AC 2009-1017: TEACHING COMMUNICATION SYSTEMS USING THE UNIVERSAL SOFTWARE RADIO PERIPHERAL (USRP) AND GNU RADIO

Joseph Hoffbeck, University of Portland

Joseph P. Hoffbeck is an Associate Professor of Electrical Engineering at the University of Portland. He has B.S.E.E, M.S.E.E., and Ph.D. degrees from Purdue University, West Lafayette, Indiana. He worked with digital cellular telephone systems at Lucent Technologies (formerly AT&T Bell Labs) in Whippany, New Jersey. He is a member of the IEEE and the ASEE, and his technical interests include communication systems, digital signal processing, and remote sensing. His email address is hoffbeck@up.edu.

Teaching Communication Systems using the Universal Software Radio Peripheral (USRP) and GNU Radio

Abstract

The Universal Software Radio Peripheral (USRP) is a moderately priced hardware board that implements software defined radio systems. It allows a computer to acquire and generate RF signals in a similar way that a sound card allows a computer capture and produce audio signals. The board includes high speed analog-to-digital converters (ADC), high speed digital-to-analog converters (DAC), digital up and down converters, and a universal serial bus (USB) interface. Various daughter boards are available that interface a wide range of radio frequency (RF) signals with the ADC's and DAC's. The USRP is an open source platform, so the schematic and design files are freely available.

In addition, the USRP is compatible with the open source project GNU Radio that has implemented a vast array of algorithms needed to construct analog and digital communication systems (modulation, source coding, error correction, interleaving, filtering, etc.). The low-level algorithms are written in C++ and are controlled by high-level Python programs. The source code for GNU Radio is also freely available.

Together the USRP and GNU Radio form a powerful and flexible platform that allows the user to implement various real-time communication systems simply by writing software. Receivers that operate in real-time can be constructed for such commercial systems as AM and FM radio, TV, WWV time signal, etc. Two-way real-time systems such as ham radio, family services radio, etc. can also be constructed. The USRP and GNU Radio also make an excellent platform for implementing custom and experimental communication systems.

This paper evaluates the USRP board and the GNU Radio software as tools for teaching communication systems courses. The capabilities and limitations of the device and software are discussed, and ideas for laboratory experiments and projects are presented. This approach to teaching communication systems is compared to software-based and hardware-based simulated systems. The evaluation is based on the capabilities and limitations of the USRP and GNU Radio, the author's experience using some of the proposed projects in a communication systems course using MATLAB (but not the USRP and GNU Radio), and on the author's experience of implementing some of the projects with the USRP and GNU Radio (but not as part of a course). None of the proposed experiments have yet been incorporated into the author's communication systems course using the USRP and GNU Radio.

Background

It is often useful to motivate students to learn the mathematical theory of communication systems by showing how the theory applies to real world systems. Some of the possible ways to show this link include using examples or homework problems dealing with real communication systems, using software to simulate communication systems[1,2,3], using hardware to perform the

processing used in communication systems[4], and using specialized hardware to capture RF signals from real systems[5,6,7]. Although the USRP has been used to teach a course specifically about software defined radio[8], this paper discusses how it may be useful in teaching a general communication systems course.

USRP

The USRP is a hardware board that digitizes RF signals so that they can be processed by a computer, and allows digital signals generated by a computer to be converted to RF (see Figure 1). This board allows the user to implement prototypes of most radio systems in software, greatly simplifying the task. The USRP can receive and transmit signals simultaneously, and operates in real-time. The USRP is open source, and so all the design files are freely available. An overview article about the USRP is available[9] and an article describing how to use the USRP to listen to FM radio has been published[10]. The USRP, which was designed by Matt Ettus, can be purchased from Ettus Research[11].

