IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received June 23, 2017, accepted July 26, 2017, date of publication August 4, 2017, date of current version August 29, 2017.

Digital Object Identifier 10.1109/ACCESS.2017.2736165

Design of a Remote Signal Processing Student Lab

ALBERTO ALBIOL[1], **ALBERTO CORBI**[2], **AND DANIEL BURGOS**[2]
[1]Department of Communications, Universitat Politècnica de València, 46022 Valencia, Spain
[2]Research Institute for Innovation and Technology in Education, Universidad Internacional de La Rioja, 26006 Logroño, Spain

Corresponding author: Alberto Corbi (alberto.corbi@unir.net)

This work was supported by the Universidad Internacional de la Rioja through the Research Institute for Innovation and Technology in Education.

ABSTRACT We describe our experience of introducing digital signal processing (DSP) concepts via a software-defined radio project using a very inexpensive TV USB capture dongle. Through a series of weekly lab exercises, the students learned and applied DSP concepts to design a completely digital FM receiver. The proposed lab experience introduced concepts, such as sampling, IQ signal representation, sample rate conversion, filter design, filter delays, and more, all with an attractive learn-by-doing approach. The first offering of this course initially took place in Fall 2014 and has been successfully offered and repeated with growing success ever since. Our experience can serve as a proof of concept of the possibility of carrying out, in a massive open online course-like fashion, certain engineering labs that require inexpensive and readily available hardware components.

INDEX TERMS Digital signal processing (DSP), software-defined radio (SDR), GNU radio, engineering education, massive open online course (MOOC).

I. INTRODUCTION

The massive introduction of personal computers in the nineties enabled a revolution in the way that most undergraduate digital signal processing (DSP) courses were taught using classical books such as [23] or [24]. Software packages such as Matlab/Simulink [21], GNU Octave [22] and LabView [32] became very popular to illustrate many of the concepts that were taught during the DSP courses and specific packages for education were also developed [4], especially in the field of digital audio processing. The direct consequence was a better balance between theory and practice, which is essential for any modern undergraduate course [11].

However, in our DSP teaching experience, we have found that simulation activities using Matlab or similar software solutions are still not inspiring enough for many students. Similar conclusions have also been drawn by other colleagues who are teaching in this field [12]. Based on our background, we thought that it might be more inspiring for the students if they could capture and process their own signals in real time. However, real-time DSP may require specialized hardware such as DSP boards [14] and field-programmable gate arrays (FPGAs) which are usually not affordable for most students [27]. As an alternative to this problem, some real-time DSP courses try to bring this kind of hardware to the students by using web applications that remotely access the same components, which are shared among several students [13], [26]. We also found that a common problem of many DSP boards is that they have to be programmed in assembly code, which might entail extra difficulties for an introductory course. This is a frustrating experience for many students, where a slow learning curve hinders the attractiveness of the DSP field.

Recently, we found an extremely inexpensive solution for performing real-time DSP based on a commercial DVB-T USB capture dongle that is equipped with the RTL2832U chipset. This dongle can be bought for just $20 on many web sites and allows any student to experiment with many different DSP projects from their own homes. The merit of this revolutionary idea stems from the combined efforts of Antti Palosaari and Eric Fry [25], who found a way to hack the aforementioned dongle, bypassing the I/Q signal and redirecting it to the host computer. The immediate consequence is that this device can be turned into a wide-band sampling module with hundreds of possible applications [18].

Meanwhile, the development of the Internet and modern e-learning learning management systems (LMS) such as Moodle, Sakai, Blackboard, Canvas, etc., and associated industry standards (i.e., IMS Global) has dramatically improved the quality and possibilities of distance learning [5]. Even though there are other platforms that are better suited for the job (i.e., edX, Coursera, etc.), these LMS platforms are capable of handling huge enrollment rates and are more than ready to provide support for massive open online course (MOOC) sites. Besides, the learning tool interoperability (LTI) specification (described in Section VI and in [28]) allows the cooperation of the aforementioned LMSs with remote web applications that deliver specific and richer content for related assignments and enables the secure submission of the achieved scores to each student's own gradebook. By combining affordable lab materials and hardware components like the ones presented above, which are detailed further in the rest of this paper, real engineering labs can be made more plausible and tangible in both face-to-face and distant (even massive) learning scenarios. MOOC campuses and associated large enrollment rate courses are a growing reality in both continuing/further and conventional/classroom higher education [6].

In Section II, we present the main topics of a DSP course that is based on these ideas. In our implementation, the students have to build a full-featured broadcast FM receiver with very accessible components that are listed in Section III. All of the DSP concepts that are explained and taught in the course are presented in Section IV and Section V. Finally, we discuss the specific academic impact of this course in Section VI, highlighting its positive acceptance by the majority of the enrolled students during four consecutive semesters (from 2013 to 2017) and its possible deployment in non face-to-face teaching.

II. COURSE DESCRIPTION

Our DSP course is conducted in the School of Telecommunications Engineering at Universitat Politècnica de Valencia (Spain). It is an introductory course that is aimed at providing undergraduate students with the fundamental concepts of DSP. Previously to our subject, the students have taken an introductory course on analog and digital linear systems and another one that introduces the elemental signal modulation techniques. The academic program follows with two more specific subjects that are focused on digital audio and image processing.

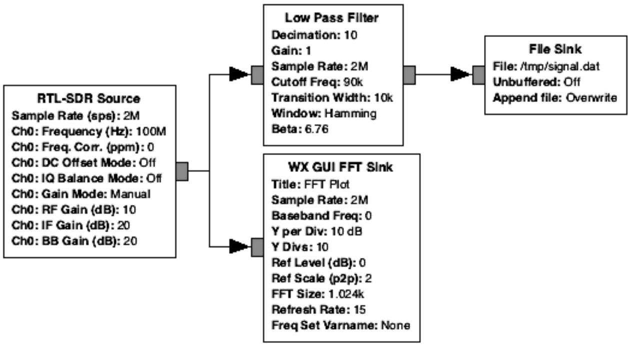
The main topics covered in the DSP course are:

- Sampling and reconstruction of baseband and bandpass analog signals.
- Decimation and interpolation.
- Multi-rate digital signal processing.
- Discrete Fourier Transform and FFT.
- Methods for efficient FIR filtering using FFT.
- Spectral analysis.
- Practical digital filter design.

A fundamental part of the course is an application design laboratory where the students have to apply the above theory concepts in lab projects. Among other projects, they have to design a fully digital FM receiver that works in real time using DSP. To achieve this goal the students only need basic programming skills in Python, GNU Radio [2], and Matlab.

