International Journal of Engineering & Technology, 7 (4.6) (2018) 279-284

SPC

International Journal of Engineering & Technology

Website: www.sciencepubco.com/index.php/IJET

Research paper

Implementation of Transceiver module for SDR system using ADALM PLUTO platform

Sowjanya.P[1]*, Satyanarayana.P[2]

[1] *Research Scholar, Department of Electronics and Communication Engineering, KLEF, Vaddeswaram, Andhra Pradesh, India.*
[2] *Professor, Department of Electronics and Communication Engineering, KLEF, Vaddeswaram, Andhra Pradesh, India.*
**Corresponding author E-mail:ponnalurusowjanya@ymail.com*

Abstract

Software Defined Radio (SDR) provides a comprehensive radio communication platform, based on which new technology can be used through software update. This leads to a large-scale reduction in expansion costs and enables the product to maintain technology development. The SDR platform can be set up with an open, standard, and programmable hardware platform, based on which the functions of the radio can be perceived by adding appropriate software modules. In this platform, the transformation and expansion of the radio functions are done in a software version without the need for a modification of the equipment. Such software radio station can easily communicate with the current or upcoming radio stations. In this article, we analyze SDR evolution and various platforms and implement various modulation techniques with the aim of successfully transferring a message wirelessly over-the-air using ADALM-PLUTO SDR platform by Analog Devices.

Keywords: *ADALM PLUTO; Evolution; Platforms; Software Defined Radio*.

1. Introduction

Software Defined Radio (SDR) is a present trending hardware device which is designed using fixed frontend hardware and software components on programmable devices like Digital Signal Processors (DSPs) and Field Programmable Gate Arrays (FPGAs). Some of the functions done in conventional radios are made with specific components, like coding / decoding, modulators / demodulators etc. But in SDR, all signal processing is done in one specialized equipment. Some of the hardware components in conventional radio can be changed with components which are implemented in software by using SDR. SDR offers several improvements on the conventional radios, including re-configurability and modern market. SDR also has long-term cycles as they can easily adapt to the needs of the future. The most important features for achieving this success is the software upgrades, reduced costs and processing power. SDRs are also used in many electronic precision instrumentation devices and cell phones. In SDR the user can select the frequency band depending upon their availability and integrate different connection technologies such as LTE, WiMAX, GSM or Wi-Fi in a single interface, but this can't be possible in conventional radio systems.

The two major portions presented in software-defined radio transceiver are: (a) an analog front-end and (b) digital signal processing modules. Analog front-end performs narrowband frequency down conversion to an analog-to-digital converted signals and the remaining signal processing flow i.e., modulation/ demodulation, filtering, and channel coding/ decoding operations were done by the digital signal processing modules. The conventional data flow in an SDR transceiver system is shown in Fig, 1. SDR implements most of the transmitter and receiver baseband signals using software domain. The ADC and DAC conversion were done further in the RF block. An ideal SDR can be designed by extending the programmability to the RF front end. Instead of replacing entire hardware in a device, by downloading a software, performance of a system can be upgraded using SDR. By implementing components in software, we are having advantage of flexibility and distinct frequency bands. All the signal processing in SDR can do digitally only when we move the ADC and DAC very close to the antenna and this becomes a major task for SDR. But, due to some of the technical problems we can't move the AD/DA conversion close to antenna.

Implementation of all the modules in digital domain was done in software and several processing units like Field Programmable Gate Arrays (FPGAs), General Purpose Processors (GPP), Graphics Processing Units (GPUs), Digital Signal Processors (DSP), or their combination can run together. Analog portion of the digital transmission was handled by RF front-end. Moving ADC/DAC closer to the antenna will depends on their sampling capabilities. To convert RF signal into digital data, maintaining Nyquist frequency sample rate is necessary and for capturing the information with higher resolution, the signal level should be high. That's why depending upon the powerful ADC and DAC the development of SDR platforms is driven.

