Software-defined radio (SDR) versus Digital Signal Processor (DSP) hardware - a comparison of approaches in teaching RF communications at Sofia University

Emil E. Vladkov, Stanimir T. Kolev

Teaching modern communication technology undoubtedly requires students to be familiar with the latest standards for digital radio frequency transmission and modulation. In times of rapid development in this field, however, this turns out to be a real challenge, as communication standards change literally in months. The introduction of new, more powerful technologies for fast transmission over radio frequency networks requires a versatile approach to familiarization with coding and modulation methods. Such an approach provides the so-called software-defined radio, which is about to replace almost completely specific hardware and software systems based on specialized radio frequency interfaces and digital signal processors. The purpose of the present study is to reveal the advantages and disadvantages of the two approaches for studying modern radio frequency communications within the framework of the respective courses offered to students at the Faculty of Physics of the Sofia University "St. Kliment Ohridski", namely the use of Analog Devices ADALM-Pluto SDR, representative of the software-defined radio modules, and the study of RF digital transmission methods using a digital signal processor development system and purpose-built hardware, representing the older approach to teaching the subject.

Keywords – (DSP, Radio Frequency Communications, Software-defined Radio, Teaching SDR, ADALM-Pluto, GNU Radio Companion).

Софтуерно-дефинирано радио (SDR) или хардуер, използващ цифрови сигнални процесори (DSP) – сравнение на подходите в обучението по радиочестотни комуникации в Софийския университет (Емил Е. Владков, Станимир Т. Колев). *Преподаването на съвременни комуникационни технологии несъмнено изисква студентите да са запознати с най-новите стандарти за цифрово радиочестотно предаване и модулация. Във времена на бурно развитие в тази сфера обаче това се оказва истинско предизвикателство, тъй като стандартите за комуникация се променят буквално за месеци. Въвеждането на нови, по-мощни технологии за бърз пренос на данни чрез радиочестотни мрежи изисква универсален подход за запознаване с методите на кодиране и модулация. Подобен подход осигурява така нареченото софтуерно-дефинирано радио, което е на път да замени почти напълно специфичните хардуерни и софтуерни системи, базирани на специализирани радиочестотни интерфейси и цифрови сигнални процесори. Целта на настоящото изследване е да разкрие предимствата и недостатъците на двата подхода за изучаване на съвременните радиочестотни комуникации в рамките на съответните дисциплини, предлагани на студентите от Физическия факултет на СУ "Св. Климент Охридски", а именно: използването на ADALM-Pluto SDR-модула на Analog Devices, представител на софтуерно-дефинираните радио модули, и изучаването на радиочестотни цифрови методи за предаване с помощта на развойна система на цифров сигнален процесор и специално създаден за целта хардуер, представляващ по-стария подход към преподаването на материята.*

The acquaintance of the students of the engineering majors of the Faculty of Physics of the Sofia University with the modern radio frequency communication systems naturally starts with the basic principles of wireless transmission and reception and the classical analog methods of modulation and demodulation. After building the basic knowledge, the moment of consideration of digital communication systems is reached, which in their generalized form follow the transmission chain presented in Fig. 1 [1].

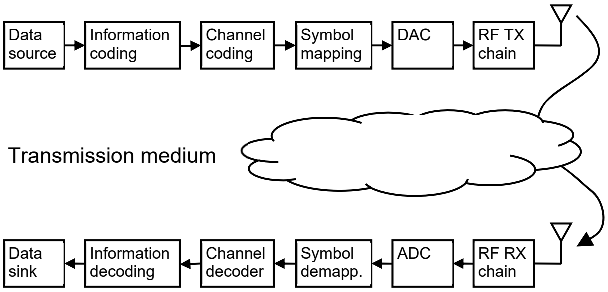


Fig.1. Generalized view of a digital communication system.

It is precisely the way in which the individual blocks are implemented in the thus presented digital communication system, which distinguishes the various concepts and architectures. If until a few years ago almost all information coding, channel coding and symbol mapping functions were implemented either by specialized hardware or by programmable digital signal processors (the latter adding a greater degree of flexibility when introducing new standards), now all the blocks in the presented diagram except the analog-to-digital and digital-to-analog converters and the RF circuits are implemented by software. This marks the transition from the hardware and mixed (with DSP methods) approach in building RF digital transmission systems to the much more versatile and flexible approach based almost entirely on software, called software-defined radio (SDR). This development naturally affected the way in which the laboratory exercises in the disciplines related to communication technologies are implemented.

The hardware/DSP approach

In earlier works of the author [2] a system was proposed and implemented as hardware, allowing the implementation of digital QAM-modulation within a digital communication system for radio frequency data transmission. The ADSP-21061 floating-point digital signal processor was used as the basis of the project, as well as a board specially developed for the purpose, including an ADC, a DAC and the digital modulator IC, enabling direct QAM-modulation. As can be seen from the generalized diagram of the system in Fig. 2, also in use for student training, the main modules in the signal chain are delegated to the software executed by the digital signal processor.