III. REQUIRED SOFTWARE AND HARDWARE

Software-defined radio or SDR is basically a radio system where most of the components that had traditionally been implemented in hardware are now implemented in software using DSP [9]. The key advantage of SDR is *reconfiguration*, which makes possible a complete change of the radio system by means of a simple software update. This feature is essential in missions where hardware replacement is not possible (i.e., Mars exploration [15]).



RTL-SDR Source
Sample Rate (sps): 2M
Ch0: Frequency (Hz): 100M
Ch0: Freq. Corr. (ppm): 0
Ch0: DC Offset Mode: Off
Ch0: IQ Balance Mode: Off
Ch0: Gain Mode: Manual
Ch0: RF Gain (dB): 10
Ch0: IF Gain (dB): 20
Ch0: BB Gain (dB): 20

Low Pass Filter
Decimation: 10
Gain: 1
Sample Rate: 2M
Cutoff Freq: 90k
Transition Width: 10k
Window: Hamming
Beta: 6.76

WX GUI FFT Sink
Title: FFT Plot
Sample Rate: 2M
Baseband Freq: 0
Y per Div: 10 dB
Y Divs: 10
Ref Level (dB): 0
Ref Scale (p2p): 2
FFT Size: 1.024k
Refresh Rate: 15
Freq Set Varname: None

File Sink
File: /tmp/signal.dat
Unbuffered: Off
Append file: Overwrite

FIGURE 1. Simple graph created using GNU Radio Companion (GRC) that captures and saves a signal.

In our course, we use GNU Radio (https://www.gnuradio.org) as the main software development kit. GNU Radio is a free, open-source software that is distributed under the terms of the GNU General Public License that provides signal processing blocks to implement software radios in Python. It can be used with external radio frequency (RF) hardware or without hardware in a simulation-like environment. It is widely used in hobbyist, academic, and commercial environments to support both wireless communications research and real-world radio systems.

Although GNU Radio projects are written in Python, there is a much easier way to create radio systems using the GNU Radio Companion or GRC [29]. GRC is a graphical tool for creating signal flow graphs and generating flow-graph source code on top of GNU Radio and is a free alternative to other graphical packages such as Simulink or LabView. Figure 1 depicts a simple graph that is created with GRC where a RF signal is captured, down-sampled, and then saved in a file while the spectrum is visualized. An important advantage of GRC for an introductory course is that the students can focus on the signal processing concepts while not being distracted with the implementation details of low-level programming. GRC saves all graphs in XML format, so it is easy to parse and verify by third party software as well. We will take advantage of this feature in Section VI.

GNU Radio is commonly used with *universal software radio peripheral* (USRP) devices. The first USRP devices were built by Ettus Research, and, since then they have made many different revisions. Later versions are drastically different because they are built as a single board solution rather than a mother/daughterboard combination. Moreover, they use USB 3.X so that the bandwidth is greatly increased and full-duplex transmissions are possible. Although Ettus products can be ideal for research labs and universities, they are probably not affordable for the average student.

A less expensive alternative is provided by the HackRF project [16]. One of the disadvantages of HackRF is that it can only operate in half-duplex mode and the sample rate is limited to 20 Msps (compared to the 60 Msps of the Ettus URSP hardware). Also, the number of bits of the *analog to digital* (ADC) and *digital to analog* converters (DAC) is reduced to 8-bit per sample. Despite these inconveniences, HackRF is a very nice option for a hobbyist that can afford around $300. HackRF is a very versatile transceiver that can work in a wide spectrum with any type of modulation.

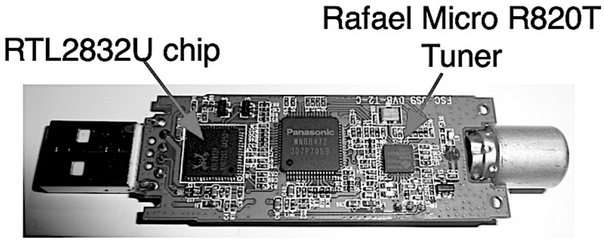


FIGURE 2. Main elements of a DVB-T USB dongle based on the RTL2832U.

However, in order to achieve the goals proposed in our course, the most convenient option is to use the consumer DVB-T USB dongle based on the RTL2832U chip shown in Figure 2. As stated in the introduction of this paper, this device can be bought for just $20. There are many manufacturers that build dongles of this kind, with the included turner being the key difference amongst them. Table 1 lists the most common tuners. The dongle outputs 8-bit I/Q-samples, although the highest theoretically possible sample rate is 3.2 MS/s. In practice, the highest rate achievable is 2 MS/s. Apparently, the bottleneck is the USB 2.X interface. It is important to emphasize that the tuner might limit the use of some dongles in some applications depending on the frequencies of interest. Nevertheless, this problem can be solved by using up- or down-converters in front of the dongle to capture signals in other band frequencies [18]. Finally, it should be noted that DVB-T dongles can only be used as receivers and cannot transmit any signal.

TABLE 1. Frequency ranges of the tuners that are usually packaged in the DVB-T dongles.

Tuner	Frequency range
Elonics E4000	52–2200 MHz with gap from 1100 to 1250 MHz
Rafael Micro R820T	24 – 1766 MHz
Fitipower FC0013	22 – 1100 MHz
Fitipower FC0012	22 – 948.6 MHz
FCI FC2580	146 – 308 and 438– 924 MHz (gap in between)

IV. DSP CONCEPTS IN A DIGITAL FM RECEIVER

The following sections explain in detail how to design and build a simple digital FM receiver using GNU Radio and all of the DSP topics that are involved.

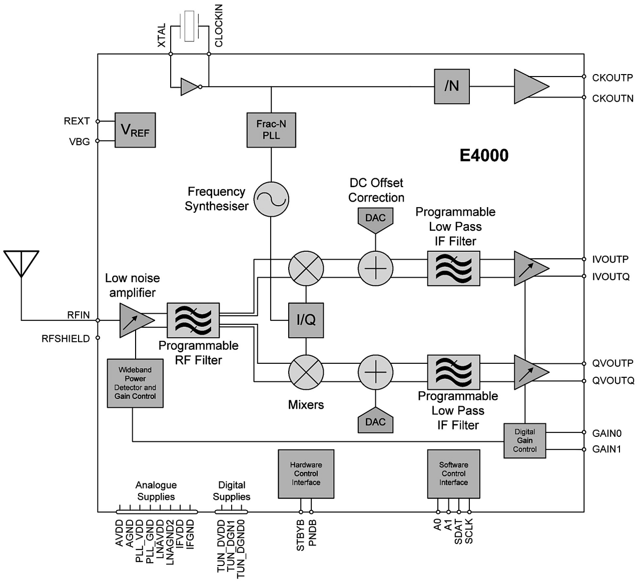
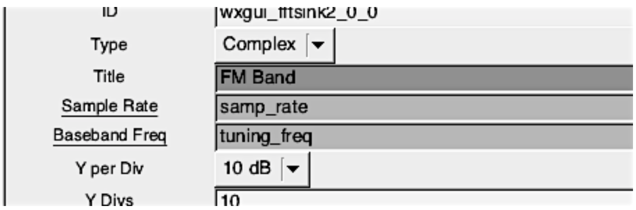


FIGURE 3. Diagram of the Elonics 4000 tuner.