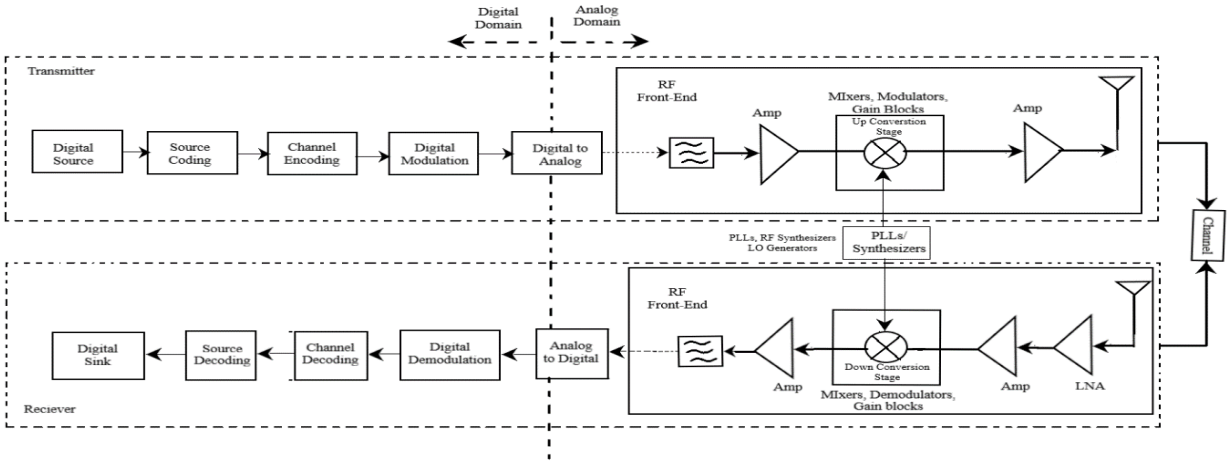


Fig. 1: SDR Transceiver system

2. Evolution of SDR

The SDR was launched by Joseph Mitola in 1992. Although the SDR model was previously offered by Hoeher and Lang in 1988. SDR as the technology comes first with a widely funded SDR creative work called "SpeakEasy I / II" in the US. The SpeakEasy I system used the Texas Instruments TMS320C40 processor (40 MHz), and the SpeakEasy II platform included the Field Programmable Gate Array (FPGA) for the first time. Digital Modulator Radio (DMR) was created by U.S. Navy with many waveforms and modes which can be controlled remotely using Ethernet interface.

The development of processing technologies and SDR technologies was shown in Table. 1 and it analyses that the progress of components increases with higher computational power and flexibility, and it enables best SDR platforms. The development of SDR platforms was done only after 2000 because of using powerful FPGAs and DSPs. After increasing the usage of wireless mobiles, we came to know different technical challenges that relate to various wireless standards, services and programs that all should support, often in the same operating environment. The significant investment in wireless network equipment's will increase if there is a case where, lot of wireless infrastructure layers are required to provide simultaneous networks in one region. Additionally, supporting multiple wireless networks at the same time may cause collapse in network performance related to interference.

The advantages of reconfigurable radio technologies are less infrastructure cost, better wireless connectivity in different networks and flexibility that can eventually improve network performance.

SDR spread from the military since 1990s and later it was used in the commercial sector, and mobile networks are the natural habitat. SDR products for mobile base stations are launched by a number of companies like Etherstack, Vanu and Airspan. A software radio regulation named as the Anywave GSM base station approved the first SDR product which was released by Vanu in 2005. Anywave base station software implemented the modules of the BSS like the TRAU (transcoder and rate adaptation unit), BTS (base transceiver station) and BSC (base station controller). SDR base stations will play a pivotal role for 3G networks, even though the successful investment is focused on SDR technology, the reality of viable use of SDR technology remains far.