The development and research of the software modules was carried out in the Analog Devices IDE development environment supporting the signal processor in question, with the executable code being loaded into the system via a serial channel from a PC.

Our several years of practice with this somewhat outdated system have shown both the advantages and disadvantages of this approach to teaching digital communications to students, and in particular:

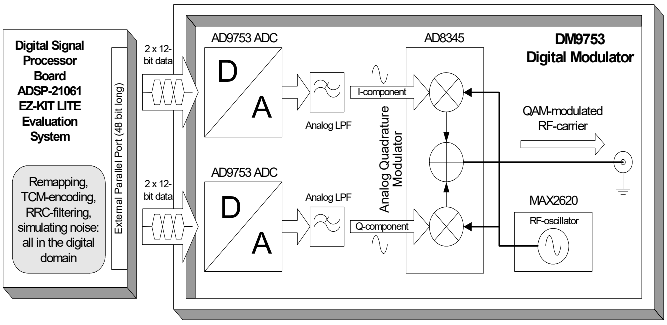


Fig.2. Digital RF TX-chain implemented through DSP and a modulator-chip.

Advantages of the hardware/DSP approach

- The use of a digital signal processor with sufficient power allows for a more efficient implementation of the encoding/decoding, filtering, mapping and synchronization algorithms between the transmit and receive side than if this were done entirely in software on a PC. The reason is that a DSP practically occupies an intermediate position between specialized hardware and a pure software approach;
- At the same time, such a DSP-based system retains its flexibility, in the sense that when standards change and new modulation schemes emerge, new algorithms can be implemented without hardware modification;
- Direct work with hardware and DSP-algorithms allows some of the students to have more direct contact with communication hardware. They are not isolated by an intermediate layer of software and so they learn to use the capabilities of the underlying digital transmission systems more effectively.

Disadvantages of the hardware/DSP approach

- Programming the system requires students to have a thorough knowledge of the hardware architecture of the digital signal processor itself, as well as the converter/modulator subsystem. In general, this makes it necessary to consider teaching the system within several exercises and somewhat distracts from the essence of the topic itself, namely the practical application of the methods of coding, mapping and the modulation itself;

- In-depth knowledge of the specific hardware/DSP circuitry may be redundant, as individual integrated circuits are subject to rapid obsolescence and are replaced by more modern and more powerful ones;
- Depending on the DSP-platform and GUI used, the system can be limited in terms of interactivity, not allowing adjustments of the algorithms on the fly and students observing the effect of adjustments on the parameters of the used modulation scheme during the execution of the DSP-code. This is the case with the system, proposed in [2] and used for many years in the DSP-Lab. Surely more sophisticated DSP-based systems are available, but they are usually proprietary and tied to specific company products.

ADALM-Pluto – the SDR-approach

As advertised by the manufacturer Analog Devices itself, the ADALM-Pluto is a self-contained RF learning module [3].

This was the reason why we decided on this SDR platform when choosing a suitable modern module for teaching the basic digital RF functionality. As can be seen from Fig. 3, representing the block diagram of the SDR-module, it consists of an AD9361/AD9363 transceiver with an operating range between 325 MHz and 3800 MHz and a Xilinx Zinq FPGA, providing the interface between the host machine and the Linux kernel of the SDR system, through which ultimately the data is fed to/from the transceiver for/after frontend RF-processing.

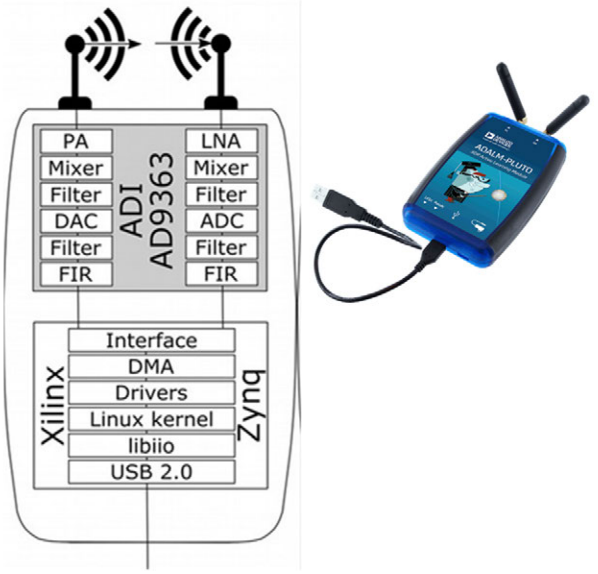


Fig.3. ADALM-Pluto block diagram and device view.