A. SIGNAL CAPTURE

The students that attend our course are already familiar with the sampling of baseband signals. However, they must understand the sampling of bandpass signals [20] and the need of a tuner for *RF down-converting* the analog signal as shown in Figure 3. The output of the bandpass sampling is a complex discrete signal (*IQ samples*), which is something new for the students. Even though good references are found in the literature that try to demystify complex signals [19], our experience indicates that this particular topic requires a good level of abstraction. For instance, at the beginning of the course, many students do not understand that the sample rate depends on the bandwidth of the bandpass signal and that the frequency of the carrier is not relevant for this.

While the RF signal is being captured, we suggest to the students that they visualize the incoming signal using a FFT operation. However, since the signal has been down-converted, the students get often very surprised when the central frequency in the FFT diagram is zero. This common mistake allows us to explain the use of the baseband frequency parameter of the *FFT sink block* (Figure 4), which only matters for visualization and shifts the frequency axis to the desired carrier frequency.



ID	wxgui_fftsink2_0_0
Type	Complex ▾
Title	FM Band
Sample Rate	samp_rate
Baseband Freq	tuning_freq
Y per Div	10 dB ▾
Y Divs	10

FIGURE 4. Setting the baseband frequency is important in order to visualize the FFT spectrum of the complex signal in the corresponding analog frequencies.

Despite its name, the FFT sink block represents the *periodogram* [30] of the incoming signal (squared modulus of the FFT). In the course, we explain in detail that the periodogram is a *high variance estimator* of the spectrum of any signal. The live capture of the RF signal offers an excellent opportunity to visualize what a high variance estimator means. The first time that the students visualize a spectrum like the one shown in Figure 5-top, they assume that the signal fluctuations are caused by random additive noise that is carried by the signal. For a better visualization, the FFT sink block can add temporal averaging so that the variance of the spectrum estimation goes down as shown in Figure 5-bottom. This experience greatly helps to motivate the students to understand the problems related with spectral estimation that are presented during the preparation of the activity.

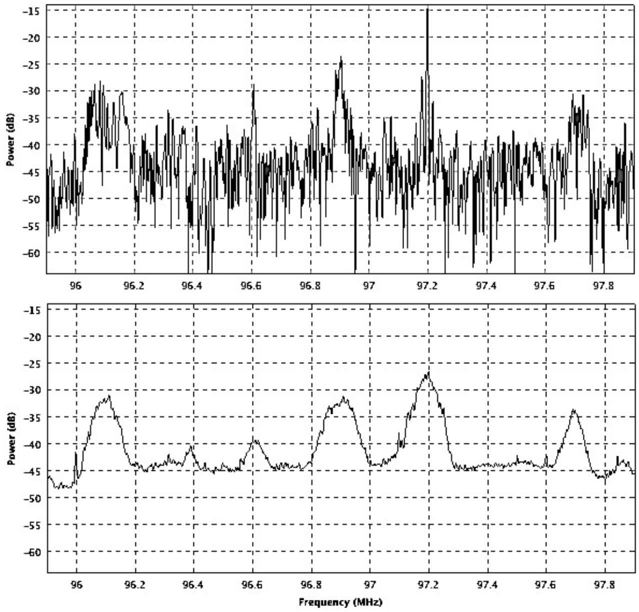
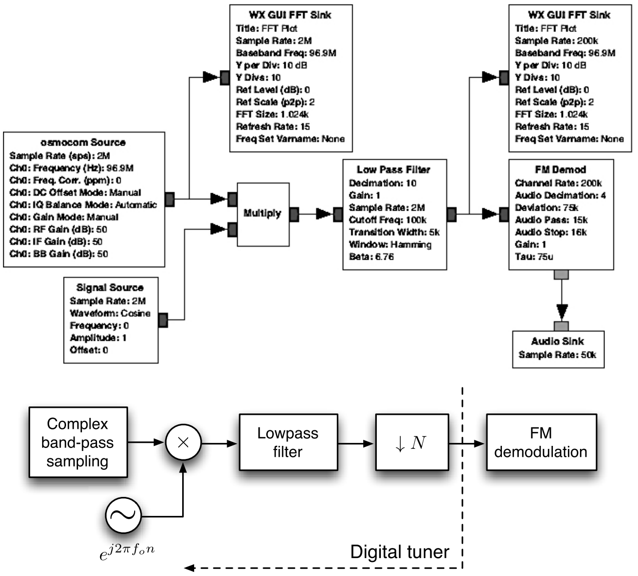


FIGURE 5. FTT of the estimated RF spectrum: without averaging (top) and with averaging (bottom).

B. DIGITAL TUNER

The bandwidth of a FM modulated broadcast signal is approximately 200 kHz. However, we encourage the students to create a prototype in which the RF signal is oversampled up to 2 Msps. With this sampling rate, the captured RF spectrum may contain several FM stations as shown in Figure 5-bottom and forces the students to insert a complex low pass digital filter to select one of the stations before FM demodulation. Initially, many students design unrealistic filters with extremely narrow transition bands in an attempt to obtain the best possible filter, but they ignore other consequences. The most obvious of these consequences is that their prototypes can not work in real time and the students learn by experience that engineering designs have plenty of *trade-offs*.

