Software-Defined Radio Wireless Communication Technology Design. LibreSDR Board Validation

Conference Paper · March 2024

DOI: 10.1109/IEEECONF60226.2024.10496728

4 authors, including:

Grigoriy Fokin
St. Petersburg State University of Telecommunication
100 PUBLICATIONS **691** CITATIONS

SEE PROFILE

Software-Defined Radio Wireless Communication Technology Design. LibreSDR Board Validation

Vladimir Grigoriev, Alexander Komissarov
Department of Wireless Networks
Limited Liability Company
"Laboratory of Infocommunication Networks"
Saint Petersburg, Russia
vgrig@labics.ru, comiss2000@mail.ru

Konstantin Ryutin, Grigoriy Fokin
Software Defined Radio Laboratory
The Bonch-Bruevich Saint Petersburg State
University of Telecommunications
Saint Petersburg, Russia
ryutin.sut@gmail.com, grihafokin@gmail.com

***Abstract*—Currently, most of the technical solutions for wireless communications and navigation systems prototyping and validation in academic and scientific community employ Software Defined Radio (SDR) technology. However, wider utilizing of popular and well proven SDR platforms, such as USRP (Universal Software Radio Peripheral), LimeSDR and AD-FMCOMMS boards, is still difficult due to high cost and low availability. One of the solutions to popularize and make SDR wireless communication technology design more accessible for academic and scientific community is the emerging low-cost SDR-platform LibreSDR. Current work is devoted to experimental validation of LTE compliant transmission and reception of reference signals on physical layer, using LibreSDR board. The results of the experimental validation prove the potential of relatively low-cost LibreSDR utilization for wireless communications and navigation systems prototyping with the same performance, as well known USRP boards, among young scientists and students during their education and research in the academic and scientific community.**

Keywords—LibreSDR; LTE; SDR; Software-Defined Radio; experimental validation

I. INTRODUCTION

In recent years research in field of wireless communications has proven the effectiveness of using Software Defined Radio (SDR) technology in the design, prototyping and experimental validation of wireless communication [1] and navigation systems [2]–[11], including proprietary [2]–[4] and compliant with LTE specifications [5]–[11]. In the field of wireless communication system design, the use of SDR technology offers many advantages [12]–[15]. Flexibility of SDR hardware and software technology allows developers to customize and adapt radio systems more flexibly, so wireless communication systems can use the spectrum more efficiently by changing transmit and receive parameters in real time. SDR applications can also improve the quality of mobile services provided, for example through the use of adaptive digital signal processing algorithms, improved modulation and coding techniques. The use of SDR technology can also help improve mobile security by enabling more sophisticated algorithms to encrypt user data. Thus, the application of SDR technology allows developers to create more flexible, efficient and secure wireless communication and navigation systems, that adapt to changing conditions and requirements.

Despite proliferation of SDR technologies in research and development community in design, prototyping and validation of wireless communication and navigation systems [1]–[15], its ubiquitous spread in academic and scientific community is still difficult due to high cost and low availability of well proven SDR platforms, such as USRP (Universal Software Radio Peripheral) B210 [16], N210 [17], LimeSDR [18], AD-FMCOMMS [19] and ADALM-PLUTO [20] boards, including UHD (USRP Hardware Driver) [21] software support from Matlab[22] and LabVIEW [23]. An alternative to the above hardware platforms is the recently commercialized LibreSDR board. This study provides the main characteristics, key components, operational features and experimental validation of the LibreSDR board in the LTE compliant transmission and reception of reference signals on the LTE physical layer according to 3GPP TS 36.211 [24].

The material of the paper is organized in the following order. Section II introduces LibreSDR characteristics. Section III presents connection and configuration of transmission and reception of reference signals on LTE physical layer. Section IV describes experimental validation test in laboratory conditions. Finally, conclusions are drawn in Section V.