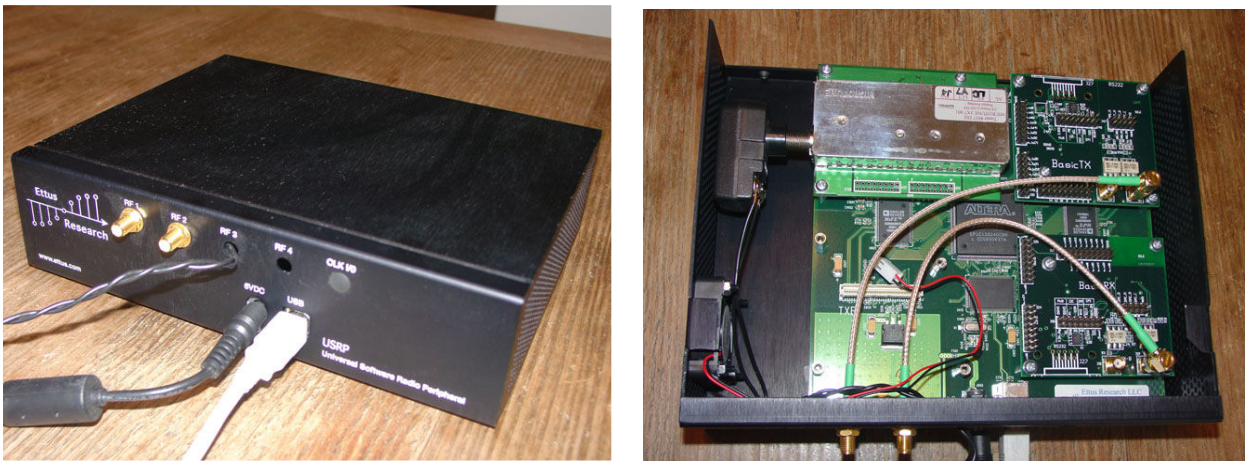

Figure 1. The USRP with the Cover On (left) and Off (right).

Although the USRP can be used by itself for a few applications, RF signals are usually interfaced to the USRP using daughterboards which plug into sockets on the USRP and are also available from Ettus Research. The USRP can accommodate up to two receive daughterboards and two transmit daughterboards as shown in Figure 2. Daughterboards are available that cover the range 0 to 5.9 GHz, although each daughterboard is designed to receive and/or transmit on a particular frequency band, and none of them cover the entire range from 0 to 5.9 GHz. The purpose of the receive daughterboards is to amplify the desired signal and shift it down to an intermediate frequency where it can be converted to a digital signal by the USRP. The transmit daughterboards convert the signal to the desired RF frequency and, in some cases, amplify it so it can be transmitted.

