Considerations for the Design of a Hands-On Wireless Communications Graduate Course Based on Software-Defined Radio

Miguel Bazdresch
Electrical, Computer & Telecom Engineering Technology Dept.
Rochester Institute of Technology
Rochester, NY
Email: mxbiee@rit.edu

***Abstract*—Essentially, a software-defined radio (SDR) receiver can be described as a radio divided into two different parts. One, the analog front-end, down-converts and samples an RF wireless signal. The other, a back-end implemented in software or digital hardware, processes the samples and recovers the transmitted information, whether it is audio, video, instrumentation data, or whatever else. An SDR transmitter performs the opposite operations. Software-defined radio has opened up tremendous education opportunities. Probably the most intriguing one is to give students the ability to experiment with actual physical signals and systems. Another benefit is that the analog front-end is flexible enough to cover a very large frequency range, allowing students to use the same device to transmit and receive a very large variety of signals. Finally, all this can be achieved without students having to design RF electronic circuits, a subject that deserves its own dedicated course. However, actually realizing the potential educational benefits promised by SDR is far from trivial. In particular, this subject has not been deeply explored in the specific context of a telecommunications engineering technology curriculum. In this paper, we offer some reflections about the experience of designing a graduate elective course on wireless communications based on SDR. We cover aspects such as prerequisites, choice of platform, and selecting appropriate educational objectives and cognitive progression. Since the digital back-end performs many functions necessary for retrieving the information from the received signal (such as carrier acquisition, symbol and frame synchronization, filtering, decimation, de-interleaving, and error control), it becomes crucial to decide how students should tackle these problems, and in which order. We also present concrete examples of student assignments that use an SDR.**

I. Introduction

The use of software-defined radio (SDR) as a teaching and learning tool in engineering education is becoming more prevalent. With experience, the challenges and benefits of this approach are becoming clearer, too. In this paper, we share our experience in designing laboratory exercises for graduate students in our Telecommunications Engineering Technology program. We describe the analysis used to reach decisions on how to use SDR, and to what specific pedagogic purpose. To the author's knowledge, this is one of the first reports on the application of SDR to engineering technology education.

It has been recognized for many years that digital signal processing, and platforms based on it, would play an increasingly important role in teaching communications systems, whether analog or digital [1]. Student motivation is a key benefit, since it allows for experimentation with real signals and systems. Also, it is generally easier to implement complex systems in the digital domain. One illustration of this is fixing implementation errors: DSP code can be debugged with software aids, whereas finding a fault in a crowded breadboard or PCB is a daunting task. Another benefit is giving students the engineering skills demanded by employers, which have adopted DSP and SDR techniques wholeheartedly.

A software-defined radio is the final product of this push towards DSP. A simple description of an SDR is that it is a radio which is divided into two parts. One part is the analog front-end, which deals with antenna transmission and reception, filtering, amplification and frequency shifting to/from an intermediate frequency. The analog front-end is designed to be as capable as possible, in terms of bandwidth, noise figure and carrier frequency. The second part is a digital back-end, which processes the baseband signal. This processing is carried out by an FPGA, a digital signal processor and/or a general purpose computer. Between the front-end and the back-end sits a digital-to-analog converter (in the transmitter) and an analog-to-digital converter (in the receiver).

The actual function of the radio is defined by the software running in the backend (hence its name). This has clear benefits for research and development. For instance, a new receiver algorithm or a new modulation type can be quickly tested by writing new software. Devices in the field can be upgraded with new features or bug fixes by delivering a firmware update. Multiband radios become possible, without requiring a different analog receiver for each band or protocol. (See more extensive introductions to the technical aspects of SDR and their pedagogical potential in [2]–[6]).

The digital signal processor has opened new horizons in signals and systems education, and the software-defined radio has started to do the same for communications courses. As the technology matures and new teaching methods are tested and refined, educators start to confront arising challenges and propose innovative solutions. Some issues often discussed in the literature are related to cost, the use of simulation vs

physical signals, teaching with analog and/or digital signals, the use of SDR as a demonstration tool or as a student-centered learning tool, and whether to use it to teach communications, or teach SDR as an end in itself.