II. LIBRESDR CHARACTERISTICS

The LibreSDR became available about a year ago and is positioned as an analog of the popular ADALM-PLUTO or PlutoSDR SDR board [20], designed for education. LibreSDR is made in a metal case and manufactured with a CNC (computer numerical control) for efficient heat compensation. LibreSDR appearance and dimensions are in Fig. 1 and Fig. 2.

Fig. 1. External view of LibreSDR board

This project is supported by the Potanin Foundation

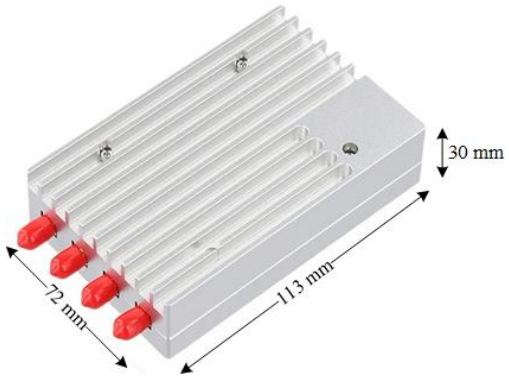

Fig. 2. LibreSDR board overall dimensions

LibreSDR is designed on a 10-layer printed circuit board, its appearance from the front and back sides is shown in Fig. 3. To reduce the cost, an 8-bit DAC (digital-to-analog converter) chip is used at the output of the PLL (Phase-Locked Loop) digital chip. The following connectors are present on the LibreSDR board (Fig. 4): 1) ewo MMCX (Female) coaxial connectors for 1 PPS and 10 MHz synchronization interfaces; 2) RJ-45 connector for Gigabit Ethernet interface, which carries IQ signal components and access to the embedded Linux console; 3) ewo USB Type-C connectors (OTG and Debug): the OTG (On-The-Go) connector is for accessing the Linux console via the serial port, and the Debug connector is for debugging and loading Xilinx firmware via the JTAG interface; both connectors are also used to supply power to the LibreSDR; 4) connector for microSD card, from which the firmware image is loaded; 5) four SMA (Female) coaxial connectors for two receiving and two transmitting channels.

Fig. 3. LibreSDR printed circuit board

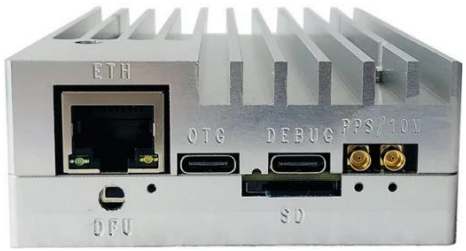

Fig. 4. LibreSDR board connectors

The device also features a hidden DFU (Device Firmware Update) button to restore the factory firmware. LibreSDR and PlutoSDR are based on RF transceiver AD9363 [25], produced by Analog Devices, but the manufacturer of LibreSDR states that, in this case, it is configured as a superior transceiver AD9361 [26], integrated in USRP. Characteristics of AD9361 and AD9363 RF transceivers are summarized in Table I.

TABLE I. CHARACTERISTICS OF AD9361 AND AD9363

	AD9361	**AD9363**
Number of channels	2 TX, 2 RX	2 TX, 2 RX
DACs and ADCs	12-bit	12-bit
TX band	47 MHz to 6.0 GHz	325 MHz to 3.8 GHz
RX band	70 MHz to 6.0 GHz	325 MHz to 3.8 GHz
Supported duplexes	TDD and FDD	TDD and FDD
Tunable channel BW	<200 kHz to 56 MHz	<200 kHz to 20 MHz

Unlike the PlutoSDR module, whose main computational element is the SoC (System on Chip) Zynq Z-7010 [27] (part number XC7Z010), LibreSDR integrates a higher performance SoC Zynq Z-7020 (part number XC7Z020). These SoCs combine two ARM Cortex A9 hardware cores (with maximum clock frequency 1 GHz) with the Artix-7 [28] family FPGA architecture. These microprocessor cores can be used for applications, that run the Linux. The main characteristics of SoC XC7Z010 and XC7Z020 chips are shown in Table II.