SDR technology is a solution to prototype wireless transceivers and networks due to its versatility, cost and using its customizable features to speed up development. Applications of SDR are national protection, public safety, transport and education. Additionally, it is important that multilevel hardware components are compatible with the ergonomic development software environment, providing effective prototyping and development.

Table 1: Processing technology versus SDR technology

Technology	1970-1979	1980-1989	1990-1999	2000-2009	2010-2020
Processing Technology: Microprocessors, DSP's, FPGS's, GPU's	* Micro Intel 8086 * DSP: TI TMS5100	* Micro: Intel 386/ARM * DSP: TI TMS32010 * FPGA: Xilinx XC2064	* Micro: Intel 486 * Micro: Pentium * FPGA: 4000/Spartan * DSP: TI C2000	* FPGA: Virtex 1-5 * DSP: ADI Blackfin * GPGPU: CUDA * DSP: TI C5000-C6000	* Micro: Intel i3, i5, i7 * FPGA: Virtex 6-7 * Micro: ARM Cortex A9

3. SDR Platforms

Although SDR technology has a great potential to access the telecommunications community and more efficient to make communication system prototyping, there are many challenges that have been resolved. To ensure wide spread of SDR technology for communication systems and networks, you need to reach the following conditions:

• SDR requires less computational horsepower and despite its cost it works on wide range of carrier frequencies and bandwidths and are handy

• SDR development companies provide a high level of communication technology to the SDR platform, a rich set of algorithms, modules and features, and the level of software accessibility (i.e., shallow learning curve)

• In real time experiments SDR hardware and powerful technical computing software enables communication technologists to use reliable software models and tools.

Fortunately, the latest achievements of SDR technology have reached the above conditions and also in prototyping communication systems and networks more accessible to the telecommunications industry.

For smaller tested labs, commercially available SDR platforms offer low-cost hardware and software solutions for rapid experimental evaluation of minor software wireless networks. A summary of compatibility between some of the existing software tools and various SDR platforms were represented in Table 2. ADALM-PLUTO SDR platform by Analog Devices was used in this test due to its capabilities.