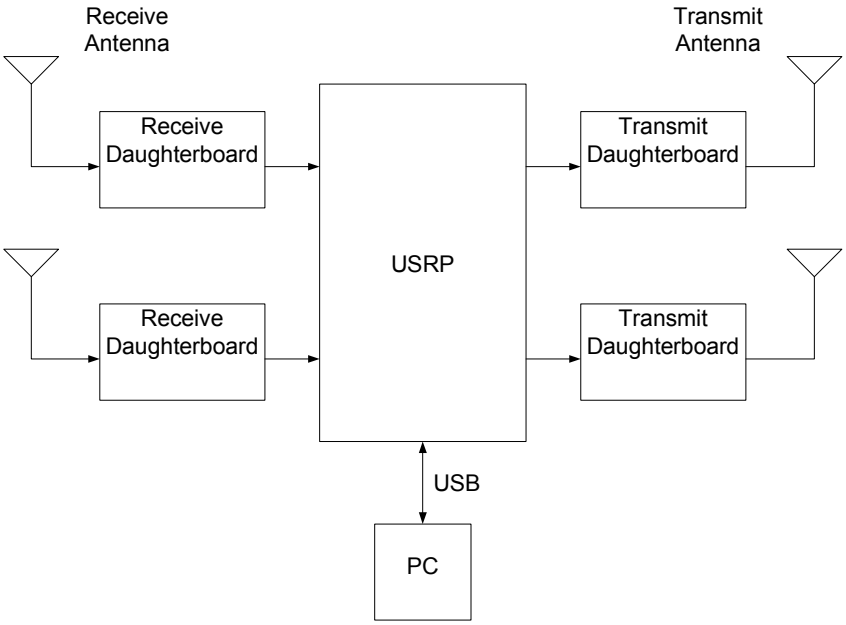

Figure 2. Block Diagram of USRP, Daughterboards, and PC

A block diagram of the USRP is shown in Figure 3. On the receive side, the USRP has programmable gain amplifiers to provide up to 20 dB of gain and four high speed analog-to-digital converters (ADC) to support IQ sampling from two RF signals (or real sampling of four signals). The ADC's are 12-bit devices that run at 64 M samples/sec. They are connected to a field-programmable gate array (FPGA) which includes a digital down converter (DDC) to reduce the data rate. The FPGA then sends the data to a computer through a USB 2.0 interface. The USRP can support a data rate through the USB bus of up to 32 M bytes/sec, which allows an RF signal with a bandwidth of up to 8 MHz to be captured by the USRP (using 16 bit samples) and transmitted to the computer.

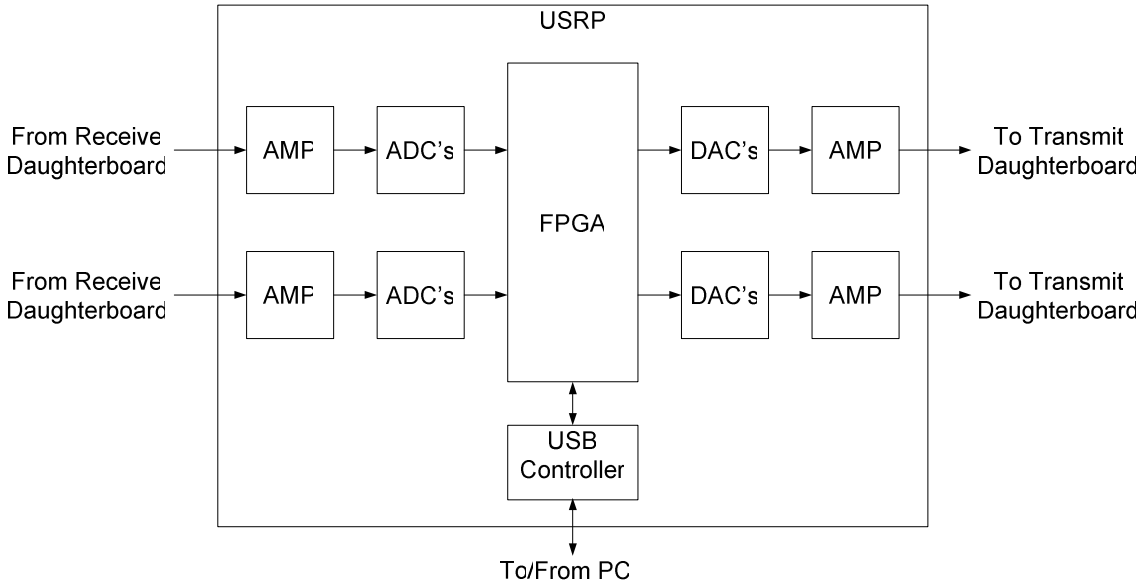

Figure 3. Block Diagram of the USRP

On the transmit side, the computer generates a digital transmit signal which is sent to the USRP through the USB 2.0 bus. This signal is interpolated to 32 M samples/sec by the FPGA and passed to four high speed digital-to-analog converters (DAC). The DAC's interpolate by a factor of four to 128 M samples/sec, perform upconversion to the desired frequency, and convert the digital signal to an analog signal[12]. The DAC's are 14 bit devices. The four DAC's can support IQ signals for two RF signals (or four real RF signals). There are programmable gain amplifiers to provide amplification up to 20 dB. A signal with a bandwidth of up to about 8 MHz (using 16 bit samples) can be generated by the computer and transmitted to the USRP over the USB 2.0 bus. The USRP can transmit and receive at the same time, but the 32 M bytes/sec capacity of the USB bus is shared by the receive and transmit operations, so the USRP can transmit 8 MHz only if it is not receiving. If the USRP is receiving and transmitting simultaneously, then the bandwidth of the receive signal plus the bandwidth of the transmit signal cannot exceed 8 MHz (using 16 bit samples).