TABLE II. CHARACTERISTICS OF SOC XC7Z010 AND XC7Z020

	XC7Z010	**XC7Z020**
FPGA family	Artix-7	Artix-7
Programmable Logic Cells	28000	85000
Look-Up Tables (LUTs)	17600	53200
Flip-Flops	35200	106400
Block RAM	2.1	4.9
DSP Slices	80	220

Thus, using LibreSDR with Zynq Z-7020 SoC is a more promising solution in terms of developing a proprietary HDL realization of wireless system, compared to using Zynq Z-7010 SoC in PlutoSDR. Considering the cost of LibreSDR (approximately $300 at the time of this study) and the above advantages, the choice of this platform appears to be highly preferable for academic and scientific community.

III. CONNECTION AND CONFIGURATION

In the following consider procedures to connect and configure LibreSDR for its interaction with a Windows PC. After connecting OTG and Debug LibreSDR connectors to PC with USB (Type-A)-USB (Type-C) cables, 5 serial ports will appear in the Device Manager, each of them with configuration 115200 8N1 (speed 115200 baud/s, 8 bits per word, no parity bit, 1 stop bit). In Fig. 5 COM3 port is intended for access to LibreSDR OS console, and other ports are for debugging and loading the firmware by means of Xilinx via JTAG interface.

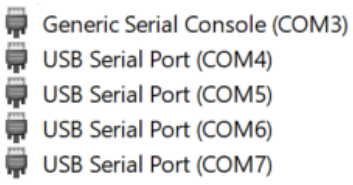

Fig. 5. LibreSDR serial ports in Device Manager

A. *Changing LibreSDR Network Settings*

By default, LibreSDR OS uses dynamic addressing by DHCP (Dynamic Host Configuration Protocol), so for convenient work with LibreSDR platform in MATLAB, it is necessary to change the addressing to static, using one of the available remote access clients, for example MobaXterm [29]. After launching the MobaXterm program, it is necessary to click on the "Session" button to open a new session, then click on the "Serial" item and select the serial port for remote access to the console with the indication of connection speed (Fig. 6).

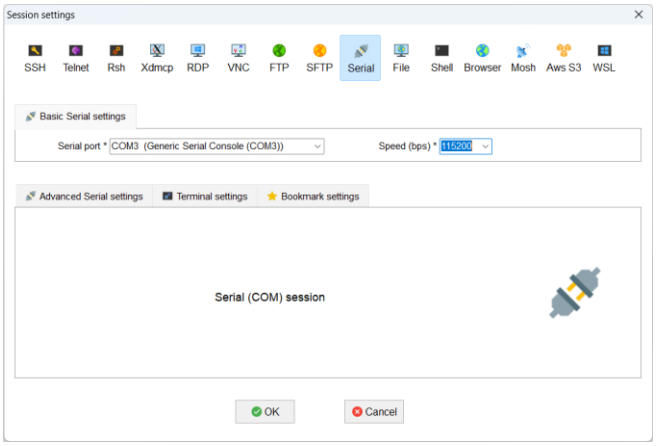

Fig. 6. Selecting a serial port and connection speed

After pressing "OK" button and entering login (analog) and password (analog), a prompt to the console will follow (Fig. 7).

```
Linaro 14.04 analog ttyGS0

analog login: analog
Password:
Last login: Thu Jan  1 00:00:44 UTC 1970 on ttyGS0
Welcome to Linaro 14.04 (GNU/Linux 4.19.0 armv7l)

 * Documentation:  https://wiki.analog.com/ https://ez.analog.com/

New release '16.04.6 LTS' available.
Run 'do-release-upgrade' to upgrade to it.

analog@analog:~$
```

Fig. 7. Invitation to the LibreSDR OS console

To change configuration file of interfaces, it is necessary to enter the superuser mode with the command `su root` and re-enter the password. Next, open file `interfaces.d` in a text editor with command `vi /etc/network/interfaces` and add desired configuration of used Ethernet port (Fig. 8).

```
# interfaces(5) file used by ifup(8) and ifdown(8)
# Include files from /etc/network/interfaces.d:
source-directory /etc/network/interfaces.d

auto eth0
iface eth0 inet static
address 192.168.2.3
netmask 255.255.255.0
```