The FPGA chip of course offers significantly greater possibilities than a simple interface, a virtual TCP/IP port and an USB connection. It can perform decimation, interpolation, digital filtering, direct digital synthesis and many more useful control and data flow functions. They all are supported by the Industrial I/O Interface (IIO) library, providing an API for direct hardware control through the C++ and python languages [4].

The radio frequency transceiver, which internal structure is shown in Fig. 4, represents a full-duplex IQ-transmitter, IQ-receiver, together with their corresponding 12-bit ADCs and DACs, as well as three fractional-N synthesizers. In practice, this integrated circuit duplicates and greatly expands the functions of the specialized circuit board that was connected to the digital signal processor in the first approach described in this article. Applying a simple hack published on many ADALM-Pluto related forums, we were able to extend the range of the transceiver (AD9363 variant) to 70 - 6000 MHz (with no guarantee of the circuit parameters given for the narrower frequency range).

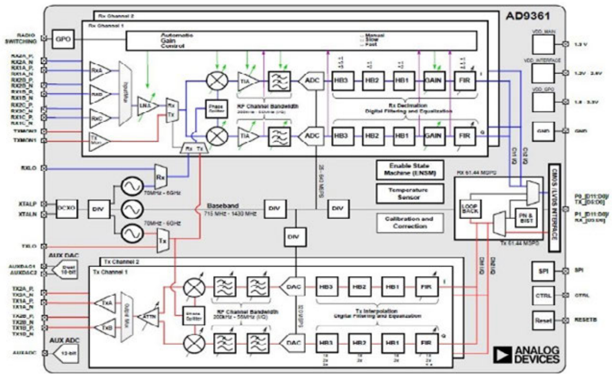


Fig.4. The internal structure of the AD9361 agile transceiver, the RF front-end of ADALM-Pluto.

The architecture of the AD9361 transceiver, just like the architecture of the IQ-modulator AD8345 from the specialized board in the DSP-approach, is of the Zero IF type (Fig. 5) [1]. This concept has been widely adopted in recent years for RF front-end implementations, as it eliminates the need to filter the intermediate frequency spectrum and greatly simplifies analog signal processing. In addition, the bandwidth requirements (sample rate) of ADCs and DACs are significantly lower because they convert baseband signals. In this part both approaches to teaching modern RF communication systems are comparable, following current trends.

Interfacing the ADALM-Pluto SDR

Teaching RF to students with the Analog Devices SDR-module of course requires a suitable interface and development environment to construct and test the various algorithms for DSP processing, coding, modulation and other baseband functions in a digital RF-chain.

To our great surprise, it turned out that the manufacturer of ADALM-Pluto does not offer any adequate interface allowing communication with the SDR-module, apart from the IIO Oscilloscope software limited to simple visualization capabilities of the received signals [5]. For the purposes of training in RF-communication technologies the latter turned out to be completely inadequate. The two main methods for interfacing and evaluation of the capabilities of the SDR-module rely therefore entirely on third-party applications.

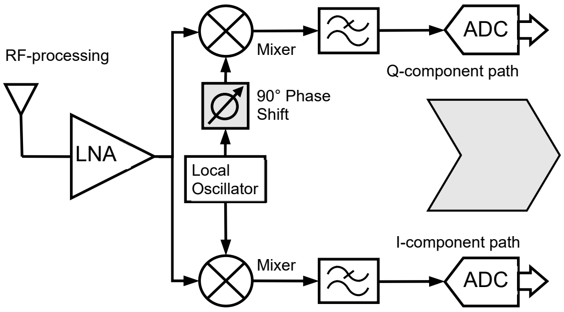


Fig.5. Zero IF architecture of the RF front-end of ADALM-Pluto (receiver side)

In our choice of development system, we were limited to MATLAB Simulink and GNU Radio. We rejected the first option due to the need for a license, as well as the fact that MATLAB is a more versatile product, not primarily focused on RF technologies. This also determined our decision to focus on the GNU Radio open-source software platform specifically dedicated to RF-technologies.

Teaching and evaluating software-defined radio with GNU Radio

More than a decade old, the free open-source toolkit targeting radio communications is equally well suited for hobbyists, professionals and academia alike. The project is an API built mainly on the languages C++ and python, and thanks to the availability of a graphical environment called GNU Radio Companion (GRC) [6], mastering programming skills is not a mandatory condition for using the many useful functions and blocks related to baseband- and RF-processing.

In addition to this, the GNU Radio Companion environment also offers many visualization tools, replacing the standard laboratory oscilloscopes (with time sinks) and spectrum analyzers (with frequency sinks), as well as very specialized ones, allowing for example the observation of constellation diagrams of QPSK- and QAM-modulated signals. The environment also allows simulation of signal imperfections and radio channel interference and offers many other useful tools for every radio engineer, emulating real-world environments. In practice, the GNU Radio Companion can be used for training in radio frequency communication technologies without the need for additional hardware. However, the combination of the graphical environment with SDR-modules such as ADALM-Pluto allows testing the simulated algorithms literally in real conditions.