One of the advantages of using the 2 Msps sample rate is that it allows a very simple digital tuner to be designed as shown in Figure 6. However, there is a key difference compared to any analog tuner that the students are familiar with. Since we are working with digital signals, the spectrum shift produced by the multiplication of a complex exponential is cyclic. This is illustrated in Figure 7 where the spectrum after the multiplication block of the diagram of Figure 6 is represented. Although the circular frequency shift property is explained throughout the course, we found that once again the practice with signals in real time is an excellent educational booster.



osmocom Source
Sample Rate (sps): 2M
Ch0: Frequency (Hz): 96.9M
Ch0: Freq. Corr. (ppm): 0
Ch0: DC Offset Mode: Manual
Ch0: IQ Balance Mode: Automatic
Ch0: Gain Mode: Manual
Ch0: RF Gain (dB): 50
Ch0: IF Gain (dB): 50
Ch0: BB Gain (dB): 50

Signal Source
Sample Rate: 2M
Waveform: Cosine
Frequency: 0
Amplitude: 1
Offset: 0

WX GUI FFT Sink
Title: FFT Plot
Sample Rate: 2M
Baseband Freq: 96.9M
Y per Div: 10 dB
Y Divs: 10
Ref Level (dB): 0
Ref Scale (p2p): 2
FFT Size: 1.024k
Refresh Rate: 15
Freq Set Varname: None

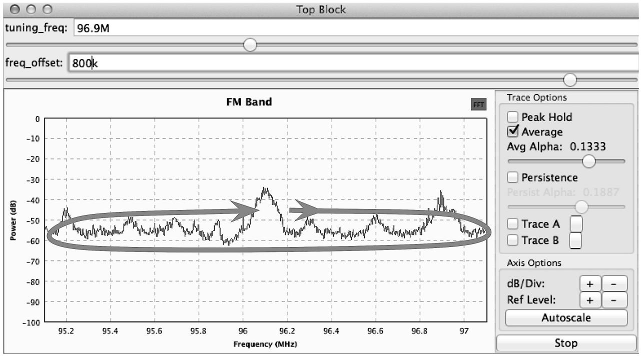
Low Pass Filter
Decimation: 10
Gain: 1
Sample Rate: 2M
Cutoff Freq: 100k
Transition Width: 5k
Window: Hamming
Beta: 6.76

WX GUI FFT Sink
Title: FFT Plot
Sample Rate: 200k
Baseband Freq: 96.9M
Y per Div: 10 dB
Y Divs: 10
Ref Level (dB): 0
Ref Scale (p2p): 2
FFT Size: 1.024k
Refresh Rate: 15
Freq Set Varname: None

FM Demod
Channel Rate: 200k
Audio Decimation: 4
Deviation: 75k
Audio Pass: 15k
Audio Stop: 16k
Gain: 1
Tau: 75u

Audio Sink
Sample Rate: 50k

FIGURE 6. Basic blocks for building a digital tuner.



tuning_freq: 96.9M
freq_offset: 800k

Trace Options
☐ Peak Hold
☑ Average
Avg Alpha: 0.1333
☐ Persistence
Persist Alpha: 0.1887
☐ Trace A
☐ Trace B

Axis Options
dB/Div: + -
Ref Level: + -

FIGURE 7. As the f_o frequency of the digital tuner of Figure 6 increases, the spectrums shifts cyclically.

Multi-rate signal processing is another important topic that is covered in the course. The idea is that the students must be aware of adjusting the sample rate at each point of the flow chart to be as low as possible (without aliasing) in order to reduce the computational burden. Using a graph similar to the one depicted in Figure 6, it is possible to experiment with the effects of down-sampling a digital signal in the frequency spectrum using many FFT probes along the graph path. In this particular example, the 2 Msps signal is down-sampled by a

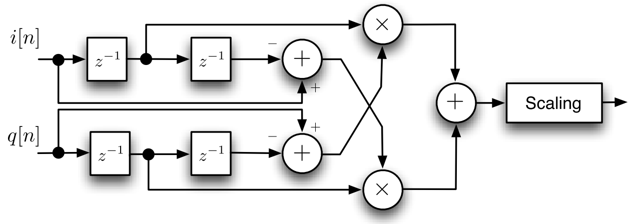


FIGURE 8. Graph that implements FM demodulation from the IQ samples.

factor of 10 before FM demodulation is carried out. Using the graph of Figure 6, the students can easily visualize how the digital spectrum widens as the signal is down-sampled.

C. FM DEMODULATION

There are many techniques to perform FM demodulation which are offered as *off-the-shelf* blocks in GNU Radio. Probably the best performance is obtained using a PLL-based (phase-locked loop) approach. Unfortunately, digital PLL design is out of the scope of our introductory course.

Before the students are allowed to use any GRC FM demodulation blocks, we ask them to implement FM demodulation using basic DSP blocks. To do this, an interesting option is to implement an FM quadrature demodulation technique that takes IQ samples as the input. Figure 8 shows the diagram that implements this FM demodulation approach (see [20] for the details). As can be observed, all of the basic blocks that build this demodulator are readily available in GRC. Another advantage of this exercise is that the students become familiar with the relation between the phase of complex baseband signals and their bandpass counterparts.

Once FM demodulation is complete, it is possible to listen to the demodulated signal (only the mono audio) by using a low pass filter and adjusting the sample rate to one of the standard rates used by sound cards. This last step involves the use of rational change of the sample rate (from 50 Ksps→ 48 ksps) and can be implemented with the use of polyphase filters, which are also studied in the course.

D. STEREO DECODING

Stereo decoding of a broadcast FM signal involves the steps depicted in Figure 9. In our opinion, stereo decoding is the part of the project that contains the most DSP concepts. Specifically, the students have to:

- Design a filter taking into account the signal delays in each branch.
- Obtain the carrier that is used to demodulate the L−R signal.
- Take care of the gains that should be applied so that the level of the L+R and L−R signals are commensurate.

The following subsections describe each of these concepts in greater detail.

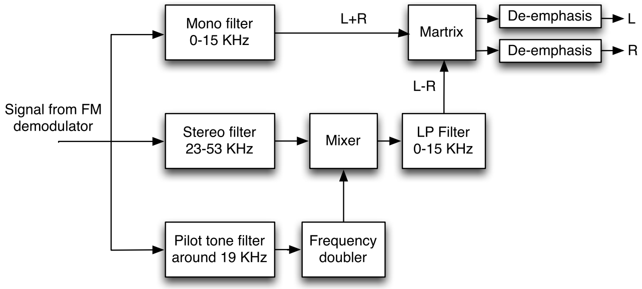
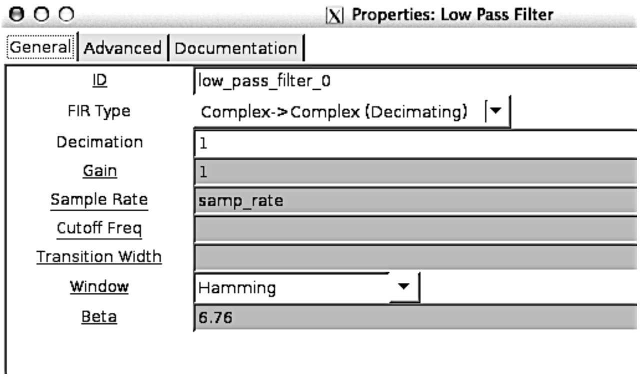


FIGURE 9. Simplified graph that implements stereo decoding of FM broadcast signals.

