LTE network.

Step 1: Configuring the software

The user needs to install the SDR drivers and build srsRAN. For USRP the UHD drivers and for LimeSDRs the SoapySDR/LimeSuite drivers can be installed as shown below:

For USRP:

1) ***sudo apt update***
2) ***sudo apt upgrade***
3) ***sudo apt install cmake***

and followed by:

1) ***sudo apt install libuhd-dev libuhd3.15.0 uhd-host***
2) ***sudo /usr/lib/uhd/utils/uhd_images_downloader.py***
3) ***## Then test the connection by typing:***
4) ***sudo uhd_usrp_probe***

For SoapySDR:

1) ***git clone https://github.com/pothosware/SoapySDR.git***
2) ***cd SoapySDR***
3) ***git checkout tags/soapy-sdr-0.7.2***
4) ***mkdir build cd build***
5) ***cmake ..***
6) ***make -j4***
7) ***sudo make install***
8) ***sudo ldconfig***

For LimeSuite:

1) ***sudo apt install libusb-1.0-0-dev***
2) ***git clone https://github.com/myriadrf/LimeSuite.git***
3) ***cd LimeSuite***
4) ***git checkout tags/v20.01.0***
5) ***mkdir builddir cd builddir***
6) ***cmake ../***
7) ***make -j4***
8) ***sudo make install***

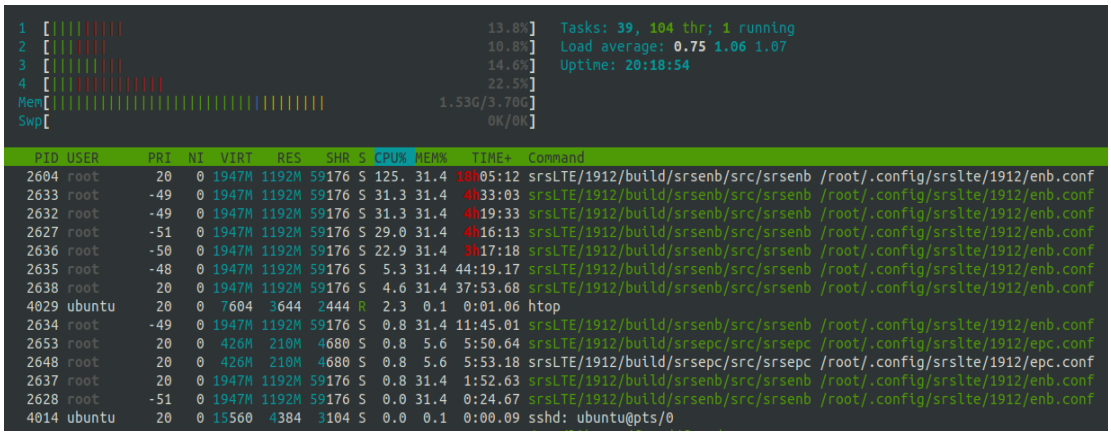

FIGURE 28. **htop (LTE connection) [38].**

9) ***sudo ldconfig***
10) ***cd ..***
11) ***cd udev-rules***
12) ***sudo ./install.sh***

13) ***## Then test the connection by typing:***
14) ***LimeUtil –find***
15) ***LimeUtil –update***
16) ***SoapySDRUtil –find***

Now to compile the srsRAN as,

1) ***sudo apt install libfftw3-dev libmbedtls-dev libboost-program-options-dev libconfig++-dev libsctp-dev***
2) ***git clone https://github.com/srsRAN/srsRAN.git***
3) ***cd srsran***
4) ***git checkout tags/release_19_12***
5) ***mkdir build cd build***
6) ***cmake ../***
7) ***make -j4***
8) ***sudo make install***
9) ***sudo ldconfig***

10) ***## copy configs to /root***
11) ***sudo ./srsran_install_configs.sh user***

Until this step, the drivers have been installed and now the user must modify the **Pi CPU scaling_governor** to ensure it is running in performance mode as:

1) ***sudo systemctl disable ondemand***
2) ***sudo apt install linux-tools-raspi***

3) ***sudo nano /etc/default/cpufrequtils***
4) **** insert:***
5) **** GOVERNOR="performance"***

6) ***## reboot***
7) ***sudo cpupower frequency-info***
8) **** should show that the CPU is running in performance mode, at maximum clock speed***

Step 2: Configuring the hardware

To start the Pi eNB, some configurations must be made regarding the USRP B210 and LimeSDRs and the choice is up to the user, which hardware to use the USRP or the SDR. For this LTE setup, the Pi4 eNodeB was tested with a 3MHz wide cell in LTE B3 (1800MHz band), DL=1878.40 UL=1783.40 [38].