All this clearly determined our choice for the second SDR-based approach for teaching radio frequency theory and practice to students. Of course, as with the DSP-based approach, the ADALM-Pluto - GNU Radio Companion combination has its own strengths and weaknesses.

Advantages of the GRC-SDR approach

- Software-defined radios allow full flexibility in terms of RF-based standards being studied and researched. Since all baseband processing blocks are software-defined, new emerging standards can be accommodated very easily. The radio frequency part (front-end) of the ADALM-Pluto module allows wide limits of setting the local oscillator frequency, as well as other parameters (gain, AGC-modes, DC-corrections in BB and RF). All this makes the SDR-module a very versatile platform;
- The ADALM-Pluto platform is supporting full duplex mode, which means that the functionality of the algorithms can be checked at the radio transmission level without the need for an additional SDR-device. In practice, a fully functional student workplace consists only of a PC with GNU Radio Companion installed on it and one unit of the ADALM-Pluto RF-teaching platform;
- There is no need for additional instruments and tools to perform practical exercises and measurements, as the GNU Radio Companion includes all necessary visualization tools as part of the GUI;
- The GNU Radio Companion is specifically designed to bring software-defined radio into more and more applications. Because of this, the

interface has an extensive set of blocks for various signal processing tasks, and the system is open-source and therefore constantly updated with new ones;

- The graphical approach of visualizing both relatively simple entry-level and complex BB-processing algorithms through so-called GRC-flowgraphs is perceived by students and beginners much easier than learning about specialized hardware or writing and debugging of program code;
- The GNU Radio Companion allows the simulation of noise, delays, fading, clock offsets, and other transmission chain imperfections, which is useful both for familiarizing students with the specifics and challenges of radio transmission and before and during testing new algorithms on real SDR-hardware, that is in this case the ADALM-Pluto.

Disadvantages of the GRC-SDR approach

- Without prior familiarity with the fundamentals of digital signal processing and baseband processing for RF-applications, there is a risk that students will be misled by the simplicity of creating even complex RF-processing algorithms in thinking that they are gaining a deep understanding of the subject, in fact acquiring superficial knowledge;
- The creation and testing of more serious GRC-flowgraphs cannot go completely without knowledge of programming. This is for example the case when using embedded python blocks. Of course, such an argument is not relevant in teaching the basics of RF-communications;
- The creation of powerful embedded systems for digital radio communications, as well as those for other radio frequency applications (radar), implies that young specialists are able to design such from scratch using modern ICs, embedded processors and programmable logic. Such systems also fall into the class of software-defined radio (algorithms implemented in software on embedded processing hardware) but cannot be presented to learners through learning platforms like ADALM-Pluto, masking to a great extend the low-level functionality of the building blocks.

A practical example of using GNU Radio Companion with ADALM-Pluto in the lab

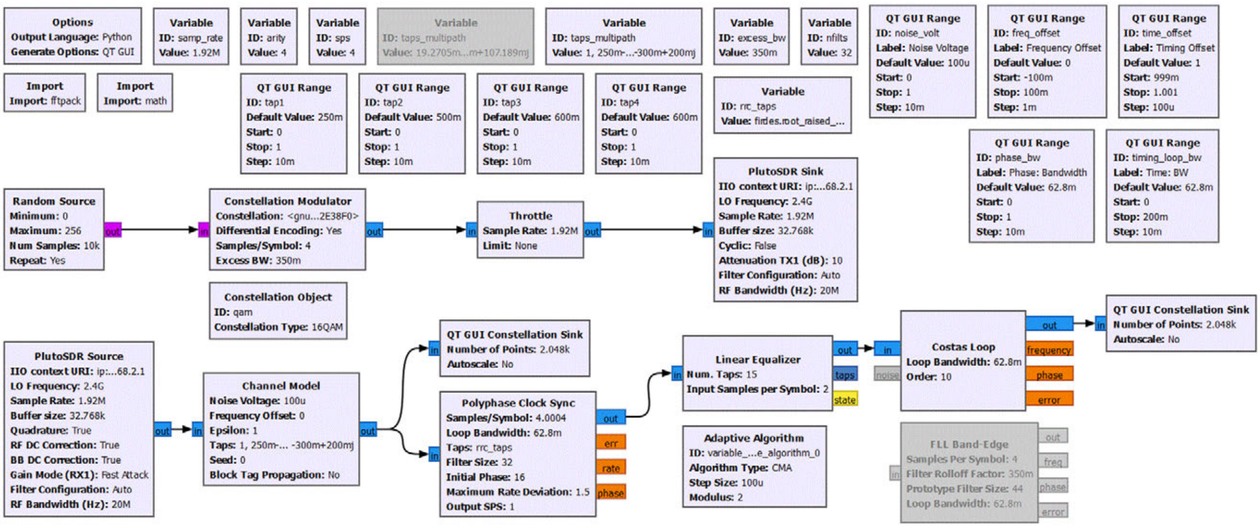
To demonstrate the great possibilities provided by the GNU Radio Companion environment in combination with the ADALM-Pluto SDR platform, here we present one of the exercises that combines elements of baseband coding and mapping (in this case QAM-16) with a simulation of a noisy transmission channel introducing time and frequency offset, visualization of the constellation diagram before and after receiver-introduced corrections, and the actual on-air transmission and reception of the signal between the ADALM-Pluto transmit and receive antennas. Such a combination of introducing software algorithms, simulation of transmission medium imperfections and real-time SDR code performance testing with a real RF-device is truly unique and cannot in any way be achieved using dedicated RF-processing hardware, even a DSP-based one. The approach is extremely useful for teaching students, as it allows them to see practically the entire process of designing and testing the RF-chain within a single flowgraph.