Fig. 8. Changing the interface configuration file

After saving the file and exiting the text editor, reboot the device. When reconnecting to the remote console via serial port, you can check that the Ethernet interface is configured in the desired way by entering the command `ifconfig` (Fig. 9).

```
root@analog:/home/analog# ifconfig
eth0      Link encap:Ethernet  HWaddr 00:0a:35:00:1e:53
          inet addr:192.168.2.3  Bcast:192.168.2.255  Mask:255.255.255.0
          inet6 addr: fe80::20a:35ff:fe00:1e53/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:61 errors:0 dropped:0 overruns:0 frame:0
          TX packets:38 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:4078 (4.0 KB)  TX bytes:6421 (6.4 KB)
          Interrupt:29 Base address:0xb000
```

Fig. 9. View the configuration of network interfaces

After these manipulations you need to check the connection to the device by sending a ping request. After successful verification of communication via Ethernet interface, you can start to configure MATLAB environment for working with LibreSDR board.

B. *MATLAB Configuration*

To work with LibreSDR in MATLAB the first step is to install MinGW compiler support package [30]. Then it is necessary to install Analog Devices Transceiver Toolbox [31] and libIIO (Library Industrial I/O) support package [32]. Also, install Xilinx Zynq-Based Radio support package [33], which is required to use streaming system objects. This support package provides necessary libIIO bindings in MATLAB, used by Analog Devices system objects. To receive or transmit data, you must create the appropriate system objects in MATLAB.

Despite all the advantages of LibreSDR, there is an effect, that negatively affects streaming transmission and reception: the signal is broken, when the transmit or receive function is called again. Thus, a signal can be transmitted or received continuously in the amount of 2^{20} samples, which is the maximum buffer size of a TCP/UDP socket.

C. *Initialization of LibreSDR in Xilinx Vivado*

To debug and flash the LibreSDR platform in the Xilinx Vivado environment [34], it is necessary to click on "Open Hardware Manager" in the "Flow" item of the top menu after starting Vivado (Fig. 10). Next, select "Open target" and then click on "Auto Connect" (Fig. 11). If everything was done correctly, the SoC Zynq Z-7020 will appear in the Hardware window (Fig. 12).