1) FILTER DESIGN

One of the common errors that students make when they design their filters is to ignore the filter delay. Taking control of the signal delay in all of the branches of the diagram of Figure 9 is essential so that the L+R and L−R signal reach the final combination matrix at the same time.

The most popular filter blocks in GRC have *finite impulse response* (FIR filter), so that the filter delay is given by $(N_{\text{coeff}} - 1)/2$, with N_{coeff} being the number of coefficients of filter impulse response. Unfortunately, the graphical interface of GRC hides this information from the user and only parameters such as *cutoff frequency* and *width of transition* bands are available (as shown in Figure 10).



Properties: Low Pass Filter

General | Advanced | Documentation

ID	low_pass_filter_0
FIR Type	Complex->Complex (Decimating)
Decimation	1
Gain	1
Sample Rate	samp_rate
Cutoff Freq	
Transition Width	
Window	Hamming
Beta	6.76

FIGURE 10. Available parameters for FIR filter design in GRC.

Once the students are aware of the delay problem, they are encouraged to do a little reverse engineering and find out which are the GNU Radio Python commands GRC uses to build a FIR filter based on the user constraints so that the length of the impulse response can be determined. Specifically, students have to check the delays at the mixer before the final combination matrix. Different delays of each branch can be easily fixed by using the delay block or by changing the transition width of the filters.

2) FREQUENCY DOUBLER

The best approach for doubling the frequency of a sinusoid is to use a PLL. However, since PLLs are out of the scope of our course as mentioned above, we propose that the students implement this step by squaring the pilot tone. This approach mimics the old analog processing.

The equation of the instantaneous deviation of the carrier frequency in FM broadcasting is:

$$\Delta f = \left[0.9\left[\frac{L+R}{2} + \frac{L-R}{2}\sin(4\pi f_p t)\right] + 0.1\sin(2\pi f_p t)\right] \tag{1}$$

where $f_p = 19$ kHz. is the frequency of the pilot tone. If we square the pilot tone, then we obtain:

$$\sin^2(2\pi f_p t) = \frac{1 - cos(4\pi f_p t)}{2} \tag{2}$$

Unfortunately, the carrier that is used to modulate the $L-R$ signal and the square of the pilot tone are in quadrature. This fact goes unnoticed by most students and the consequence is that a null L−R signal is obtained. To overcome this problem, it is necessary to shift the phase of the output of the mixer by $\pi/2$.

To do this, we suggest using a FIR filter with two non-zero coefficients:

$$h_{\pi/2}[n] = a\delta[n] + b\delta[n-1] \tag{3}$$

The FIR coefficients can be easily found by imposing the only condition that:

$$H_{\pi/2}(e^{j\omega})|_{38kHz} = j \tag{4}$$

In our opinion, this problem with the carrier phase illustrates the importance of phase in coherent very well demodulation and shows a simple way of designing filters that should meet restrictions at a number of finite frequencies.

3) GAIN ADJUSTMENTS

In order to mix the L+R and L−R correctly, it is essential to pay special attention to the signal levels in each branch. More specifically, the students have to compensate the 10% amplitude of the pilot tone given by Equation (1), the −6dB gain introduced by the squaring block given by Equation (2), and the −6dB produced by the mixer.

E. FM DE-EMPHASIS

De-emphasis is the last step before the audio signal can be heard. Pre- and de-emphasis in FM broadcasting gives improvement in received signal-to-noise ratio. Even though the higher audio frequencies are boosted at the transmitter, they can be attenuated at the receiver using a simple RC low pass filter.

A common option to approximate analog filters using DSP is the bilinear z-transform with pre-warping [17]. This transform preserves stability and maps every point of the frequency response of a continuous-time filter, $H_a(j\omega_a)$ to a corresponding point in the frequency response of the discrete-time filter $H_d(z)$.

Since the bandwidth of the modulating FM signals is about 15kHz, many of students down-sample the $L+R$ and $L-R$ signals before the combination matrix of Figure 9. However, the bilinear transformation may introduce a distortion in the frequency, and as a rule of thumb, the frequencies of interest should be less than one tenth of the sampling frequency. Figure 11 illustrates the attenuation that is produced by the bilinear approximation for different sampling frequencies.

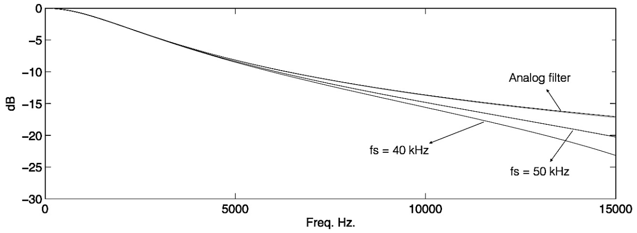
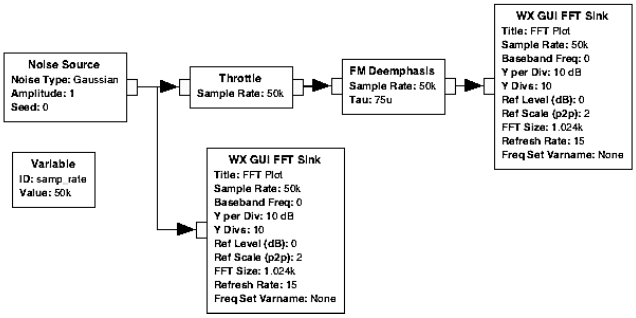


FIGURE 11. Distortion produced by bilinear approximation of the de-emphasis filter depending on the sampling rate.

To visualize the response of the de-emphasis filter, we asked the students to create a graph like the one shown in Figure 12, where the attenuation of the de-emphasis filter can be estimated from the shaping of the white noise spectrum.


Noise Source
Noise Type: Gaussian
Amplitude: 1
Seed: 0

Throttle
Sample Rate: 50k

FM Deemphasis
Sample Rate: 50k
Tau: 75u

WX GUI FFT Sink
Title: FFT Plot
Sample Rate: 50k
Baseband Freq: 0
Y per Div: 10 dB
Y Divs: 10
Ref Level (dB): 0
Ref Scale (p2p): 2
FFT Size: 1.024k
Refresh Rate: 15
Freq Set Varname: None

Variable
ID: samp_rate
Value: 50k

WX GUI FFT Sink
Title: FFT Plot
Sample Rate: 50k
Baseband Freq: 0
Y per Div: 10 dB
Y Divs: 10
Ref Level (dB): 0
Ref Scale (p2p): 2
FFT Size: 1.024k
Refresh Rate: 15
Freq Set Varname: None

FIGURE 12. Graph used to estimate the frequency response of the de-emphasis filter.