For instance, the cost of commercially available SDR platforms can be high; strategies for addressing this issue have been reported in [7]–[11]; on the opposite end of the cost spectrum, positive results with very high end equipment are reported in [12].

Another commonly reported analysis is related to the extent to which it is convenient to stick to discrete-time numerical simulation (using Matlab or similar tools) or emulation using a DSP to process baseband signals, compared to using an SDR with actual passband, usually externally generated signals. The question before instructors and curriculum designers is how to balance theoretical complexity, hardware complexity, and time available for laboratories with the desired educational outcomes. The DSP baseband approach is presented in [7]–[9]. The simulation approach is evaluated in [13], [14]. A hybrid approach with digital downconversion and offline processing in Matlab is presented in [11].

Most of the papers reporting on the use of SDR focus on analog modulations (AM or FM) either exclusively or to a large degree [2], [4], [5], [12], [15], [16]. Some courses include digital modulation, often in addition to analog modulation, and frequently based on decoding an FM station's RDBS data [7], [9], [11], [13], [17]–[19]. Of these, only [9] makes reference to the block diagram of a complete digital receiver; most stop with a constellation or eye diagram plot. Analog modulations, while old-fashioned, retain a very high educational value because of their simplicity and ubiquity. With a few lines of DSP code, a student can obtain the spectrum of a broadcast station's signal, and listen to it. On the other hand, a digital communications receiver is more modern and relevant to today's industry, but it requires several blocks (for example, carrier, symbol and frame synchronizers, a matched filter, a decision device, error decoders, etcetera), each of which has a certain degree of mathematical complexity. While the design of such a system is within the reach of undergraduate students [20], [21], it takes a considerable amount of time and effort.

Another aspect related to the technical difficulty involved in the design of a communications systems is the question of whether to use an SDR platform as a demonstration tool, instead of as a vehicle for hands-on student experience. Papers that focus on demonstrations or on pre-assembled examples are [3], [9], [16], [17], [19]. Papers that focus on SDR use for senior or final projects are [5], [22].

Another consideration is whether to use SDR to teach communications, to teach a much more specific subject, or to teach implementation and design of software defined radios themselves. Examples of courses that are very specific are [8] (focused on bandpass sampling and rate conversion), and [23] (focused on cybersecurity for wireless networks). Courses that emphasize the design and implementation of software defined radios are found in [2], [15]. Each approach has unique challenges and benefits that are described in the referenced papers.

The rest of the paper is organized as follows. First, we summarize a series of considerations that affected the design of the proposed SDR-based laboratory activities. In Section II, we present technical considerations, while we focus on educational aspects in Section III. In Section IV we present our proposed laboratory assignments in more detail, focusing on the educational outcomes we chose to emphasize and on how to achieve them. In section V we present results regarding student achievements and feedback. We present our conclusions in Section VI.

II. Technical Considerations

In this section we address some technical aspects that should be considered when designing a course that uses software defined radio as a learning tool. We focus on cost and feasibility. We have tried to include considerations that are up to date and not frequently mentioned in the literature.

A. Cost

Not surprisingly, the cost of SDR hardware has decreased dramatically in the past few years. Today, it is possible to receive and process signals for a total investment of approximately $50 USD. This is possible by the emergence of inexpensive RF front-ends (such as the RTL-SDR device), as well as computers such as the Raspberry Pi, which have enough processing power to handle a variety of signals. If one is willing to learn about its idiosyncrasies, open-source software offerings miss few of the capabilities of much more expensive commercial software. For students, becoming productive with an unfamiliar software interface may be a valuable skill.

With increasing cost, the front end becomes more capable in four aspects: transmission capability, bandwidth, sampling rate, and bits per sample. These features cannot be ignored except in the simplest cases. Being able to transmit is important for activities where students design their own signals, instead of just processing existing signals. Increased bandwidth and sampling rates open up the capability to work with wideband signals, such as WiFi. It is worth noting that a large majority of new and upcoming wireless standards use wideband signals. Finally, a lower quantization noise (provided by 16-bit converters instead of 8-bit in the less expensive radios) enables better reception of weak signals and improved bit recovery with higher-order digital modulation schemes. Other features of more costly devices include an embedded FPGA that can be used to offload functions from the main processor. In courses with a digital design focus this may be an interesting option [15].