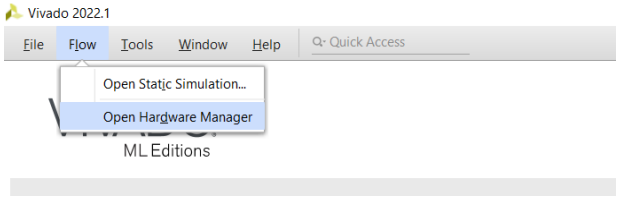

Fig. 10. Opening the Devices Manager

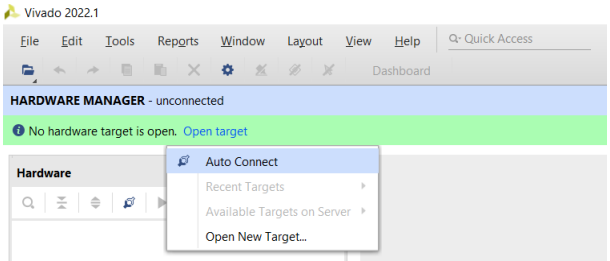

Fig. 11. Connecting to the LibreSDR

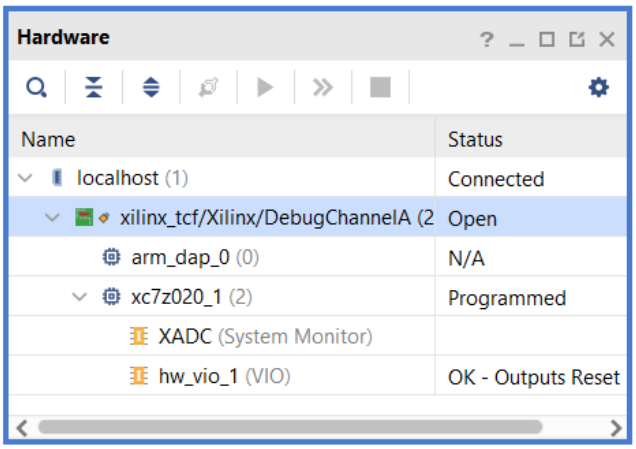

Fig. 12. Hardware window with SoC Zynq Z-7020

IV. Experimental Validation

Experimental testing of LibreSDR board was performed in MATLAB environment and include two parts: 1) transmission and reception of LTE compliant reference signals in laboratory conditions, where transmitter and receiver were implemented on two LibreSDR boards; 2) reception of LTE compliant signal on the air from the operator's base station (eNB) at known carrier frequency. The photo of the experimental layout is shown in Fig. 13. On the left side of Fig. 13 there is LTE transmitter, including a laptop, which runs MATLAB-program, which realizes the functionality of the signal transmitter at the baseband; LibreSDR is connected to laptop, which transmits the generated signals on radio frequency into the air. On the right side of Fig. 13 there is the receiver of LTE signals, which also includes a laptop with LibreSDR, connected to it. MATLAB-program on the receiver side initializes LibreSDR, receives IQ-samples from the air and calculates correlations on PSS (Primary Synchronization Signal) and SSS (Secondary Synchronization Signal) to determine Cell ID.

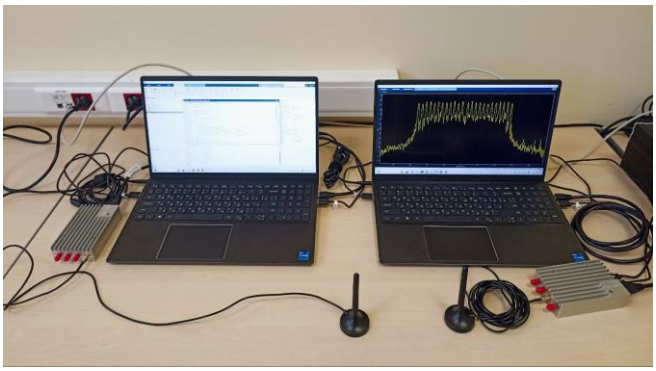
Fig. 13. Experimental Layout

A. Transmit and Receive LTE Signal with LibreSDR

In this experiment, PSS/SSS synchronization signals and Cell-Specific Reference Signal (CRS) were generated at the transmitter side. Transmit and receive parameters were as follows: 1) the carrier frequency is 5 GHz (to verify the correct configuration of the AD9363 is similar to the AD9361); 2) the bandwidth is 20 MHz; 3) sample rate is 30.72 MHz; 4) the Cell ID parameter was set to 377.

The correctness of the system operation was checked by matching Cell ID, initialized at the transmitter, to Cell ID, estimated by PSS and SSS correlations at the receiver. After transmitting and receiving LTE signal from the air, spectrum was plotted in Fig. 14. After synchronizing the LTE frames and calculating the PSS and SSS correlations, a Cell ID was obtained on the receiver side that matched the one on the transmitter side (Fig. 15). Since the MIB (Master Information Block) was not generated on the transmitter side, the transmitted signal bandwidth was not correctly extracted on the receiver side (NRBDL parameter in Fig. 15).