Table 2: Various SDR Platforms and their compatibility

SDR Platform	Hardware	Interface To Host computer	Frequency range	No. of TX/RX antennas	ADC bits/ speed	DAC bits/ speed	RF Band Width per I/Q channel	Software Compatibility
USRP B210	Xilinx Spartan 6 XC6SLX150	USB 3.0	70 MHz – 6 GHz	2/2	12 bits/61.44 MS/s	12 bits/ 61.44 MS/s	30.72 MHz	GNU Radio, C++ and Python APIs
USRP B200	Xilinx Spartan 6 XC6SLX75 FPGA	USB 3.0	70 MHz – 6 GHz	1/1	12 bits/61.44 MS/s	12 bits/ 61.44 MS/s	56 MHz	GNU Radio, C++ and Python APIs
USRP X310	Xilinx Kintex-7 XC7K410T	Gigabit Ethernet, 10 Gigabit Ethernet, PCIe	DC – 6 GHz	2/2	14 bits/ 200 MS/s	16 bits/ 800 MS/s	120MHz	GNURadio, C++, API/Python
USRP X300	Xilinx Kintex-7 XC7K325T	Gigabit Ethernet, 10 Gigabit Ethernet, PCIe	DC – 6 GHz	2/2	14 bits/ 200 MS/s	16 bits/ 800 MS/s	120MHz	GNURadio, C++, API/Python
USRP N310	Xilinx Zynq 7100	Type A USB Host,	10 MHz to 6 GHz	4/4	16 bits/ 153.6 MS/s	14 bits/ 122.88 MS/s	100 MHz	GNU Radio, C/C++, Python
USRP N210	Xilinx Spartan 3A-DSP 3400 FPGA	Gigabit Ethernet	DC – 6 GHz	2/2	14-bits/ 100 MS/s	16-bit/ 400 MS/s	25 MHz	GNU Radio, LabVIEW and Simulink
USRP N200	Xilinx Spartan 3A-DSP 1800 FPGA	Gigabit Ethernet	DC – 6 GHz	2/2	14-bits/ 100 MS/s	16-bit/ 400 MS/s	25 MHz	GNU Radio, LabVIEW and Simulink
USRP E310	Xilinx Zynq 7020	USB	70 MHz to 6 GHz	2/2	12 bits/ 61.44 MS/s	12 bits/ 61.44 MS/s	56 MHz	GNU Radio, C/C++, Python
BLADE RF	Altera Cyclone 4E	USB 3.0	300MHz - 3.8GHz	2/2	12 bits/ 40MS/s	12 bits/ 40MS/s	28MHz	GNURadio
Lime SDR	Altera Cyclone 4E	USB 3.0, PCIe	100 kHz – 3.8 GHz	2/2	12 bits/ 61.44 MS/s	12 bits/ 61.44 MS/s	61.44 MHz	GNURadio
Matchstiq	Xilinx Spartan6 LX45T	Gigabit ethernet, USB 2.0	< 1 MHz to 6 GHz	1/1	12 bits/ 61.44 MS/s	12 bits/ 61.44 MS/s	50 MHz	GNURadio
AD-FMCOMMS5-EBZ	Xilinx Zynq 7000	FMC then USB 2.0 or Gigabit Ethernet.	70 MHz – 6 GHz	4/4	12 bits/ 61.44 MS/s	12 bits/ 61.44 MS/s	<200 kHz to 56 MHz	MATLAB/Simulink, GNU Radio , C
AD-FMCOMMS4-EBZ	Xilinx Zynq 7000	FMC then USB 2.0 or Gigabit Ethernet.	70 MHz – 6 GHz	1/1	12 bits/ 61.44 MS/s	12 bits/ 61.44 MS/s	<200 kHz to 56 MHz	MATLAB/Simulink, GNU Radio , C
AD-FMCOMMS3-EBZ	Xilinx Zynq 7000	FMC then USB 2.0 or Gigabit Ethernet.	70 MHz – 6 GHz	2/2	12 bits/ 61.44 MS/s	12 bits/ 61.44 MS/s	<200 kHz to 56 MHz	MATLAB/Simulink, GNU Radio , C
AD-FMCOMMS2-EBZ	Xilinx Zynq 7000	FMC then USB 2.0 or Gigabit Ethernet.	2400 – 2500 MHz	2/2	12 bits/ 61.44 MS/s	12 bits/ 61.44 MS/s	<200 kHz to 56 MHz	MATLAB/Simulink. GNU Radio , C
ANAN-8000DLE	Altera Cyclone 4E	Gigabit Ethernet	0 kHz - 61.44 MHz	1/1	16 bits	16 bits	10KHz	GNU Radio
ADALM-PLUTO	Xilinx Zynq 7010	USB 2.0, Ethernet & WLAN with USB-OTG adapter	325 MHz – 3.8 GHz	1/1	12 bits/ 61.44 MS/s	12 bits/ 61.44 MS/s	20 MHz	MATLAB, Simulink, GNU Radio sink and source blocks

The basics of Radio Frequency (RF), wireless communications and Software-Defined Radio (SDR) are easy to commence in ADALM-PLUTO active learning component. This component will useful for developing a foundation in communications and real-world RF. It can be used for both teaching and research purpose. It is having two different transmit and receive channels which is worked as full duplex. The frequency range of this active learning component was from 325 MHz to 3800 MHz at a speed of 61.44 MSPS. It was supported by libiio drivers, PlutoSDR supports Windows, Linux and OS X allowing us to study on a range of devices. Fig. 2 shows the setup of ADALM PLUTO with Windows OS based Laptop.