For USRP B210 changes to be made in default enb.conf file:

1) ***sudo nano /root/.config/srsran/enb.conf***
2) ***[enb]***

3) ***mcc = <yourMCC>***
4) ***mnc = <yourMNC>***
5) ***mme_addr = 127.0.1.100 ## or IP for external MME, eg. 192.168.1.10***
6) ***gtp_bind_addr = 127.0.1.1 ## or local interface IP for external S1-U, eg. 192.168.1.3***
7) ***s1c_bind_addr = 127.0.1.1 or local interface IP for external S1-MME, eg. 192.168.1.3***
8) ***n_prb = 15***
9) ***tm = 2***
10) ***nof_ports = 2***

11) ***[rf]***
12) ***dl_earfcn = 1934***
13) ***tx_gain = 80 ## this power seems to work best***
14) ***rx_gain = 40***
15) ***device_name = UHD***
16) ***device_args = auto ## does not work with anything other than 'auto'***

For LimeSDR-USB or LimeSDR-Mini changes to be made in default enb.conf file:

1) ***sudo nano /root/.config/srsran/enb.conf***

2) ***[enb]***
3) ***mcc = <yourMCC>***
4) ***mnc = <yourMNC>***
5) ***mme_addr = <ipaddr>***
   ## IP for external MME, eg. 192.168.1.10
6) ***gtp_bind_addr = <ipaddr> local interface***
   IP for external S1-U, eg. 192.168.1.3
7) ***s1c_bind_addr = <ipaddr> local interface***
   IP for external S1-MME, eg. 192.168.1.3
8) ***n_prb = 15***
9) ***tm = 1***
10) ***nof_ports = 1***

11) ***[rf]***
12) ***dl_earfcn = 1934***
13) ***tx_gain = 60 ## this power seems to work best***
14) ***rx_gain = 40***
15) ***device_name = soapy***
16) ***device_args = auto***
    ## does not work with anything other than 'auto'

And the changes to be made in the default configs of srsRAN core network:

1) ***sudo nano /root/.config/srsran/epc.conf***

2) ***[[mme]]***
3) ***mcc = <yourMCC>***
4) ***mnc = <yourMNC>***
5) ***mme_bind_addr = 127.0.1.100 ## or local interface***
   IP for external S1-MME, eg. 192.168.1.10
1) ***sudo nano /root/.config/srsran/user_db.csv***

2) **** add details of your SIM cards***

To keep in mind, when running the srsRAN core network (srsepc) on an external device like another Pi, the user must open incoming firewall ports to allow the S1-MME and S1-U connections from srsenb.

S1-MME = sctp, port 36412 || S1-U = udp, port 2152

If using iptables,

1) ***sudo iptables -A INPUT -p sctp -m sctp - -dport 36412 -j ACCEPT***

2) ***sudo iptables -A INPUT -p udp -m udp - -dport 2152 -j ACCEPT***

Finally, when running the Pi4 eNodeB, the user must launch the software in separate ssh windows or using a screen and for SDR an external power source must be used. When running the srsenb for the first time the user has to wait a few minutes for it to finish [38].

Now, Launch Pi4 eNodeB as:

1) ***sudo srsenb /root/.config/srsran/enb.conf***

For LimeSDRs, between runs when using the LimeSDR-USB, the user sometimes needs to physically unplug and reconnect the SDR to power cycle it. Now, the user can launch the core network (on a separate device, or the Pi4 eNodeB when using USRP B210) as:

1) ***sudo srsepc /root/.config/srsran/epc.conf***
2) ***sudo /usr/local/bin/srsepc_if_masq.sh eth0***

From, Fig. 28, we can see the resource utilization when running the software on the Pi 4B /4GB RAM with x2 UEs attached to the USRP B210 cell. It is important to note here that Fig. 28 shows the results of the srsRAN software running for more than 18 hours without any problems, only half of the RAM is used, and the CPU cores are sitting at around 25% [38].

In these three ways, the users can build and run inexpensive 4G/5G networks virtually and practically through using srsRAN and the required hardware. There are many PC and SDR combinations possible to be able to run the srsRAN, but for those users, who want to best explore the functionalities of srsRAN, we list three hardware packages that can help build the required setup to run srsRAN and provide information to the users of what they can buy. These setups are grouped according to price ranging from 400$ to 16000$ [32].