Currently, a $1,000 USD SDR features a transceiver, 16 bits per sample, a large FPGA, at least 50 MHz bandwidth, at least 6 GHz carrier frequency, connection to an external high-precision clock reference, and a USB 3.0 connection. These capabilities should be enough for most educational purposes.

B. Feasibility

Beyond pure financial cost, there are other important considerations that may influence the feasibility of adding SDR

topics to a course. For instance, number of radios per student. On one hand, it may be desirable to provide every student with one radio. Such a setup involves one computer, one extra power supply, at least one antenna, and several cables and adapters per seat. This extra inventory must be budgeted and managed. In addition, most available SDR platforms are designed as prototyping devices, not necessarily designed to withstand the typical abuse suffered by more rugged laboratory instruments. Related to this aspect is the question of providing students with radios to take home with them. Confining the availability of hardware to a shared-use laboratory will restrict the time students have to work on them, which may in turn limit the complexity and depth of the proposed experiments and activities.

The space required for an SDR-based laboratory may also be an important consideration. Assigning one SDR per student may be difficult for large groups. In our experience, setting students up in teams of two works very well.

III. Educational Considerations

In this section we present some considerations related to the educational choices that can be made regarding the desired outcomes of the SDR-based laboratory activities.

A. Pre-requisites

Even the simplest AM receiver is a complex system, and the complexity increases quickly as one moves to digital signaling, higher order modulations and wireless fading channels. In this sense, designing an SDR-based communications system is a multidisciplinary task [22]. Every approach to SDR requires specific pre-requisites from the students. However, some of them are practically unavoidable in all but the simplest SDR experiments, and are listed below. Consider these to be the minimum signal-processing concepts that students are required to master before being able to design, implement and test an amplitude-modulated double sideband, large carrier transceiver.

1) Sampling. A good grasp of sampling theory and its practical consequences is unavoidable. In most SDR platforms, the DAC and ADC operate at fixed rates, which determine the maximum signal bandwidth.
2) Multirate signal processing. This arises in almost all cases when two hardware devices need to share samples. In the case of an AM receiver, the samples are produced by a radio at a certain rate, and consumed by the sound card, at (almost surely) a different rate. Changing the sampling rate is also necessary when dealing with signals that are much narrower than the hardware's sampling rate, in order to reduce the computational load on the processor.
3) Quadrature and complex envelope. All software defined radios operate in quadrature, producing and consuming complex samples. Furthermore, the baseband signals they produce and consume are complex envelope signals. This issue can be swept under the carpet by zeroing the imaginary part of all samples; however, in our opinion it is better not to ignore it and help students get a good understanding of it.
4) Frequency up and downconversion. This is related to the confusion that can arise between the concepts of modulation and frequency shifting. In SDR, a baseband signal is modulated in software (for example, an FSK signal is generated from data bits), and it is upconverted to a carrier frequency by the analog front-end.
5) Filtering. A good grasp of the different kinds of filters and when to use them is necessary. An example is receiving a 10 kHz AM station with a radio sampling at 256,000 samples per second. Most of the spectrum contains noise, which must be filtered to improve signal quality. Since the filter is digital, the student must understand how to select its cutoff frequency, its transition bandwidth, and its order. It is also a good idea to understand the correlation between filter order and computational load; every student is initially tempted to use extremely high-order filters, and is thereby introduced to the concept of buffer overflow, when the processor cannot keep up.
6) Hardware imperfections. The difference between theory and practice is very quickly brought to the forefront by software defined radios. Harmonics are very easy to generate by clipping the radio's input. Likewise, out of band emissions are very clearly seen in a spectrum analyzer. DC offset is an important concept, because practically all radios suffer from it in some measure; students must learn to tell it apart from actual signals.
7) Hardware interfaces. Many signals can be produced by hardware (a microphone, a signal generator, a sensor), or sent to different hardware once processed (for example, to a sound card in the AM receiving example). Students must learn how to connect these hardware devices and configure their interfaces.
8) Programming language or tool. Since an SDR's function is provided by software, all but the simplest, pre-assembled experiments will require some programming. Some tools, such as Matlab's Simulink and GNU Radio Companion present visual interfaces to an underlying library. Even then, the tools are not trivial to use and configure. Students should be comfortable with concepts such as streaming, real time processing and signal flowgraphs.