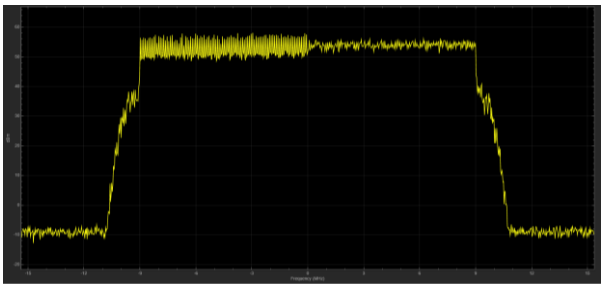
Fig. 14. Spectrum of the signal received from LibreSDR

```
Performing cell search...
Timing offset to frame start: 3095 samples
Cell-wide settings after cell search:
            NDLRB: 6
       DuplexMode: 'FDD'
     CyclicPrefix: 'Normal'
          NCellID: 377
        NSubframe: 0


Performing frequency offset estimation...
Frequency offset: 492.469Hz
```
Fig. 15. LibreSDR transmitter parameters returned after reception

B. LTE Signal Receiving from the Operator's eNB

In this experiment, the reception of the operator's eNB, defined with the help of Netmonitor [35] application, was carried out. This application displays main parameters of the eNB, to which this cell phone is connected (Fig. 16). The most important parameters in this experiment are: absolute EARFCN (E-UTRA Absolute Radio Frequency Channel Number), frequency band number, bandwidth, duplex mode and PCI (Physical Cell ID). Knowing EARFCN $N_{DL} = 38100$ and the frequency band number (in this case, it is 38), it is easy to calculate the carrier frequency F_{DL} in MHz, on which the operator's BS broadcasts by the formula (1).

$$F_{DL} = F_{DL\_Low} + 0.1 \times (N_{DL} - N_{DL\_Offset}) = 2605 \text{ MHz}, \quad (1)$$

where $F_{DL\_Low} = 2570$ MHz is the lower frequency of the given band and $N_{DL\_Offset} = 37750$ is the offset used to calculate the EARFCN in the downlink. Both of these parameters are defined in [36]. After receiving the LTE signal from the operator's eNB, the spectrum was plotted in Fig. 17. After synchronization of LTE frames and calculation of correlations by PSS and SSS on the receiver side, Cell ID was obtained, which coincides with one, displayed in Netmonitor (Fig. 18). In this case, MATLAB-program on the receiver side also decoded the information block MIB, the parameters of which coincide with displayed in the Netmonitor program.

Fig. 16. Screenshot of the Netmonitor application

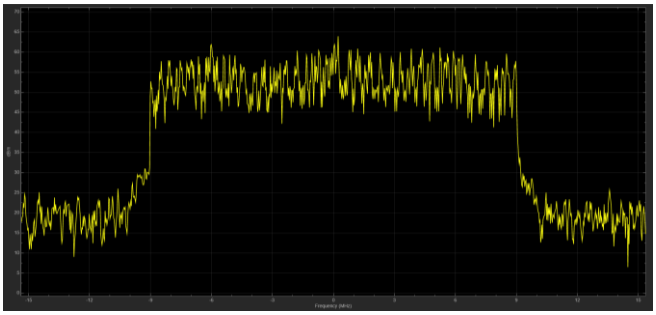

Fig. 17. Spectrum of the signal received from the operator BS

```
Resampling from 30.720Ms/s to 1.920Ms/s for cell search / MIB decoding...

Performing cell search...
Timing offset to frame start: 1411 samples
Cell-wide settings after cell search:
          NDLRB: 6
      DuplexMode: 'TDD'
    CyclicPrefix: 'Normal'
         NCellID: 220
       NSubframe: 0


Performing frequency offset estimation...
Frequency offset: 5929.836Hz
Performing OFDM demodulation...

Performing MIB decoding...
Cell-wide settings after MIB decoding:
           NDLRB: 100
       DuplexMode: 'TDD'
     CyclicPrefix: 'Normal'
          NCellID: 220
        NSubframe: 0
        TDDConfig: 0
              SSC: 0
         CellRefP: 4
    PHICHDuration: 'Normal'
               Ng: 'One'
           NFrame: 783
```

Fig. 18. Parameters of the operator eNB received after reception

Thus, both experiments confirm the correct operation of LibreSDR for receiving and transmitting LTE signal.

V. Conclusion

In the course of current project, recently emerged SDR-platform LibreSDR was experimentally validated. Experiment results revealed, that LibreSDR is not inferior in features and capabilities to well-known SDR platforms such as USRP, LimeSDR and AD-FMCOMMS, while having a lower cost.