Baseband processing algorithm and signal synthesis

In Fig. 6, representing a flowgraph of a QAM-16 modulator, there are several blocks connected - a source of a pseudorandom signal (Random Source), a universal modulator (Constellation Modulator) with the setting of the constellation diagram (Constellation Object). The complex IQ-signal is sent to the SDR-module (PlutoSDR Sink). The 2.4 GHz signal emitted by the ADALM-Pluto transmitter antenna is received by the receiving antenna and via the PlutoSDR Source block it is fed back into the flowgraph for further processing.

Simulation

The received at RF front-end of the real physical device and demodulated to BB complex IQ-signal did not suffer significant degradation due to the short transmission path between the transmitting and receiving antenna. Therefore, to simulate a real-world environment, it is fed to the GNU Channel Model block, which introduces additional noise, frequency offset and time offset. The specific values of these imperfections are set by means of GUI-controls, which are also defined in the flowgraph and are part of the graphical interface after its execution is started. They are all of the QT GUI Range type, differing by their ID (noise_voltage, freq_offset, time_offset) [6]. With such controls, students are given the opportunity to experiment by observing the effect of different levels of RF-signal degradation when passing through the transmission medium. By means of a suitable filter (via the Taps-coefficients in the channel model) multipath signal propagation and distortion can also be simulated. After the impairments are introduced the IQ-signal with the simulated noise and with introduced


Options
Output Language: Python
Generate Options: QT GUI

Variable
ID: samp_rate
Value: 1.92M

Variable
ID: arity
Value: 4

Variable
ID: sps
Value: 4

Variable
ID: taps_multipath
Value: 19.2705m...m+107.189mj

Variable
ID: taps_multipath
Value: 1, 250m-...-300m+200mj

Variable
ID: excess_bw
Value: 350m

Variable
ID: nfilts
Value: 32

QT GUI Range
ID: noise_volt
Label: Noise Voltage
Default Value: 100u
Start: 0
Stop: 1
Step: 10m

QT GUI Range
ID: freq_offset
Label: Frequency Offset
Default Value: 0
Start: -100m
Stop: 100m
Step: 1m

QT GUI Range
ID: time_offset
Label: Timing Offset
Default Value: 1
Start: 999m
Stop: 1.001
Step: 100u

Import
Import: fftpack

Import
Import: math

QT GUI Range
ID: tap1
Default Value: 250m
Start: 0
Stop: 1
Step: 10m

QT GUI Range
ID: tap2
Default Value: 500m
Start: 0
Stop: 1
Step: 10m

QT GUI Range
ID: tap3
Default Value: 600m
Start: 0
Stop: 1
Step: 10m

QT GUI Range
ID: tap4
Default Value: 600m
Start: 0
Stop: 1
Step: 10m

Variable
ID: rrc_taps
Value: firdes.root_raised_...

QT GUI Range
ID: phase_bw
Label: Phase: Bandwidth
Default Value: 62.8m
Start: 0
Stop: 1
Step: 10m

QT GUI Range
ID: timing_loop_bw
Label: Time: BW
Default Value: 62.8m
Start: 0
Stop: 200m
Step: 10m

Random Source
Minimum: 0
Maximum: 256
Num Samples: 10k
Repeat: Yes

Constellation Modulator
Constellation: <gnu...2E38F0>
Differential Encoding: Yes
Samples/Symbol: 4
Excess BW: 350m

Throttle
Sample Rate: 1.92M
Limit: None

PlutoSDR Sink
IIO context URI: ip:...68.2.1
LO Frequency: 2.4G
Sample Rate: 1.92M
Buffer size: 32.768k
Cyclic: False
Attenuation TX1 (dB): 10
Filter Configuration: Auto
RF Bandwidth (Hz): 20M