Features

- Compact self-contained RF learning component
- Economical experimentation platform
- Built on Analog Devices AD9363 Highly Integrated RF Agile Transceiver and Xilinx Zynq Z-7010 FPGA
- RF coverage from 325MHz to 3.8GHz
- Up to 20MHz of instantaneous bandwidth
- Flexible rate, 12-bit ADC and DAC
- Single transmitter and single receiver, half or full duplex
- MATLAB Simulink support
- GNU Radio sink and source blocks
- Libiio, a C, C++, C#, and Python API
- USB 2.0 Interface
- High quality plastic enclosure

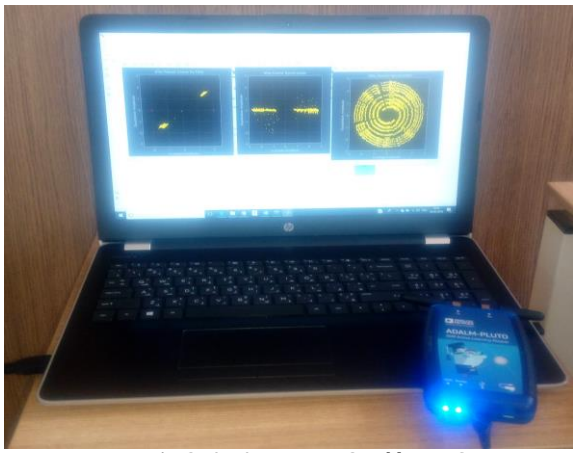


Fig. 2: ADALM PLUTO Table Top Setup

4. Design of Software Defined Radio Transceiver

In this article, using SDR an introduction of basic communication systems are demonstrated. We introduce the operations of SDR reception and transmission of a message with some simple modulation techniques like BPSK, QPSK and 8PSK. For this purpose, we used an ADALM-PLUTO module with center-frequency of 915MHz for transmitting and receiving purpose. Because of having fullduplex feature in this module it is advantage for using transmission and reception in one module. In this experiment we build the transmitting and receiving blocks in MATLAB SIMULINK software. The transmitting and receiving block diagrams are shown in Fig. 3 and Fig. 4.