When choosing for the computing hardware, the users can consider the following criteria given in Table 4:

TABLE 4.

S. No.	Compute Criteria
1	Overall cost of the machine.
2	Number of cores effect the overall performance.
3	CPUs running at lower frequencies may struggle under heavy computational load.
4	Greater cache memory size greater the speed of certain computations.
5	Number of threads increase the processor's execution speed.

This criterion doesn't include all the features, because depending upon the use-case, the choice of features may vary and include features like processor cinebench score, cooling ability, and portability. For SDR criteria, just the like commute criteria, there are many features to look for but depending upon the use-case, the features may vary. In Table 5, the SDR criteria is given:

TABLE 5.

S. No.	SDR Criteria
1	Cost per unit of the SDR.
2	The drivers used by SDR (Soapy, UHD, etc).
3	The frequency range(s) the SDR operates in.
4	Maximum possible bandwidth available.
5	Clock rate of the SDR.
6	How many channels the SDR support (SISO, MIMO, etc).
6	The specifications of the onboard FPGA.

Based on the Tables 4 and 5, the users can decide on the compute and SDR criteria then compare which hardware options are more favorable to them. By following the hardware combinations given below they can directly use them with srsRAN to build a 4G/5G network, again it depends upon the use case and the resources available to the user. Each combination contains an SDR and computer

hardware section, to run a full end-to-end system at least two SDRs and two compute platforms are required [32].

TABLE 6. Combination 1

SDR	Computer Hardware
Lime SDR mini	Raspberry Pi 4
Price: $200	Price: $72
Driver: SoapySDR	# Cores: 4
Frequency Range: 10 Mhz – 3.5 GHz	Frequency: 1.5 Ghz
RF Bandwidth: 30.72 Mhz	Cache Size: 1 MiB
Clock: 30.72 MHz onboard VCTCXO	# Threads: 4
# Channels: 1x1	.
FPGA: Intel Altera MAX 10	

Combination 1 shown in Table 6, will enable users to set up a cheap end-to-end wireless network and just under 650$. To run the full end-to-end system, the user will need two LimeSDR minis and three Raspberry Pi4 units, each RF frontend for eNB and UE and a Pi4 for EPC, eNB, and UE. This combination is highly ideal for demos and testing of networks and applications due to small size and portability [32]. Moving to the second combination shown in Table 7:

TABLE 7. Combination 2

SDR	Computer Hardware
BladeRF micro 2.0 xA4	HP Omen 15 Intel i5-10300H
Price: $550	Price: $1,049.99
Driver: SoapySDR	# Cores: 4
Frequency Range: 47 Mhz – 6 GHz	Frequency: 2.4 – 4.5 Ghz
RF Bandwidth: 56 Mhz	Cache Size: 8 MiB
Clock: 38.4 MHz onboard VCTCXO	# Threads: 8
# Channels: 2x2	.
FPGA: Altera Cyclone V (49 kLE)	

The BladeRF micro 2.0 xA4 offers users a 2X2 MIMO configuration, higher bandwidth, a larger frequency range, and a larger FPGA. On the other hand, HP Omen 15 is a gaming notebook, built for high performance and high CPU load for a certain time length. The intel i5 10300H is the main focus here, having scored highly in the cinebench r20 benchmarking test, therefore this combination is an upgrade over the previous combination 1 in terms of price and performance [32].

The combination shown in Table 8 by far offers the most potential in terms of RF frontends and PC performance. The Ettus x310 offers users the largest frequency range,

TABLE 8. Combination 3

SDR	Computer Hardware
Ettus x310	Dell Precision 3340 Workstation Intel i7-10700
Price: $7,013.00	Price: $1349.00
Driver: UHD	# Cores: 8
Frequency Range: DC - 6GHz (w/ Daughter Cards)	Frequency: 2.9 - 4.8 GHz
RF Bandwidth: 160 MHz (w/ Daughter Cards)	Cache Size: 16 MiB
Clock: Configurable	# Threads: 16
# Channels: 2x2	.
FPGA: KINTEX7-410T	

from DC to 6 GHz with the use of the appropriate daughter cards, a potential bandwidth of 160 MHz (requires the correct daughter cards), a multi-cell configuration, and a powerful Kintex7 FPGA. The 3340 workstation offers an intel i7-10700 which is capable of high-intensity computations without a significant drop-off in performance over sustained periods [32]. The workstation offers 10 Gbps ethernet connection, which allows users full utilization of the 10 Gbps connection available on the x310. The full end-to-end system would cost around $16724.