This is a daunting list, and shows the difficulty in hands-on activities involving communications and SDR. The benefit is that the students who succeed in getting their systems to work will have acquired a mastery of these subjects that is hard to achieve with lecturing alone. With careful planning, most students are capable of succeeding at this. Depending on students' previous courses and experience, some of these topics will be well known, superficially known, or completely new. It is up to each instructor to identify in advance the skill set of the students and plan accordingly. In the worst case, a large percentage of lecture and laboratory time might

be spent in background material, precluding more complex experiments.

Specific approaches will require other pre-requisites. For instance, a focus on algorithm implementation on an FPGA will require a good grasp of digital design principles, VHDL or Verilog, synthesis, and hardware testing. A focus on RF design requires strong electronic design skills and being able to create an interface to the DAC and ADC circuits.

B. Educational purpose

An SDR platform can be used to fulfill many different educational purposes, some of which are listed below.

1) As part of a course on communications, whether analog, digital, wireline or wireless.
2) On a course on digital design, and on implementation of communications algorithms in digital hardware [3], [15].
3) As part of a course on radio frequency electronic design [18].
4) On a course about the implementation and evaluation of software defined radio platforms [2].
5) As part of a course on a specific aspect of telecommunications and networking.

Furthermore, for each of these alternatives student involvement can varied, from observers of demonstrations that complement a lecture-based course, to implementers and testers in a more creative laboratory course. For each of these a different set of pre-requisites and pedagogical objectives can be defined.

IV. Course Description

The course described in this section is part of a telecommunications engineering technology. It is an elective graduate course. Results are described in the next section.

It is common in telecommunications engineering technology programs to emphasize networking [24], and ours is no exception. Our students have a superficial grasp of the prerequisites listed in section III-A, either because these were not emphasized in their undergraduate courses or because a long time has passed. This dictates that a significant proportion of the time will be spent reviewing these pre-requisites. On the positive side, graduate students who choose this elective can be relied on to be mature, motivated and willing to spend the required study time.

The course as offered today consists of 12 weeks of lecture and 3 weeks of laboratory time. This somewhat unusual arrangement is temporary and will be substituted by two courses, one lecture and one laboratory.

We have selected the following educational outcomes for the laboratory portion of the course. At the end of the laboratory activities, we wish students to have:

1) A much more in-depth knowledge of each of the eight topics listed as pre-requisites in section III-A.
2) Proven ability to develop and test an analog communications system capable of transmitting and receiving AM DSB-LC audio signals.
3) Proven ability to receive a digital signal modulated with BPSK and QPSK and obtain their eye and constellation diagrams.

Regarding tools, we have chosen to use Ettus Research B100 SDRs, which offer the required flexibility. On the software side we use GNU Radio Companion, a free and open source front-end to the extensive GNU Radio signal processing library. The use of a visual tool simplifies the design of flowgraphs and the hardware configuration. Contrary to other reports [17], we have not found the lack of familiarity with Linux and GNU Radio Companion to be a problem for students, which pick up their use in a matter of minutes.

During the three weeks devoted to the laboratory, students perform the following tasks:

1) Generation and transmission of simple sinusoidal signals, complex and real, and verification of correct transmission using an spectrum analyzer.
2) Creation of signals that match a pre-established spectrum, using real and complex sinusoids plus amplifiers. Use the main instrumentation blocks in GNU Radio: FFT sink, Scope sink, and Waterfall sink. Interpret the measurements.
3) Use the SDR to explore the frequency spectrum, from 50 MHz to 2.2 GHz. This is enabled by a wideband antenna. Students must identify and document five different kinds of signals. Examples of commonly found signals are aeronautical beacons, public safety channels, TV stations, and LTE links.
4) Resampling by connecting the radio to and from the computer's sound card. Transmission of a sinusoid of variable frequency and amplitude to another computer, where it is reproduced.
5) Implementation of an AM DSB-SC transmitter with a naive (no PLL) receiver, to witness the reality of the consequences of the lack of frequency and phase synchronization between transmitter and receiver. Subsequently, implementation of an AM DSB-LC system to eliminate these issues.
6) Generation of generic BPSK and QPSK signals in Matlab, which are saved to a WAV file. The signals are transmitted over the air, and upon reception, their eye diagrams and constellations are plotted. Fading effects (such as constellation rotation) are identified.
7) Implementation of a system to receive RDBS signals from an FM station, and plotting of the eye diagram and constellation of the resulting signal. The main difference with the previous activity is that the digital signal is an actual broadcast signal that carries actual data instead of being generated artificially in Matlab.

As can be seen, the activities start slowly, reviewing and building upon the pre-requisites. Simple examples (AM) of actual signal transmission and reception are proposed later, and digital communication experiments are performed quite quickly at the end. It is a testament to the power and flexibility of SDR as a teaching tool that students can complete so much

work in so little time.

V. Results

In this section, we describe the results obtained in two editions of the course described in the previous section.

A. Student accomplishments

One hundred percent of students (11 students over two editions of the course) completed the assigned tasks successfully and submitted complete and correct reports. It must be emphasized that the level of accomplishment is very high. This level of success could not have been predicted strictly from the corresponding accomplishment in the lecture part of the course, in which the variance is much higher. However, after the laboratory experiments, students are able to answer theoretical questions better than before performing the experiments. Both from conversation and from the improvement in their work, we are confident in saying that the educational outcomes that we set for our students are being met.

B. Student feedback

Student feedback is uniformly positive and enthusiastic; the only complain is that they would wish to perform more experiments. A common proposal is to interlace lecture with lab, so that theoretical concepts and ideas can be quickly tested and reinforced with a hands-on activity in the laboratory. This will be implemented in a future version of the course.

VI. Conclusions

In this paper, we have shared a number of considerations that influenced our decisions when planning a graduate engineering technology course that incorporates software defined radio as a learning tool. These considerations are general enough that they may be of use to instructors planning courses along these lines. We have described the course we designed, where the educational objectives and activities are a direct result of the considerations mentioned earlier. We have described a very high level of student accomplishment and satisfaction with the course. We finish by heartily recommending all instructors of communications courses to look into adding SDR to their courses.

References

[1] M. A. Yoder, J. H. McClellan, and R. W. Schafer, "Crystal radios or DSP first?" in *28th Annual Frontiers in Education Conference, 1998*, vol. 2, Nov 1998, pp. 695–699 vol.2.

[2] C. Dietrich, D. Miller, F. Kragh, and J. Reed, "Education in software defined radio design engineering," in *2008 Annual Conference & Exposition*. Pittsburgh, Pennsylvania: ASEE Conferences, June 2008, https://peer.asee.org/4112.

[3] L. S. Nagurney, "Software defined radio in the electrical and computer engineering curriculum," in *2009 39th IEEE Frontiers in Education Conference*, Oct 2009, pp. 1–6.

[4] J. Hoffbeck, "Teaching communication systems using the universal software radio peripheral (USRP) and GNU Radio," in *2009 Annual Conference & Exposition*. Austin, Texas: ASEE Conferences, June 2009, https://peer.asee.org/5126.

[5] S. Mao, Y. Huang, Y. Li, P. Agrawal, and J. K. Tugnait, "Introducing software defined radio into undergraduate wireless engineering curriculum through a hands-on approach," in *2013 ASEE Annual Conference*. Atlanta, Georgia: ASEE Conferences, June 2013, https://peer.asee.org/19836.

[6] S. Mao, Y. Huang, and Y. Li, "On developing a software defined radio laboratory course for undergraduate wireless engineering curriculum," in *2014 ASEE Annual Conference*. Indianapolis, Indiana: ASEE Conferences, June 2014, https://peer.asee.org/22880.

[7] C. H. G. Wright, T. B. Welch, and M. G. Morrow, "An inexpensive method to teach hands-on digital communications," in *33rd Annual Frontiers in Education Conference, 2003*, vol. 2, Nov 2003, pp. F2E–20–5 Vol.2.