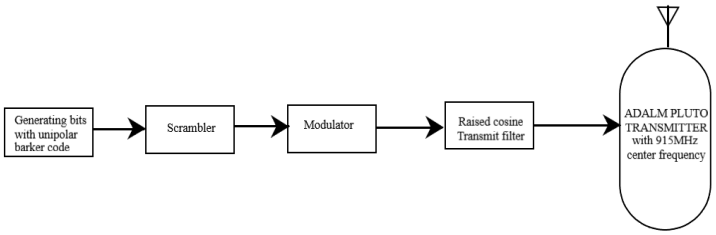


Fig. 3: SDR Transmitter block diagram

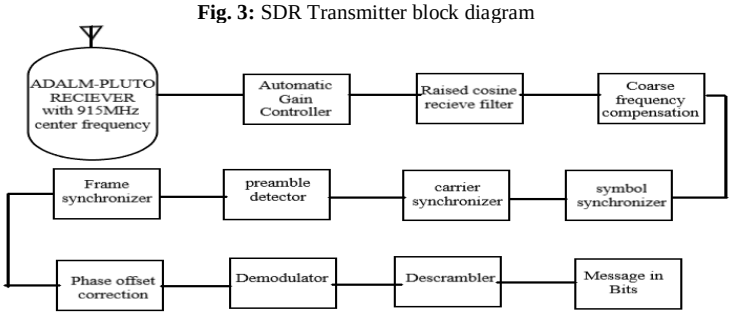


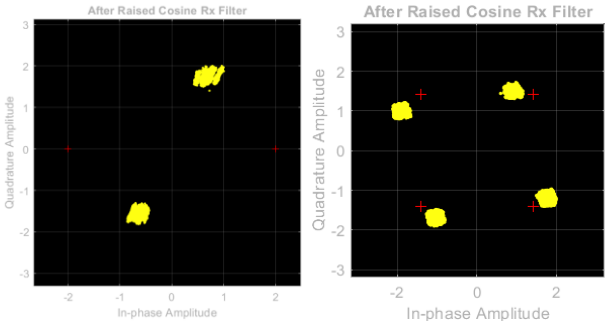
Fig. 4: SDR Receiver block diagram

5. Results:

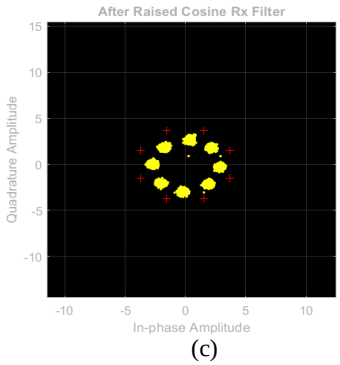
At transmitter side we converted the input message into bits and added with a barker code which is used for phase offset correction after the scrambling process was code and the message was modulated using three various modulation techniques those are BPSK, QPSK and 8PSK. Square root raised cosine transmitter filter is used before transmitting the bits with interpolation of 2. Digital to Analog conversion was made in transmitter block presented in the module and the signal was transmitted through transmitting antenna with a central frequency of 915MHz.

At the receiver side with same central frequency used in transmitter the receiver will receive the signal. The Analog to digital conversion was done in the module and after receiving the signal automatic gain controller was used to control the signals and the signal was passed through root raised cosine received filter the constellation diagram of signal after passing through root raised cosine filter for three modulations was shown in Fig. 3. After that synchronization was done for correction of time offset and those synchronizations are divided as symbol synchronizer and carrier

synchronizer and the constellation diagrams for the signals after doing the synchronization techniques was shown in Fig. 4 and Fig. 5. Using the barker bits which was send with message bits we can do frame synchronization and phase offset correction. Frame synchronization is nothing but detecting the starting of a frame. Finally, demodulation and descrambling were done for recovering original message bits.



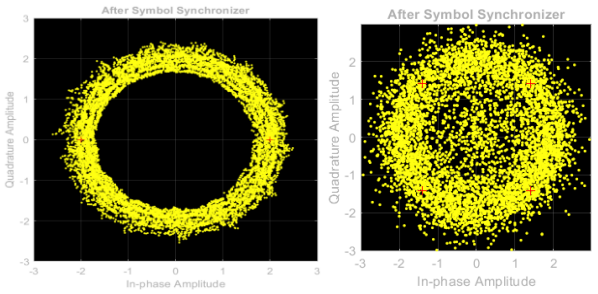
(a) (b)



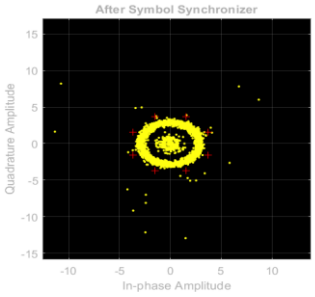
(c)

Fig. 3: Constellation diagram after passing through Root Raised cosine filter for (a) BPSK (b) QPSK and (c) 8PSK

The raised cosine filter is used in wireless transmission to pulse-shape the chip stream output before it is modulated to the RF. The spectrum is bandwidth limited to avoid interferences with neighbour symbols. The Symbol Synchronizer block corrects for symbol timing clock skew for different modulation schemes. The Carrier Synchronizer block compensates for carrier frequency and phase offsets using a closed-loop approach for various modulation schemes.