Constellation Object
ID: qam
Constellation Type: 16QAM

QT GUI Constellation Sink
Number of Points: 2.048k
Autoscale: No

PlutoSDR Source
IIO context URI: ip:...68.2.1
LO Frequency: 2.4G
Sample Rate: 1.92M
Buffer size: 32.768k
Quadrature: True
RF DC Correction: True
BB DC Correction: True
Gain Mode (RX1): Fast Attack
Filter Configuration: Auto
RF Bandwidth (Hz): 20M

Channel Model
Noise Voltage: 100u
Frequency Offset: 0
Epsilon: 1
Taps: 1, 250m-... -300m+200mj
Seed: 0
Block Tag Propagation: No

QT GUI Constellation Sink
Number of Points: 2.048k
Autoscale: No

Polyphase Clock Sync
Samples/Symbol: 4.0004
Loop Bandwidth: 62.8m
Taps: rrc_taps
Filter Size: 32
Initial Phase: 16
Maximum Rate Deviation: 1.5
Output SPS: 1

Linear Equalizer
Num. Taps: 15
Input Samples per Symbol: 2

Adaptive Algorithm
ID: variable_...e_algorithm_0
Algorithm Type: CMA
Step Size: 100u
Modulus: 2

Costas Loop
Loop Bandwidth: 62.8m
Order: 10

FLL Band-Edge
Samples Per Symbol: 4
Filter Rolloff Factor: 350m
Prototype Filter Size: 44
Loop Bandwidth: 62.8m

Fig.6. Example flowgraph for evaluation of QAM-16 modulation in GNU Radio Companion.

conditions for desynchronization and ISI is fed to the next blocks for correction and visualization.

Receiver side correction algorithms and visualization of the constellation diagram before and after them

Signal disturbances introduced by the channel model are immediately reflected in the received constellation diagram. This is visualized using the first QT GUI Constellation Sink block in the flowgraph. Several blocks follow, which aim to compensate for the effects introduced by the communication channel and maximally restore the original transmitted constellation diagram.

The first of them is the Polyphase Clock Sync block. It functions based on 32 filters introducing an increasing phase shift and determining the presence of a zero sample in the derivative of the sinc-shaped (through RRC-filtering) pulses. So, the necessary phase shift of the clock signal of the receiver is determined, which aims sampling the received pulse at the correct point in time and achieving zero intersymbol interference. In this way a time synchronisation between transmitter und receiver is achieved.

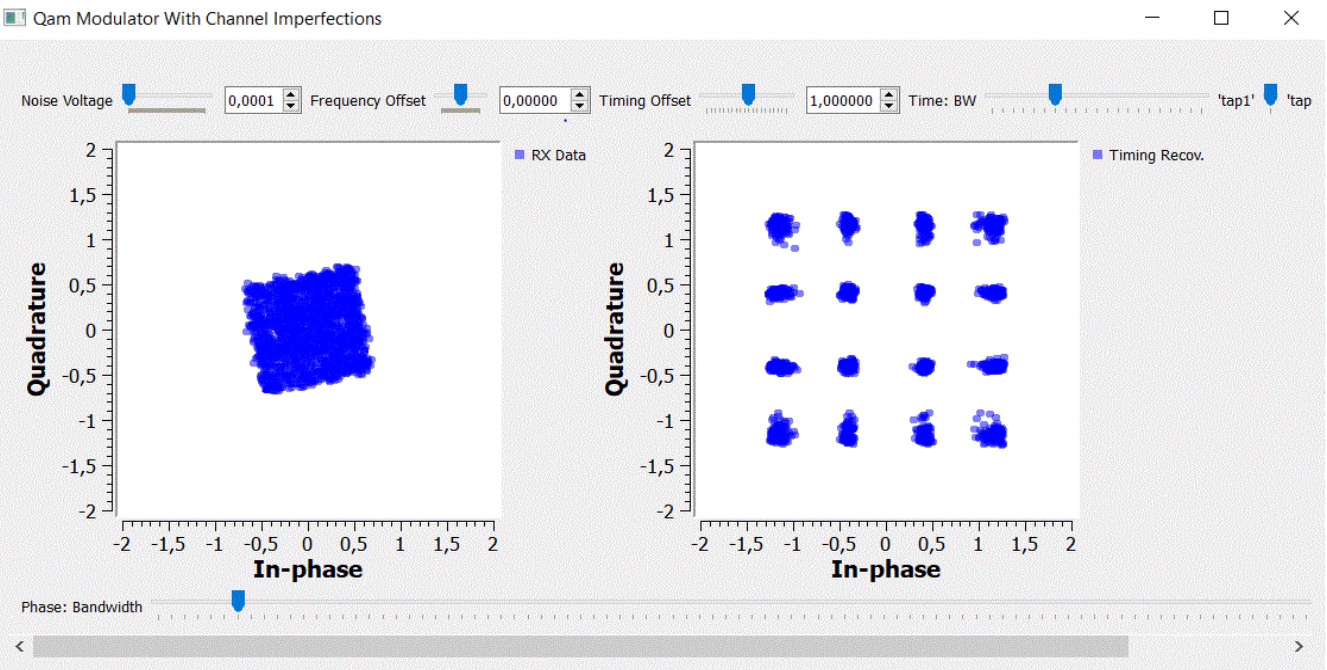
After the Polyphase Clock Sync the signal at the receiver side is passed to the Linear Equalizer block, which eliminates the effects of the simulated multipath signal propagation by applying an adaptive algorithm, described by the Adaptive Algorithm object, residing below the equalizer.