[8] C. Wright, M. Morrow, and T. Welch, "Using inexpensive hardware and software tools to teach software defined radio," in *2010 Annual Conference & Exposition*. Louisville, Kentucky: ASEE Conferences, June 2010, https://peer.asee.org/15922.

[9] J. P. Hoffbeck and M. M. Sugiyama, "Real-time FM radio for teaching DSP and communication systems," in *2013 IEEE Frontiers in Education Conference (FIE)*, Oct 2013, pp. 1087–1090.

[10] C. J. Prust, S. Holland, and R. W. Kelnhofer, "Ultra low-cost software-defined radio: A mobile studio for teaching digital signal processing," in *2014 ASEE Annual Conference*. Indianapolis, Indiana: ASEE Conferences, June 2014, https://peer.asee.org/23216.

[11] M. Morrow, T. Kent, T. Welch, and C. Wright, "Teaching with software defined radios," in *2009 Annual Conference & Exposition*. Austin, Texas: ASEE Conferences, June 2009, https://peer.asee.org/4938.

[12] T. Welch, R. Kubichek, and C. Wright, "A comprehensive suite of tools for teaching communications courses," in *2006 Annual Conference & Exposition*. Chicago, Illinois: ASEE Conferences, June 2006, https://peer.asee.org/913.

[13] J. Frolik, "Laboratory enhancement of digital and wireless communications courses," in *2005 Annual Conference*. Portland, Oregon: ASEE Conferences, June 2005, https://peer.asee.org/15072.

[14] A. Melton and J. Hoffbeck, "RF signal database for a communication systems course," in *2006 Annual Conference & Exposition*. Chicago, Illinois: ASEE Conferences, June 2006, https://peer.asee.org/584.

[15] S. Bilen, "Implementing a hands on course in software defined radio," in *2006 Annual Conference & Exposition*. Chicago, Illinois: ASEE Conferences, June 2006, https://peer.asee.org/1232.

[16] S. Katz and J. Flynn, "Using software defined radio (SDR) to demonstrate concepts in communications and signal processing courses," in *39th IEEE Frontiers in Education Conference, 2009*, Oct 2009, pp. 1–6.

[17] A. M. Wyglinski and D. J. Cullen, "Digital communication systems education via software-defined radio experimentation," in *2011 Annual Conference & Exposition*. Vancouver, BC: ASEE Conferences, June 2011, https://peer.asee.org/17783.

[18] T. Barton and M. Alvira, "An advanced level radio-frequency circuit course with laboratory and design exercises that emphasize complete system performance," in *2011 Frontiers in Education Conference (FIE)*, Oct 2011, pp. S4G–1–S4G–6.

[19] J. P. Hoffbeck, "Teaching communication systems with simulink and the USRP," in *2012 ASEE Annual Conference*. San Antonio, Texas: ASEE Conferences, June 2012, https://peer.asee.org/22000.

[20] M. Bazdresch, "A real-time, Matlab-based undergraduate digital communications course," in *Digital Signal Processing Workshop and IEEE Signal Processing Education Workshop (DSP/SPE), 2011 IEEE*, Jan 2011, pp. 408–413.

[21] J. Gunther and T. Moon, "Synchronization and demodulation programming projects to accompany a first course on digital communications," in *Digital Signal Processing and Signal Processing Education Meeting (DSP/SPE), 2013 IEEE*, Aug 2013, pp. 311–316.

[22] J. Flynn and S. Katz, "Using software defined radio for multidisciplinary senior design projects," in *2011 Annual Conference & Exposition*. Vancouver, BC: ASEE Conferences, June 2011, https://peer.asee.org/18524.

[23] C. Sahin, D. Nguyen, J. Chacko, and K. R. Dandekar, "Wireless cybersecurity education via a software defined radio laboratory," in *Frontiers in Education Conference (FIE), 2015.*, Oct 2015, pp. 1–8.

[24] A. Lozano, "Satellite communications experiences for electrical engineering technology students," in *2002 Annual Conference*. Montreal, Canada: ASEE Conferences, June 2002, https://peer.asee.org/10917.