Acknowledgment

This project is supported by the Potanin Foundation

References

[1] G. Fokin, D. Volgushev, A. Kireev, D. Bulanov and V. Lavrukhin, “Designing the MIMO SDR-based LPD transceiver for long-range robot control applications,” 2014 6th International Congress on Ultra Modern Telecommunications and Control Systems and Workshops (ICUMT), St. Petersburg, Russia, 2014, pp. 456-461.

[2] G. Mashkov, E. Borisov and G. Fokin, “Experimental validation of multipoint joint processing of range measurements via software-defined radio testbed,” 2016 18th International Conference on Advanced Communication Technology (ICACT), PyeongChang, Korea (South), 2016, pp. 268-273.

[3] G. Mashkov, E. Borisov and G. Fokin, “Experimental validation of multipoint joint processing of range measurements via software-defined radio testbed,” 2017 19th International Conference on Advanced Communication Technology (ICACT), PyeongChang, Korea (South), 2017, pp. 979-984.

[4] G. Mashkov, E. Borisov and G. Fokin, “Positioning accuracy experimental evaluation in SDR-based MLAT with joint processing of range measurements,” 2016 International Conference on Radar, Antenna, Microwave, Electronics, and Telecommunications (ICRAMET), Jakarta, Indonesia, 2016, pp. 7-12.

[5] K. E. Ryutin and G. A. Fokin, “Software-Defined Radio Network Positioning Technology Design. MIB Transceiver Development,” 2023 Seminar on Signal Processing, Saint Petersburg, Russian Federation, 2023, pp. 115-119.

[6] G. Fokin, D. Volgushev, V. Grigoriev and K. Ryutin, “Software-Defined Radio Network Positioning Technology Design. Field Experiment Demonstrator,” 2023 Intelligent Technologies and Electronic Devices in Vehicle and Road Transport Complex (TIRVED), Moscow, Russian Federation, 2023, pp. 1-6.

[7] G. Fokin, K. Ryutin, V. Grigoriev and V. Bobrovskiy, “Software-Defined Radio Network Positioning Technology Design. Synchronization Subsystem,” 2023 Systems of Signal Synchronization, Generating and Processing in Telecommunications (SYNCHROINFO, Pskov, Russian Federation, 2023, pp. 1-6.

[8] G. Fokin and D. Volgushev, “Software-Defined Radio Network Positioning Technology Design. Receiver Processing Procedures,” 2023 Systems of Signals Generating and Processing in the Field of on Board Communications, Moscow, Russian Federation, 2023, pp. 1-7.

[9] D. Volgushev and G. Fokin, “Software-Defined Radio Network Positioning Technology Design. Receiver Development,” 2022 Intelligent Technologies and Electronic Devices in Vehicle and Road Transport Complex (TIRVED), Moscow, Russian Federation, 2022, pp. 1-6.

[10] G. Fokin and D. Volgushev, “Software-Defined Radio Network Positioning Technology Design. Transmitter Development,” 2022 IEEE International Black Sea Conference on Communications and Networking (BlackSeaCom), Sofia, Bulgaria, 2022, pp. 153-158.

[11] G. Fokin and D. Volgushev, “Software-Defined Radio Network Positioning Technology Design. Problem Statement,” 2022 Systems of Signals Generating and Processing in the Field of on Board Communications, Moscow, Russian Federation, 2022, pp. 1-6.

[12] T. F. Collins, R. Getz, D. Pu, A. M. Wyglinski. Software-defined radio for engineers. Artech House, 2018.

[13] R. W. Stewart, K. W. Barlee, D. S. W. Atkinson, L. H. Crockett. Software defined radio using MATLAB & Simulink and the RTL-SDR. Strathclyde Academic Media, 2015.

[14] J. M. Reyland. Software Defined Radio: Theory and Practice. . Artech House, 2024.