A new version of the USRP, called the USRP2, is being designed to support faster ADC's and DAC's with higher resolution (14 bit, 100 M samples/sec ADC's, and 16 bit, 400 M sample/sec DAC's), a faster connection to the computer (Gigabit Ethernet), and stand-alone operation. Both the USRP and the USRP2 are designed to work with GNU Radio software.

GNU Radio

GNU Radio is an open source software project led by Eric Blossom. It includes a vast array of routines useful in implementing communication systems, such as modulation and demodulation, source coding and decoding, channel coding and decoding, filtering, etc., as well as the routines needed to interface the computer to the USRP and other hardware[13]. The low level routines, which are written in C++, are connected to each other using Python. This organization allows GNU Radio to benefit from the high speed and efficiency of C++ and the relative ease of use of the high level interpreted language Python. In addition, a program called GNU Radio Companion is available that allows the user to connect blocks using a graphical interface instead of Python. There is a good tutorial for the USRP and GNU Radio available[14], and there is a good tutorial for Python at the main Python website[15].

Comparison to Other Communication Systems Platforms

There are other systems that perform a similar function to the USRP and GNU Radio. For example, a product called SignalWAVe, which is sold by Lyrtech, has been used in a course on software defined radio[16]. SignalWAVe is a platform for implementing software defined radios, and it allows the user to program the device using Mathwork's Simulink graphical-based programming environment. Unlike the USRP which uses a PC to perform the baseband processing, the SignalWAVe system includes a Texas Instruments DSP chip on board. Sundance has a similar system called the SMT8036E Software Defined Radio Development (SDR) Kit which is also integrated with Simulink and contains a DSP chip. It is, perhaps, an advantage to have a DSP chip on the board, as most software defined radios would have, but the cost of these two systems is much higher than the cost of the USRP, and neither of these systems is open source system.

There are other systems that can capture real RF signals for off-line (non-real-time) processing such as the Tektronix real time spectrum analyzers and the vector signal analyzers from National Instruments and Agilent[5,6,7]. These instruments are easier to use than the USRP, but they are much more expensive.

Emona markets a system called the TIMS-301 that includes many basic hardware modules such as adders, multipliers, filters, oscillators, and amplifiers. These modules can be easily connected to perform modulation, demodulation, and many other operations needed to implement communication systems. However this system is set up to operate only on low frequency signals (below about 100 KHz), so students cannot build systems to demodulate RF signals from real systems such as AM or FM radio, and it is much more expensive than the USRP. A much less expensive approach has been proposed that uses custom circuitry to demonstrate AM and FM modulation and demodulation[4]. This approach has the advantage of demonstrating the chips and circuits that are likely to be used for real systems and it is much less expensive than the USRP, but it is limited to processing low frequencies (below about 300 KHz) and is not as flexible as the USRP.

Ways to use USRP and GNU Radio in Communication Courses

The USRP and GNU Radio are very powerful and flexible, and therefore could be used in teaching a communication systems course in several ways, such as implementing real-time communication systems, recording real RF signals for off-line processing, and converting signals that were generated by a computer to RF.

Some relatively simple real-time communication systems that could be implemented using the USRP are listed below.

- FM Radio Receiver. The USRP along with the TVRX daughterboard could be used to implement an FM receiver. The TVRX daughterboard receives frequencies between 50 and 870 MHz and contains a preamplifier, so it can be connected directly to an antenna such as the SA7000 wideband receive only antenna sold by AOR. The students would know immediately if their receiver was implemented correctly by listening to the received signal.

- RDS Decoder. The USRP and the TVRX daughterboard could also be used to decode the digital RDS signal from an FM radio station that contains the name of the program, song, and artist, along with the current time and other information[17]. The students could verify the output of the decoder by comparing the RDS text output with the song currently being played on the FM radio station.