TABLE 2. DSP concepts that are covered during the design of an FM receiver.

DSP CONCEPT	WHERE
Sampling of pass band signals	RF signal capture
Spectrum of complex signals	RF signal capture
Variance of periodogram	Spectrum analysis w/ blocks
Filter design	Digital tuner and stereo decoding
FFT cyclic spectrum	Digital tuner using complex phasor
Bandpass and baseband signals	FM demodulation
Filter delay	Stereo decoding
Change of carrier phase	Demodulation of L-R signal
FFT modulation property	Demodulation of L-R signal
Analog filter approximation	FM De-emphasis
Filter shape estimation	White noise to study de-emphasis
Importance of correct sampling rate	Signal and FM de-emphasis

Table 2 summarizes the main DSP concepts that appear when designing an FM receiver. It can be observed that this simple project covers practically all of the course objectives presented in Section II.

V. ADVANCED TOPICS

This section gives some ideas of how to improve the basic FM stereo receiver that we just described. Some of these extensions are out of the scope of the average student, but they may be of great interest for highly motivated students.

A. AUTOMATIC MUTE

The first extension to the FM receiver is to mute the output audio when there isn't any tuned station. This can be easily performed in GNU Radio using the *power squelch* block that measures its input power and can block the input signal if its power is below a certain threshold.

B. STEREO MUTE

It is well known that noise in FM modulation affects higher frequencies more. For this reason several approaches have been proposed in the literature for blending the audio output from a received FM signal between stereo and mono [1].

A simple alternative is to measure the signal-to-noise ratio around the pilot tone using the probe blocks of GNU Radio. Figure 13 shows the spectrum of a demodulated FM signal with different signal-to-noise ratios. In cases where the measured S/N is too low (Figure 13-bottom), the L−R signal could be totally muted to improve the perceived audio quality.

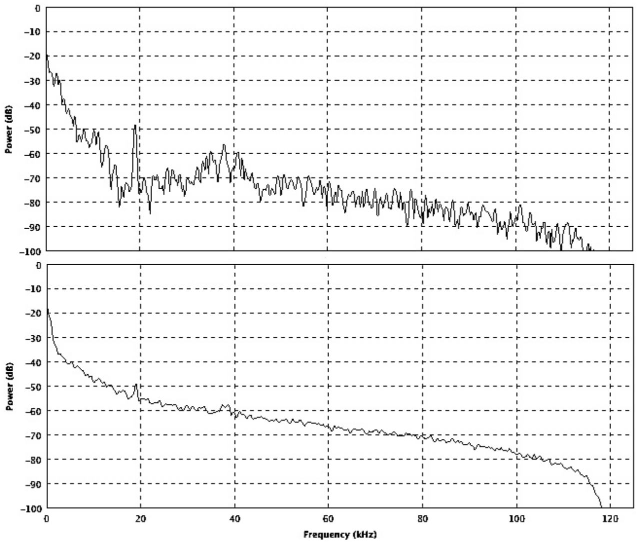


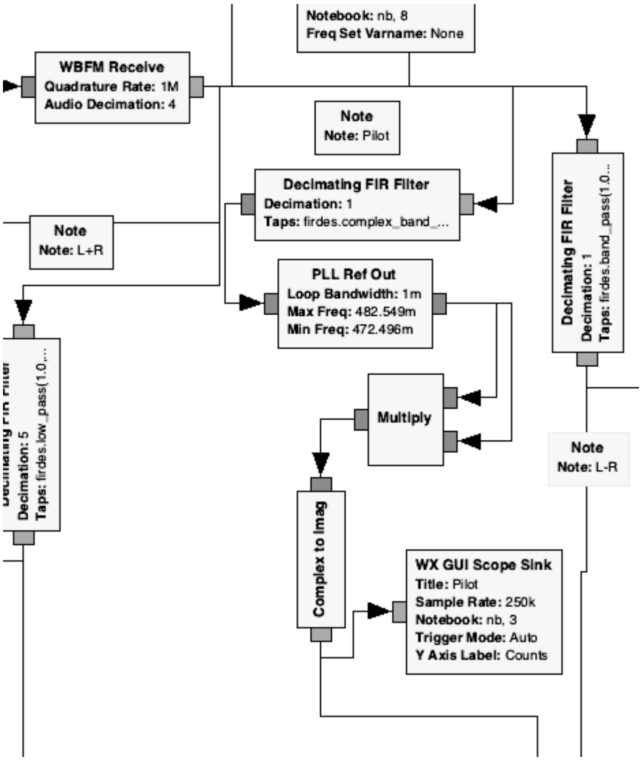
FIGURE 13. **Spectrum of the demodulated FM signal: with high S/N (top) and low S/N (bottom).**

C. PLL FM DEMODULATION

In the previous section, we selected a quadrature FM demodulation for its simplicity. However, for interested students, it could be possible to replace this block by one of the PLLs provided by GNU Radio or even implement one from scratch.

D. PLL SYNCHRONIZATION

Another important application of PLLs is carrier recovery. In the case of stereo FM, it is possible to use a PLL to recover the pilot tone as shown in Figure 14. In this case, it is recommended to use of the Xlating FIR filters which obtain the complex low-pass equivalent of band-pass signals


Notebook: nb, 8
Freq Set Varname: None

WBFM Receive
Quadrature Rate: 1M
Audio Decimation: 4

Note
Note: Pilot

Decimating FIR Filter
Decimation: 1
Taps: firdes.complex_band_...

Note
Note: L+R

PLL Ref Out
Loop Bandwidth: 1m
Max Freq: 482.549m
Min Freq: 472.496m

Decimating FIR Filter
Decimation: 1
Taps: firdes.band_pass(1.0...

Decimating FIR Filter
Decimation: 5
Taps: firdes.low_pass(1.0,...

Multiply

Note
Note: L-R

Complex to Imag

WX GUI Scope Sink
Title: Pilot
Sample Rate: 250k
Notebook: nb, 3
Trigger Mode: Auto
Y Axis Label: Counts

FIGURE 14. **Carrier recovery and frequency doubler using a PLL.**

(multiplying by a complex exponential followed by a complex low-pass filter). The advantage of this approach is that it retrieves both phase and quadrature components of the pilot tone and avoids the use of the phase shift filter that we mentioned in Equation (4).