[15] C. R. Johnson, Jr., W. A. Sethares, A. G. Klein. Software Receiver Design: Build Your Own Digital Communication System in Five Easy Steps. Cambridge University Press, 2011.

[16] USRP B210. Ettus Research. Accessed: February 22, 2024. [Online]. Available: https://www.ettus.com/all-products/ub210-kit/.

[17] USRP N210. Ettus Research. Accessed: April 17, 2024. [Online]. Available: https://www.ettus.com/all-products/un210-kit/

[18] LimeSDR. Accessed: February 22, 2024. [Online]. Available: https://limemicro.com/products/boards/limesdr/.

[19] AD-FMCOMMS3-EBZ Evaluation Board. Accessed: February 22, 2024. [Online]. Available: https://www.analog.com/en/resources/evaluation-hardware-and-software/evaluation-boards-kits/eval-ad-fmcomms3-ebz.html

[20] ADALM-PLUTO Evaluation Board. Accessed: February 22, 2024. [Online]. Available: https://www.analog.com/en/resources/evaluation-hardware-and-software/evaluation-boards-kits/adalm-pluto.html

[21] UHD (USRP Hardware Driver). Ettus Research. Accessed: April 17, 2024. [Online]. Available: https://www.ettus.com/sdr-software/uhd-usrp-hardware-driver/

[22] USRP Support from Communications Toolbox. MathWorks. Accessed: April 17, 2024. [Online]. Available: https://www.mathworks.com/hardware-support/usrp.html

[23] LabVIEW. NI. Accessed: April 17, 2024. [Online]. Available: https://www.ni.com/ru-ru/shop/labview.html

[24] 3GPP TS 36.211. Evolved Universal Terrestrial Radio Access (E-UTRA); Physical channels and modulation. V17.0.0, 2021-12.

[25] AD9363 Datasheet and Product Info. Accessed: April 17, 2024. [Online]. Available: https://www.analog.com/en/products/ad9363.html

[26] AD9361 Datasheet and Product Info. Accessed: April 17, 2024. [Online]. Available: https://www.analog.com/en/products/ad9361.html

[27] Zynq 7000 SoC. Accessed: April 17, 2024. [Online]. Available: https://www.xilinx.com/products/silicon-devices/soc/zynq-7000.html

[28] Artix 7 FPGA Family. Accessed: April 17, 2024. [Online]. Available: https://www.xilinx.com/products/silicon-devices/fpga/artix-7.html

[29] MobaXterm free Xserver and tabbed SSH client for Windows. Accessed: February 22, 2024. [Online]. Available: https://mobaxterm.mobatek.net/.

[30] MATLAB Support for MinGW-w64 C/C++/Fortran Compiler. Accessed: February 22, 2024. [Online]. Available: https://www.mathworks.com/matlabcentral/fileexchange/52848-matlab-support-for-mingw-w64-c-c-fortran-compiler.

[31] Analog Devices, Inc. Transceiver Toolbox. Accessed: February 22, 2024. [Online]. Available: https://www.mathworks.com/matlabcentral/fileexchange/72645-analog-devices-inc-transceiver-toolbox.

[32] IIO System Object. Accessed: February 22, 2024. [Online]. Available: https://wiki.analog.com/resources/tools-software/linux-software/libiio/clients/matlab_simulink.

[33] Get Started with Communications Toolbox Support Package for Xilinx Zynq-Based Radio. Accessed: February 22, 2024. [Online]. Available: https://www.mathworks.com/help/supportpkg/xilinxzynqbasedradio/getting-started.html.

[34] Vivado Overview. Accessed: February 22, 2024. [Online]. Available: https://www.xilinx.com/products/design-tools/vivado.html.

[35] Netmonitor: Cell & WiFi. Accessed: February 22, 2024. [Online]. Available: https://play.google.com/store/apps/details?id=com.parizene.netmonitor.

[36] 3GPP TS 36.104 V18.4.0 (2023-12) Evolved Universal Terrestrial Radio Access (E-UTRA); Base Station (BS) radio transmission and reception (Release 18).