- AM Radio Receiver. The daughterboards BasicRX and LFRX can be connected to an antenna to receive strong AM radio stations, but an external preamplifier would be needed to receive moderate or weak stations.

- Radio-Controlled Clock. The WWV or WWVB stations in Colorado transmit the official current time from the atomic clocks in a simple digital BCD format[18] which could be received by the USRP with the BasicRX and LFRX daughterboards and an external preamplifier. The WWV station also contains an analog audio signal that announces the time.

- AM or FM Transmitter. An AM or FM transmitter could be implemented using the USRP, the BasicTX daughterboard, and an external amplifier. The transmitter could be tested by transmitting the signal to either a commercial radio or a USRP and listening to the result.

- Family Radio Services (FRS) Walkie-Talkie. Students could implement a FRS walkie-talkie and communicate with commercial FRS walkie-talkies or other USRP's using the RFX400 daughterboard.

- Remote control. Students could implement a remote control transmitter to allow a computer to generate and send commands to control a remote controlled car or other device.

- Many other communication devices could also be implemented with the USRP.

Another way to use the USRP and GNU Radio in a communication system course is to use the device to capture real RF signals from existing systems, save the signals to a file, and process the signals off-line using a high level language such as MATLAB. This approach reduces the complexity of the programming, but still retains a direct connection to real systems. The GNU Radio command usrp_rx_cfile.px can be used to capture an RF signal and save it to a file. By default, this command saves the data in 32-bit floating point numbers with the in-phase values interleaved with the quadrature values. This file can then be read and processed by almost any computer language. For example, the students might use the USRP to record an AM or FM radio station and then load the file, demodulate it, and listen to the result using MATLAB. This approach requires no low level real-time programming, but still allows the students to hear or see the output of a real system. Recordings of real RF signals could also be used to demonstrate the bandwidth, modulation type, data rate, or other characteristics of the signal.

In a similar way, the USRP could be used to take a signal that was generated off-line by a PC and transmit it to a commercial system. So, for example, a student might write a program to perform frequency modulation in a high level language, and then test the program by using the USRP to transmit the signal to a commercial FM radio and listen to the result.

Another possible approach is to have the students implement only part of a system, and interface it to an existing program from GNU Radio. For example, a receiver for terrestrial over-the-air digital TV (DTV) has been written for GNU Radio, and since it is an open source project, the source code is freely available. Therefore it is possible to have the students write software to perform only part of the processing needed to decode the signal and test it by inserting the code into the GNU Radio DTV receiver. For example, the students might implement the phase-locked loop, matched filter, and timing recovery algorithms, and use the existing GNU Radio algorithms

to perform the rest of the processing (equalizer, de-multiplexing, trellis decoding, de-interleaving, channel decoding, de-randomizer, source decoding, etc.). Using this approach, the students can see video that verifies that their programs are working correctly without the need for writing the software for the entire system.

Comparison of Different Approaches

The big advantage of using the USRP and GNU Radio in teaching communication system courses is that it allows the students to implement actual real-time communication systems in software, as opposed to running simulations. The concepts and algorithms taught in the course can be implemented in software and tried out in real systems. Thus the link to the theory and how it is used in practice is made obvious, and implementing a real system vividly illustrates the context and the usefulness of the material. Hearing or seeing a real system operate which was implemented by the student should generate excitement and satisfaction beyond that generated by a simulation.

A potential disadvantage of using the USRP and GNU Radio is that the programming environment may be unfamiliar to the students and the instructors. Although the USRP is compatible with Windows, Mac, and Linux operating systems, it appears that most of the users use Linux and that support for the USRP and GNU Radio is focused on Linux. Learning how to write the low level routines and connect them to other GNU Radio routines would require some time. If students or instructors are not familiar with Linux, do not already know Python, or are not experienced with C++, significant time may be required to become proficient in implementing communication systems in this environment.