VII. CONCLUSION AND FUTURE WORKS

In this manuscript, we presented a comprehensive tutorial on how to build a 5G network using open source software and off-the-shelf hardware. The main purpose of this paper is to inform and guide individuals on how to build a 5G network, in a very simple, easy, and straightforward way. We have compiled this tutorial in such a way, that it clearly explains the evolution of the wireless systems from closed RAN to open RAN and why the closed RAN approach is unfruitful. We further discussed the legacy RANs and the companies related to them. Then we described in detail the numerous approaches available to build a 5G network due to the open RAN concept that includes RAN, CORE, and EDGE frameworks. These frameworks also highlight the main ideas and technologies implemented in 5G networks. Then we demonstrated each technology, software, and application used in building and simulating the 5G network. There are three procedures that we illustrated in the paper, that includes virtually building an end-to-end LTE network on a single computer by using ZMQ Virtual Radios, a 5G NSA End-to-End Network using ZMQ, and a 4G LTE Network using srsRAN on RaspberryPi 4 as a PC and limeSDR USRP. We have clearly explained these three methods in the paper with the required assumptions. In the end, we concluded the paper with recommendations of appropriate hardware and SDR combinations that will result in the best experience when building a 5G network according to relevant usage and resources.

REFERENCES

[1] A. G. Zaballos, E. I. Rodriguez, K. W. Kim, and S. Park, "5G, the driver for the next-generation digital society in latin america and the caribbean." IDB, Mar 2020.
[2] Qualcomm, "Everything you need to know about 5G." Qualcomm, https://www.qualcomm.com/5g/what-is-5g.
[3] V. Singh, Sr. Research Analyst Market Research and N. Tanwer, Research Analyst Market Research, "5G companies: Which players are leading the market?" greyb, 2020, https://www.greyb.com/5g-companies/.
[4] "AWS Private 5G, easily deploy, manage, and scale a private cellular network," Amazon Web Services, 2021, https://aws.amazon.com/private5g/.
[5] E. Udin, "A 5G base station is quite expensive, why destroy it?" Gizchina Media, 2020, https://www.gizchina.com/2020/04/05/a-5g-base-station-is-quite-expensive-why-destroy-it/.
[6] G. E. Gonçalves, "Flying to the clouds: The evolution of the 5G radio access networks," Palgrave Studies in Digital Business Enabling Technologies. Palgrave Macmillan, Cham, 2020.
[7] "O-RAN," O-RAN Alliance, 2019, https://www.o-ran.org/.
[8] "What Is vRAN (Virtual Radio Access Network)?" SDxCentral, 2018, https://www.sdxcentral.com/5g/ran/definitions/vran/.
[9] D. Jones and C. Bernstein, "Radio access network (RAN)," TechTarget, 2021, https://www.techtarget.com/searchnetworking/definition/radio-access-network-RAN.
[10] ONF, "SD RAN," ONF Mobile Projects, https://opennetworking.org/sd-ran/.
[11] S. Partners, "10 RAN vendors: challengers, newcomers and heavyweights," STL Advisory Limited, 2017, https://stlpartners.com/telco_cloud/10-ran-vendors/.
[12] M. Lore, "Open or closed, RAN vendors face a 2020s squeeze," Light Reading, 2021, https://www.lightreading.com/open-ran/open-or-closed-ran-vendors-face-2020s-squeeze/a/d-id/766804.
[13] "NVIDIA aerial," NVIDIA Corporation, https://developer.nvidia.com/aerial-sdk.
[14] "An O-RAN compliant near-RT RAN intelligent controller for het RANs," 5G-EmPOWER, http://5g-empower.io/.
[15] "A 5G americas white paper, transition toward open interoperable networks," 5G Americas, 2020, https://www.5gamericas.org/transition-toward-open-interoperable-networks/.
[16] "FLEXRAN," mosaic5g, https://mosaic5g.io/flexran/.
[17] NI, "What is the difference between NI and ettus research USRP devices?"," NI, 2021, https://www.ni.com/en-tr/innovations/white-papers/19/what-is-the-difference-between-ni-and-ettus-usrps.html.
[18] "OpenAirInterface 5G Radio Access Network Project," OpenAirInterface, 2021, https://openairinterface.org/oai-5g-ran-project/.