After the phase alignment of the receiver clock and the correction of the frequency response of the received signal by means of the equalizer, the last signal recovery block follows, namely the correction of the phase and above all the frequency offsets introduced by the channel (often a direct consequence of the Doppler effect, when transmitters or receivers are moving). It is implemented by the Costas Loop block (a classic PLL optimized to work with various phase modulation schemes, like BPSK, QPSK and xPSK and with slight modifications also with QAM [7]).

The students can alternatively experiment with the FLL Band-Edge block, which is an algorithmic implementation of Band-Edge Frequency Locked Loop. The latter is a variant of the Maximum Likelihood Frequency Detector, generating an error signal and aiming for maximum overlap of the spectrum of the received signal z (multiplied at the transmitter side by a square-root Nyquist shaping filter) and the frequency response of the matched filter at the receiver side (another square-root Nyquist filter). This is done by trying to maximize the energy of the complex signal surviving both spectrums overlap. When the maximum is reached, it means that also the correct carrier frequency offset (CFO) has been estimated and applied. The frequency error detector determines the maximum by iteratively (on the sample-by-sample basis) looking for a zero in the derivative of the signal energy and producing the corresponding in-phase error signal in the feedback loop as shown in (1).

$$(1)\quad e_D[n] = \frac{d}{dF_\Delta}|z(nT_S)|^2 \sim \{z(nT_S)\cdot\{\dot{z}(nT_S)\}^*\}_I$$

For a practical implementation it is more efficient instead of computing the derivative of z in real time to use the pre-computed derivative of the matched filter frequency response (convolution of the impulse responses of the derivative-computing filter h_D and the matched filter p as shown in (2)), which has a non-zero value in the two boundary frequency bands (excess



Qam Modulator With Channel Imperfections

Noise Voltage 0,0001 Frequency Offset 0,00000 Timing Offset 1,000000

Fig.7. QAM-16 constellation diagrams before and after timing and clock offset compensation and equalizing.

bandwidth). This also gave the name of the Band-Edge FLL-algorithm [8].

(2) $\dot{z}(nT_S) = z(nT_S) * h_D(nT_S) = \tilde{x}(nT_S) * \{p(-nT_S) * h_D(nT_S)\}$

We intentionally mention such details about the Band-Edge FLL-algorithm to stress the importance of understanding the underlying signal processing math behind a simple looking GRC-block.

Finally, after applying all the corrections, the IQ-signal processed in the described way is fed to the second QT GUI Constellation Sink, where the reconstructed QAM-16 constellation diagram can be observed. The visualization of the two constellations, before and after the correction on the receiving side, is presented in Fig. 7. The success of the restoration process is determined by multiple settings of the parameters of the corresponding correction blocks in the flowgraph (filter coefficients, bandwidths of the time, phase and frequency loops, roll-off factors). By interactively experimenting with the values of these parameters, not only optimization of the correction algorithms is possible, but also obtaining valuable practical experience in these quite non-trivial issues of modern RF communication systems.

Final thoughts on the GNU Radio Companion – SDR-approach

A cursory glance at the complexity of the flowgraph in Fig. 6, as well as the inclusion of heterogeneous algorithms for digital signal processing (FIR-filtering, feedback theory) in it shows that the SDR approach in studying RF-communications is practically impossible without previously acquired in-depth knowledge in the field. Table 1 summarizes the necessary prior knowledge regarding the individual blocks of the example flowgraph, presented in Fig. 6.