Another challenge is navigating the documentation for USRP and GNU Radio. Many resources are available in various locations. The main GNU Radio website[13] contains instructions for downloading and installing the software as well as some information about the USRP and the USRP2. Although the instructions are probably fine for experienced Linux users and developers, people new to Linux may find installing the software and getting it running to be a non-trivial task. The main GNU Radio website has a link to the email list for discussing issues regarding GNU Radio and asking the user community questions, which can be a very valuable resource. Much of the information needed to get started is in the FAQ's from this mailing list, the USRP and GNU Radio tutorial[14], and the Python tutorial[15].

When compared to other methods of teaching communication systems, the USRP and GNU Radio have the potential to engage the students and generate excitement about the material in a way that simulations do not, but this excitement requires mastering a potentially unfamiliar and complex programming environment.

Conclusion

The USRP and GNU Radio represent a flexible and powerful platform that is useful for teaching communication systems. They allow students to implement actual real-time communication systems in software. The challenges of using this system include a possibly unfamiliar software development environment and less-than-perfect documentation. The benefits are the low cost, open source environment, and the ability to allow the students to implement a real world communication system.

References

1. Johnson, C. Richard Jr. and Sethares, William A., Telecommunication Breakdown, Pearson Prentice Hall, 2004, ISBN: 0-13-143047-5.
2. Silage, Dennis, "Teaching Digital Communications in a Wireless World: Who Needs Equations?", 2006 ASEE Annual Conference and Exposition, Chicago, Illinois, June 18-21, 2006.
3. Tanyel, Murat and Nguru, Kathrine, "Preparation of a Virtual Toolkit for Communication Systems," 2002 ASEE Annual Conference and Exposition, Montréal, Québec, June 16-19, 2002.
4. Dunne, Bruce, "Design of a Hardware Platform for Analog Communications Laboratory," 2008 ASEE Annual Conference and Exposition, Pittsburgh, Pennsylvania, June 22-25, 2008.
5. Kubichek, Robert; Welch, Thad; and Wright, Cameron, "A Comprehensive Suite of Tools for Teaching Communications Courses," 2006 ASEE Annual Conference and Exposition, Chicago, Illinois, June 18-21, 2006.
6. Hoffbeck, Joseph P., "RF Signal Database for a Communication Systems Course," 2006 ASEE Annual Conference and Exposition, Chicago, Illinois, June 18-21, 2006.
7. Welch, Thad B. and Kubichek, Robert F., "The Incredible Hulk and Other Techniques for Teaching Waveform Demodulation," 2005 ASEE Annual Conference and Exposition, Portland, Oregon, June 12-15, 2005.
8. Kragh, Frank; Reed, Jeffery; Dietrich, Carl; Miller, Donna, "Education in Software Defined Radio Design Engineering," 2008 ASEE Annual Conference and Exposition, Pittsburgh, Pennsylvania, June 22-25, 2008.
9. Blossom, Eric, "GNU Radio: Tools for Exploring the RF Spectrum," Linux Journal, Issue 122, June 2004. An updated version is available at www.gnu.org/software/gnuradio/doc/exploring-gnuradio.html.
10. Blossom, Eric, "Listening to FM Radio in Software, Step by Step," Linux Journal, September 1, 2004.
11. Ettus Research, LLC, www.ettus.com.
12. USRP FAQ, Digital Upconverter Questions, http://gnuradio.org/trac/wiki/UsrpFAQ/DUC.
13. GNU Radio, http://gnuradio.org/trac.
14. Documentation – RadioWare Project, University of Notre Dame, https://radioware.nd.edu/front-page/documentation.
15. Python Programming Language -- Official Website, www.python.org.
16. Bilen, Sven, "Implementing a Hands-On Course in Software-Defined Radio," 2006 ASEE Annual Conference and Exposition, Chicago, Illinois, June 18-21, 2006.
17. Hoffbeck, Joseph P., "Using Real RF Signals Such as FM Radio to Teach Concepts in Communication Systems," 2008 ASEE Annual Conference and Exposition, Pittsburgh, Pennsylvania, June 22-25, 2008.
18. NIST Time and Frequency Division, http://tf.nist.gov.