E. RADIO DATA SYSTEM (RDS)

Perhaps, the most complex DSP that can be done in an FM receiver is the recovery of the RDS data, although many of the tools that are required to achieve it are more related to an introductory course on *digital communications*. There is an excellent GNU project that provides the RDS data parser blocks to do this [10]. However, the most exciting part for the students, from the point of DSP, is how to extract the bits from an analog sampled signal using the Mueller and Muller algorithm [8]. This algorithm is provided in GNU Radio by the multiple phase-shift keyed (MPSK) receiver block. Figure 15 shows the binary phase-shift keying (BPSK) constellation of the received RDS bits.

VI. ACADEMIC IMPACT

As stated in the introduction, the course presented here took place during four fall/winter academic semesters. At the end of each edition, all of the students completed a survey to determine the level of satisfaction in different areas (methodology, materials, and overall learning experience). As Figure 16 shows, the majority of students have given increasingly positive reviews about the content and methodology of the presented course.

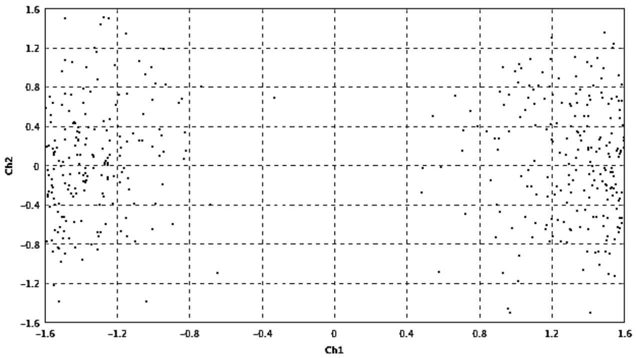


FIGURE 15. Constellation of the RDS data received.

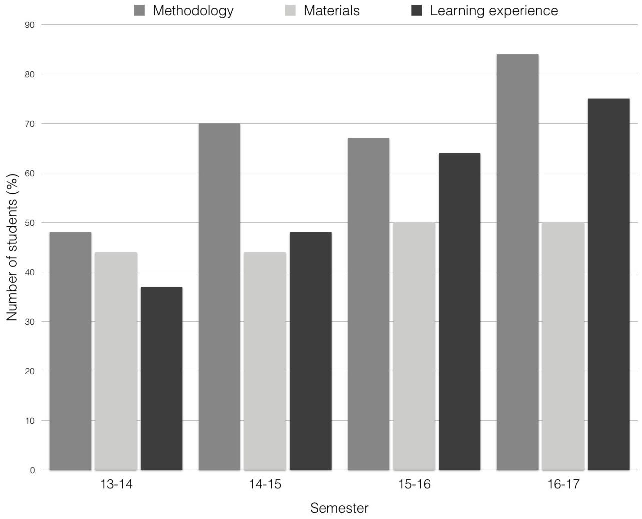


FIGURE 16. Evolution for each semester of the satisfaction of the enrolled students regarding several aspects of the DSP course (methodology, materials, and learning experience). Each bar represents the number of students (in %) who graded these aspects with a positive or very positive score.

Also, a great number of students completed all or some of the tasks from home. This fact shows that this approach can be applied in distant learning and even in MOOC-like scenarios.

In our online campus, we use a specific LTI tool [28] that allows the students to submit their homework and that autocorrects some of the items, as also explained in [7]. The autocorrection mechanism automatically sends back the achieved score/grade to the institution's gradebook. This grading tool performs an initial evaluation of the submitted homework, which basically consists of the validation of the XML files obtained with GRC (introduced in Section III). All initial student grades are later supervised by the course teacher. This is one of the key advantages of LTI: tool providers can communicate with central LMS systems to ease the correction process. This feature is key for the design of a future MOOC version of the course described here (or a similar one). As of today, MOOCs (especially MOOCs in the engineering field) face a number of challenges (such as content quality, methodology, quality, evaluation, and integration into the official curriculum) before they can be effectively used [3], [31]; however we believe this work represents a clear step forward.

VII. CONCLUSIONS

In this paper, we have presented an inexpensive way of practicing DSP concepts using a DVT TV capture dongle to digitalize RF signals and GRC as a graphical programming interface. The use of GRC makes it very easy to link and reconfigure different DSP elements and encourages students to experiment and learn by experience. We have shown that a simple project such as designing an FM receiver can cover most of the theory of an introductory DSP course. Although the course is focused on building this specific radio receiver, it is also possible to extend it to other areas such as the development of 433 MHz receivers, the analysis of a *global system for mobile communications* signals (GSM), the scaffolding of *automatic identification system* (AIS) and *automatic dependent surveillance* (ADS B) receivers, as well as practically any activity related to the interplay of signals that can be sampled up to 2 Msps.

The students submitted their homework electronically through the school LMS, but some of the tasks were designed to be computer-reviewed and the attained score was automatically inserted into each student's gradebook through an IMS Global LTI compliant application.

This experience has proven to be increasingly successful and may represent a nice starting point for the implementation of future MOOC environments in electronic engineering and DSP.

REFERENCES

[1] J. L. Allen, J. Elenes, and G. D. Champagne, "Methods and systems for blending between stereo and mono in a FM receiver," U.S. Patent 8 417 206, Apr. 9, 2013.

[2] E. Blossom, "GNU radio: Tools for exploring the radio frequency spectrum," *Linux J.*, vol. 2004, no. 122, p. 4, 2004.

[3] D. Burgos, "OER y currículo. colaboración sin servilismo," Res. Inst. Innov. Technol. Edu., Univ. Internacional de La Rioja, La Rioja, Spain, Tech. Rep., 2015.

[4] J. Campbell, F. Murtagh, and M. Kokuer, "DataLab-J: A signal and image processing laboratory for teaching and research," *IEEE Trans. Edu.*, vol. 44, no. 4, pp. 329–335, Nov. 2001.

[5] J. Chauhan *et al.*, "Towards adapting Sakai for e-learning provider," in *Proc. CSEDU*, vol. 1. 2015, pp. 306–314.

[6] G. Conole, "MOOCs as disruptive technologies: Strategies for enhancing the learner experience and quality of MOOCs," *Rev. Edu. Distancia*, no. 39, 2015.

[7] A. Corbi and D. Burgos, "Semi-automated correction tools for mathematics-based exercises in MOOC environments," *Int. J. Interact. Multimedia Artif. Intell. (IJIMAI)*, vol. 3, no. 3, pp. 89–95, 2015.

[8] G. R. Danesfahani and T. G. Jeans, "Optimisation of modified Mueller and Muller algorithm," *Electron. Lett.*, vol. 31, no. 13, pp. 1032–1033, Jun. 1995.

[9] M. Dillinger, K. Madani, and N. Alonistioti, *Software Defined Radio: Architectures, Systems and Functions*. Hoboken, NJ, USA: Wiley, 2003.