Table 1

QAM-16 example GRC blocks prerequisites

Random Source	Pseudorandom Number Generator theory, Linear-Feedback Shift Registers
Constellation Modulator	Digital Modulation principles
Throttle	Sampling theory, Data rate synchronization
Pluto SDR Sink	SDR, Zero IF modulation, LO theory, RF-bandwidth filtering, TCP/IP
Pluto SDR Source	SDR, Zero IF modulation, LO theory, RF and BB DC-corrections, RF-bandwidth filtering, TCP/IP, Automatic Gain Control
Channel Model	Transmission channel theory, wireless propagation, noise theory, FIR-filters
Polyphase Clock Sync	Symbol sampling, ISI, timing recovery, pulse shaping, Nyquist filtering, RRC-filters
Linear Equalizer	Multipath signal propagation, equalizer theory, channel frequency response, FIR-filters, adaptive filters, adaptive equalizers

Costas Loop	Automatic Control systems, feedback theory, PLL, VCO
FLL Band-Edge	Automatic Control systems, Digital Phase-Locked Loop, matched filtering
QT GUI Constellation Sink	Digital Modulation principles, Visualization theory

Conclusion

The use of SDR-platforms such as ADALM-Pluto in combination with powerful APIs such as GNU Radio, especially in its form with a graphical interface (GRC), opens up enormous possibilities for teaching the diverse and sometimes too complex algorithms used in modern RF-chains. To some extent, this creates for students the feeling of acquiring knowledge on complex matter almost like in a game. Our experience of using the classic approach to acquire practical knowledge and skills through a specialized DSP-based hardware and dedicated radio front-end, compared to the SDR-approach intended to replace it, showed one thing unambiguously. The idea of completely replacing the hands-on exercises implementing low-level RF-systems (DSP-based) with the significantly easier to both implement and teach software-defined radio modules is not justified. An optimal effect in terms of acquired knowledge is achieved when exercises with SDR build on the basic exercises of applying BB-algorithms in the RF-chain on specialized hardware and/or digital signal processors. And if the latter systems seem somewhat old-fashioned in the context of modern methods of designing radio-frequency communication equipment [9], they are indispensable in acquiring the necessary thorough basic knowledge, needed also for successful implementation of software-defined radio. This is also the way for students to gain a real insight into the capabilities of modern SDR-based RF modules.

Our observations presented so far are based on the difficulties encountered by the students in carrying out the exercises in both considered approaches. The students were successful in both cases, so the conclusions drawn are based on qualitative rather than quantitative (by grading) observations of their work in the lab and shared feedback.

Acknowledgements

We express our gratitude to the Department of "Radiophysics and Electronics" at the Faculty of Physics of Sofia University "St. Kliment Ohridski" for the purchase of the ADALM-Pluto SDR-module, which made possible its application in the Communication Technologies workshop for the "Communications and Physical Electronics" bachelor degree program and helped gain valuable knowledge related to the application of software-defined radio in teaching.

REFERENCES

[1] Collins T., R. Getz, D. Pu, A. Wyglinski, Software-defined radio for engineers., Mobile Communication Series, Artech House, 2018, pp. 5-6, p.12.

[2] Vladkov E. E., A general concept and implementation of a DSP-based QAM digital modulator, “Electrotechnica & Elektronica E+E”, Vol. 50. No 1-2/2015, pp.18-25.

[3] Software-Defined Radio Active Learning Module, Analog Dialogue, https://www.analog.com/en/design-center/evaluation-hardware-and-software/evaluation-boards-kits/adalm-pluto.html#eb-overview.

[4] Ranson Richard G., A new golden age of radio. Development tools for Software Defined Radio., Radio System Design Ltd., Continuing Education Institute, 2019.

[5] IIO Oscilloscope Analog Devices Wiki Page., https://wiki.analog.com/resources/tools-software/linux-software/iio_oscilloscope.

[6] GNU Radio Companion Tutorials, https://wiki.gnuradio.org/index.php?title=Tutorials.

[7] Hagemann E., The Costas Loop – an introduction, 2001.

[8] Chaudhari Q., Wireless communications from the ground up. An SDR Perspective., wirelesspi.com, 2018, pp. 396-409.

[9] Qutab ud din M., A. Waqas, T. Naeem, SDRs and their application in modern communication, Tampere University of Technology, Finland, Academia.edu.

***Assoc. Prof. Dr. Emil E. Vladkov** is currently a lecturer at Sofia University, Faculty of Physics, Dept. of Radiophysics and Electronics, he graduated from Sofia University in 1996 and became a PhD in 2001. His scientific interests cover the area of communication networks and protocols, Wireless Sensor Networks (WSN) and Digital Signal Processing Algorithms and hardware implementations.*

tel.: 0888 099199 e-mail: evladkov@phys.uni-sofia.bg

***Assoc. Prof. Dr. Stanimir T. Kolev** is an associate professor at Sofia University, where he also obtained his PhD in 2007. His research interests are mainly in the field of plasma and gas discharge physics. He is also involved in a number of projects related to different applications like plasma-assisted dissociation of CO2, negative ions sources for nuclear fusion reactors, plasma sources for absorption spectroscopy, plasma assisted waste treatment and others.*

tel.: 0887 631420 e-mail: skolev@phys.uni-sofia.bg

Received on: 12.08.2023