[19] "Open Virtualized RAN (Open vRAN), Mavenir Open vRAN Brings Network Elasticity, Flexibility, and Best in Class Automation," Mavenir, 2021, https://www.mavenir.com/portfolio/mavair/radio-access/vran/.
[20] "Converged multi-access and core COMAC," Open Networking Foundation, 2021, https://opennetworking.org/comac/.
[21] M. Core, "An open-source platform for building carrier-grade networks," Magma Core, 2021, https://www.magmacore.org/.
[22] free5GC, "What is free5GC?" Communication Service/Software Laboratory (CS Lab), 2019, https://www.free5gc.org/.
[23] C. Cascone, "SD-Core," Open Networking Foundation, 2021, https://opennetworking.org/sd-core/.
[24] L. Peterson, "Central office re-architected as a datacenter (CORD)," Open Networking Foundation, 2021, https://opennetworking.org/cord/.
[25] "LL-MEC," mosaic5g, https://mosaic5g.io/ll-mec/.
[26] "LIGHTeDGe," https://lightedge.io/.
[27] O. Sunay, "Aether Architecture," ONF, https://opennetworking.org/aether/.
[28] bwojtowi, "openLTE," Slashdot Media, 2021, https://sourceforge.net/projects/openlte/.
[29] "Akraino," lfedge, 2020, https://www.lfedge.org/projects/akraino/.
[30] srsRAN, "Your own mobile network," srsRAN, 2019, https://www.srslte.com/.
[31] "srsRAN Features," Software Radio Systems, 2019, https://docs.srsran.com/en/latest/feature_list.html.
[32] "Hardware Options," srsRAN, 2021, https://docs.srsran.com/en/latest/app_notes/source/hw_packs/source/index.html.
[33] "SRS," Software Radio Systems, 2012, https://www.srs.io/.
[34] "srsRAN Application Notes," Software Radio Systems, 2019, https://docs.srsran.com/en/latest/app_notes/source/index.html.
[35] "srsRAN with ZMQ Virtual Radios," Software Radio Systems, 2019, https://docs.srsran.com/en/latest/app_notes/source/zeromq/source/index.html.
[36] Almon, "Welcome to the ubuntu community installation guide," Ubuntu, 2018, https://help.ubuntu.com/community/Installation.
[37] "VirtualBox," https://www.virtualbox.org/.
[38] "srsRAN on Raspberry Pi 4," srsRAN, 2021, https://docs.srsran.com/en/latest/app_notes/source/pi4/source/index.html.
[39] "5G NSA End-to-End," https://docs.srsran.com/en/latest/app_notes/source/5g_nsa_zmq/source/index.html#g-nsa-overview.
[40] "How to Check if Your Raspberry Pi 4 Model B is Rev1.2?" Cytron, 2020, https://tutorial.cytron.io/2020/02/22/how-to-check-if-your-raspberry-pi-4-model-b-is-rev1-2/.

SADIQ IQBAL received the B.E. Degree in Electrical Engineering from QUEST University Nawabshah, Pakistan in 2016. He is presently pursuing the Master (M.Sc.) degree in Electrical and Computer Engineering from Antalya Bilim University, Antalya, Turkey.

His research interests include physical layer technology, 5G communication networks, Artificial Intelligence, Deep Learning, and IoT and its applications.

JEHAD M. HAMAMREH is the Founder and Director of WISLAB, and A. Professor with the Electrical and Electronics Engineering Department, Antalya Bilim University. He received his Ph.D. degree in telecommunication engineering and cyber systems from Istanbul Medipol University, Turkey, in 2018. Previously, he worked as a Researcher at the Department of Electrical and Computer Engineering at Texas AM University. He is the inventor of more than 20+ Patents and an author of more than 75+ peer-reviewed scientific papers along with several book chapters. His innovative patented works won the gold, silver, and bronze medals by numerous international invention contests and fairs.

His current research interests include wireless physical and MAC layers security, orthogonal frequency-division multiplexing and multiple-input multiple-output systems, advanced waveforms design, multidimensional modulation techniques, and orthogonal/non-orthogonal multiple access schemes for future wireless systems. He is a serial referee for various scientific journals as well as a TPC member for several international conferences. He is an Editor at Researcherstore, RS-OJICT journal, and Frontiers in Communications and Networks. Email: jehad.hamamreh@gmail.com OR jehad.hamamreh@researcherstore.com.