[10] M. Fähnle, "Software-Defined Radio with GNU Radio and USRP/2 hardware frontend: Setup and FM/GSM applications," Hochschule Ulm Univ. Appl. Sci., Inst. Commun. Technol., Ulm, Germany, Tech. Rep., 2010.

[11] L. D. Feisel and A. J. Rosa, "The role of the laboratory in undergraduate engineering education," *J. Eng. Edu.*, vol. 94, no. 1, pp. 121–130, Jan. 2005.

[12] A. J. S. Ferreira and F. J. O. Restivo, "Grasping the potential of digital signal processing through real-time DSP laboratory experiments," in *Proc. IEEE 10th Digit. Signal Process. Workshop, 2nd Signal Process. Edu. Workshop*, Oct. 2002, pp. 286–291.

[13] R. Ferzli and L. J. Karam, "An online Web-based real time digital signal processing course," in *Proc. 36th Annu. Frontiers Edu. Conf.*, Oct. 2006, pp. 6–11.

[14] W.-S. Gan, "Teaching and learning the hows and whys of real-time digital signal processing," *IEEE Trans. Edu.*, vol. 45, no. 4, pp. 336–343, Nov. 2002.

[15] J. Hamkins, M. K. Simon, and J. H. Yuen, Eds., *Autonomous Software-Defined Radio Receivers for Deep Space Applications*. Hoboken, NJ, USA: Wiley, 2007.

[16] O. Hazarika and A. K. Mishra, "A review of hardware platforms for whitespace communication," in *White Space Communication*. Springer, 2015, pp. 33–48. [Online]. Available: https://books.google.es/books/about/White_Space_Communication.html?id=G0vPBAAAQBAJ&source=kp_cover&redir_esc=y

[17] E. Ifeachor and B. Jervis, *Digital Signal Processing: A Practical Approach*, 2nd ed. London, U.K.: Pearson Education, 2002.

[18] C. Laufer, *The Hobbyist's Guide to the RTL-SDR: Really Cheap Software Defined Radio*. Seattle, WA, USA: Amazon, 2014.

[19] R. Lyons, "Quadrature signals: Complex but not complicated," Tech. Rep., 2008. [Online]. Available: https://dspguru.com/files/QuadSignals.pdf

[20] R. G. Lyons, *Understanding Digital Signal Processing*, 3rd ed. Englewood Cliffs, NJ, USA: Prentice-Hall, 2011.

[21] S. K. Mitra, *Digital Signal Processing: A Computer—Based Approach*, 4th ed. New York, NY, USA: McGraw-Hill, 2011.

[22] J. W. Eaton, D. Bateman, and S. Hauberg, *GNU Octave*. London, U.K.: Network Theory, 1997.

[23] A. V. Oppenheim and R. W. Schafer, *Discrete-Time Signal Processing*. Englewood Cliffs, NJ, USA: Prentice-Hall, 1989.

[24] J. G. Proakis and D. G. Manolakis, *Digital Signal Processing: Principles, Algorithms, and Applications*, 3rd ed. Upper Saddle River, NJ, USA: Prentice-Hall, 1996.

[25] M. H. Rahman and M. M. Islam, "A practical approach to spectrum analyzing unit using RTL-SDR," Dept. Elect. Electron. Eng., Chittagong Univ. Eng. Technol., Chittagong, Bangladesh, Tech. Rep., 2016, vol. 44, pp. 151–159.

[26] I. Ruano, P. Cano, and J. Gámez, and J. Gómez, "Advanced LMS integration of SCORM Web laboratories," *IEEE Access*, vol. 4, pp. 6352–6363, 2016.

[27] T. Sansaloni, A. Perez-Pascual, V. Torres, V. Almenar, J. F. Toledo, and J. Valls, "FFT spectrum analyzer project for teaching digital signal processing with FPGA devices," *IEEE Trans. Edu.*, vol. 50, no. 3, pp. 229–235, Aug. 2007.

[28] C. Severance, T. Hanss, and J. Hardin, "IMS learning tools interoperability: Enabling a mash-up approach to teaching and learning tools," *Technol., Instruct., Cognit. Learn.*, vol. 7, nos. 3–4, pp. 245–262, 2010.

[29] S. Sriram, G. Srivasta, R. Gandhiraj, and K. Soman, "Plug-ins for GNU radio companion," *Int. J. Comput. Appl.*, vol. 52, no. 16, pp. 11–16, 2012.

[30] P. Stoica and R. L. Moses, *Introduction to Spectral Analysis*, vol. 1. Upper Saddle River, NJ, USA: Prentice-Hall, 1997.

[31] UNESCO, "Qingdao declaration," in *Proc. Int. Conf. ICT Post Edu.*, Qingdao, China, May 2015.

[32] W. Zhan, J. R. Porter, and J. A. Morgan, "Experiential learning of digital communication using LabVIEW," *IEEE Trans. Edu.*, vol. 57, no. 1, pp. 34–41, Feb. 2014.

ALBERTO ALBIOL received the degree in telecommunications engineering and the Ph.D. degree with a focus on multimodal video indexing from the Universitat Politècnica de València (UPV) in 1995 and 2004, respectively. He started his teaching activities in 1995 at the University of Zaragoza. From 1999 until 2003, he stayed several times with Purdue University, IN, USA. He is currently an Associate Professor of signal and image processing with UPV. He has authored over 26 refereed publications in journals and in international conferences, and actively participates on the review process of many related journal and conference proceedings. His research interests include video analysis, image face recognition, pattern recognition, and computer vision applications.

ALBERTO CORBI received the M.Sc. degree in ocean-atmosphere interaction and the Ph.D. degree in medical physics from the Spanish Council for Scientific Research. He is currently a Senior Researcher with the Research Institute for Innovation and Technology in Education and also an Assistant Professor with the Engineering School, Universidad Internacional de La Rioja. With a background in physics, computing and education, he is currently involved in research fields around eLearning standards, open educational resources, and systems interoperability.

DANIEL BURGOS received the B.A. degree, the Ph.D. degree in communication, the Dr.Ing. degree in computer science, the Ph.D. degree in education, and the Ph.D. degree in anthropology. He is currently a Vice-Rector for Knowledge Transfer and Technology, the UNESCO Chair on eLearning and the ICDE Chair in OER with the Universidad Internacional de La Rioja. He is also the Director of the Research Institute for Innovation and Technology in Education. He contributes to research and practical projects about social entrepreneurship, regional open education, and clustered open education, funded by the European Commission among other institutions. His interests are mainly focused on educational technology and innovation. He has authored over 100 scientific papers and 15 books on these matters. He has been the Jury Chair of the 2016 UNESCO Prize for the Use of ICTs in education.