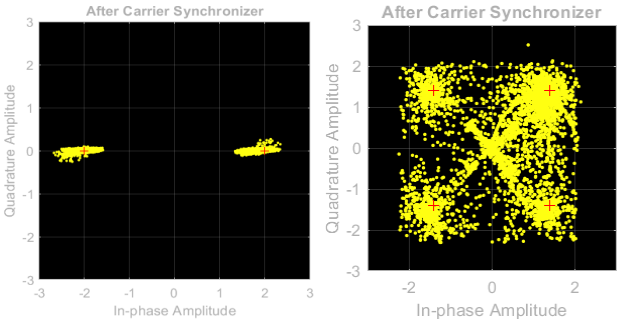


(a) (b)

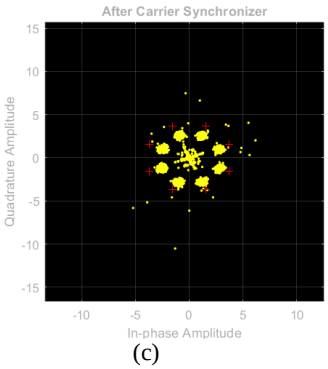


(c)

Fig. 4: Constellation diagram after passing through Symbol Synchronizer for (a) BPSK (b) QPSK and (c) 8PSK



(a) (b)



(c)

Fig. 5: Constellation diagram after passing through Carrier Synchronizer for (a) BPSK (b) QPSK and (c) 8PSK

6. Conclusion:

This article has evaluated the source and development of SDR and gathered some of the information about commercially available hardware and software platforms and categorised them according to their capability to promote rapid prototyping, testing, and evaluation. We have presented design and implementation of a transceiver module for software defined radio system with ADALM PLUTO platform. This module has number of attractive features such as being low cost, small size, low power consumption, easy to carry, easy to configure and easy to deploy. In future we must develop new modulation techniques like OFDM in SDR platform and compare the BER performance for different SNR values.

References

[1] Raquel G. Machado, Alexander M. Wyglinski, " Software-Defined Radio: Bridging the Analog–Digital Divide", *Proceedings of the IEEE*, Vol.103, No.3, March 2015, 0018-921, 10.1109/JPROC.2015.2399173.

[2] Víctor P. Gil Jiménez, Alejandro Lancho Serrano, Borja Genovés Guzmán, and Ana García Armada," Learning Mobile Communications Standards through Flexible Software Defined Radio Base Stations", *IEEE Communications Magazine*, May 10.1109/MCOM.2017.1601219

[3] George Sklivanitis, Adam Gannon, Stella N. Batalama, and Dimitris A. Pados , " Addressing Next-Generation Wireless Challenges

with Commercial Software-Defined Radio Platforms", *IEEE Communications Magazine*, January 2016, 0163-6804/16.

[4] Alexander M. Wyglinski, Don P. Orofino, Matthew N. Ettus, and Thomas W. Rondeau, "Revolutionizing Software Defined Radio: Case Studies in Hardware, Software, and Education", *IEEE Communications Magazine*, 0163-6804/16.

[5] http://www.analog.com/en/design-center/evaluation-hardware-and-software/evaluation-boards-kits/adalm-pluto.html

[6] David Carey, Robert Lowdermilk, and Michael Spinali, " Testing Software Defined and Cognitive Radios using Software Defined Synthetic Instruments", *IEEE Instrumentation & Measurement Magazine*, April 2015, 1094-6969/15.

[7] XIN CAI, MINGDA ZHOU, AND XINMING HUANG, " Model-Based Design for Software Defined Radio on an FPGA", *IEEE Access*, Vol.5 2017.

[8] Eric A. M. Klumperink and Bram Nauta, "Software Defined Radio Receivers Exploiting Noise Cancelling: A Tutorial Review", *Integrated Circuits for Communications*, October 2014, 0163-6804/14.

[9] Patricia Atungire, Talha Faizur Rahman, Fabrizio Granelli, and Claudio Sacchi, "Open-Field Emulation of Cooperative Relaying in LTE-A Downlink Using the GNU Radio Platform", *IEEE Network*, September/October 2014. 0890-8044/14.

[10] Carlos Ribeiro, Atı ´lio Gameiro, "A software-defined radio FPGA implementation of OFDM-based PHY transceiver for 5G", Analog Integr Circ Sig Process, 91:343–351, February 2017.