STUDYING MEDIA ACCESS AND CONTROL PROTOCOLS

Alaelddin Fuad Yousif Mohammed
Department of Computer Engineering, Faculty of Engineering and Technology, University of Gezira,Wad Medani, Sudan

Keywords: GNU Radio, USRP, MAC, Bluetooth, IEEE 802.11.

Abstract: The goal of this project is to enable undergraduate students to gain insight into media access and control protocols based upon carrying out laboratory experiments. The educational goal is to de-mystifying radio and other link and physical layer communication technologies as the students can follow packets from the higher layers down through the physical layer and back up again. The project fills the gap between the existing documentation for the Universal Software Radio Peripheral (USRP) resources and the knowledge of undergraduate students. The project is targeted at (1) instructors of undergraduates who might use this work to develop their own lesson plans and course material and (2) students of physical and link layer protocols who want a practical tool for carrying out experiments in these layers.

1 INTRODUCTION

Software-defined radio (SDR) introduces flexibility and rapid development to radio communication systems by using a software-oriented approach. As software-based approach offers greater flexibility when developing wireless communication systems, since the wireless system architecture is not frozen into the hardware, but can be changed at any time via changing the software which is loaded into the device. By delaying the binding of design decisions until execution time, the designers can incorporate the latest developments - enabling them to improve the performance of the systems. This reduces the difference between the state of the art and the state of practice for wireless communication systems. Additionally, this software-oriented approach to wireless communication devices allows both flexibility and simpler maintenance, as most upgrades can be done by loading new software, rather than changing physical modules.

The Ettus Research Universal Software Radio Peripheral (USRP) is an example of an SDR. It provides an “RF front end for a computer running the GNU Radio software, converting radio waves picked up by an antenna into digital copies that the computer software can handle or, conversely, converting a wave synthesized by the computer into a radio transmission” (1). In Figure 1, "IF" standards for intermediate frequency, representing a version of the signal at a lower frequency that the actual RF.

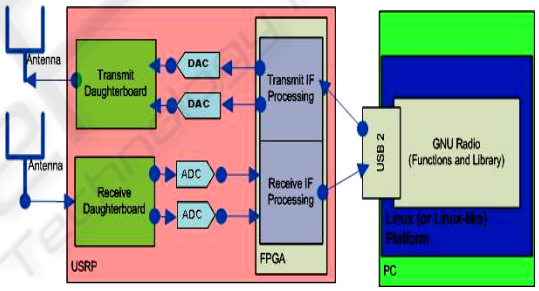

Figure 1: Introduction to USRP and GNU Radio.

2 SOFTWARE DEFINED RADIO (SDR) HISTORY

A SDR is a radio in which software defines signals, frequencies, modulation, and (optionally) cryptography. SDR design began 1987, when the United States Air Force’s Rome Laboratory (AFRL) developed a programmable modem. The modem was based on the Integrated Communications, Navigation, and Identification Architecture (ICNIA) (2). The term software defined radio was introduced by Joseph Mitola III in 1991 "to signal the shift from digital radio to multiband multimode software-defined radios where "80%" of the functionality is provided in software, versus the "80%" hardware of the 1990's." (3).

Fuad Yousif Mohammed A. (2010).
STUDYING MEDIA ACCESS AND CONTROL PROTOCOLS.
In *Proceedings of the International Conference on Data Communication Networking and Optical Communication Systems*, pages 131-134
DOI: 10.5220/0002999301310134

Table 1: Time line with some representation examples.

Prject	Size	Features
ICNIA, 1978	Fit in a small room	A collection of several single-purpose radios in one box
Speakeasy Phase I, 1992	Six foot (182 cm) rack	Included a programmable cryptography chip.
Speakeasy Phase II, 1995	Stack of two pizza boxes	The first SDR to include a voice coder and digital signal processing resources.
Digital Modular Radio, 2000	44.45 x 48.90 x 55.9 cm	Implemented four full duplex channels and could be remotely controlled using the Simple Network Management Protocol via an Ethernet interface.
USRP, 2004	Fit in 21 x 17 x 5.5 cm box	Allows creating a software radio using any computer with a USB2 port.

3 THE UNIVERSAL SOFTWARE RADIO PERIPHERAL (USRP)

The USRP has a Cypress FX2 USB 2.0 interface, four high speeds digital to analog converters, four high speed analog to digital converters, and a large Altera Cyclone field programmable gate array (FPGA) that interconnects all of the aforementioned devices.

The USRP includes Analog Devices Mixed Signal Processor (AD9862). Each AD9862 contains four ADCs. Programmable gain amplifiers, placed in-front of the ADCs provides input signal level adjustment. Further details of the AD9862 can be found at (6). Each of ADCs runs at 64 Million samples per second (64 Msps) with 12 bits per sample, the DAC accept as input 14 bits per sample generating 128 Msps. As the maximum signaling rate of a USB 2.0 link is 480 Mbps, this means that we can not simply forward the entire received signal to an attached processor - nor can we receive a signal from an attached processor and output it directly via the DAC. Reducing the sample rate in the receive path and increasing the sample rate in the transmit path must be accomplished by the FPGA.

3.1 USRP Daughter Boards

There are four expansion slots on the USRP mother board. These enable a user to plug in up to two transmitter daughter boards and two receiver daughter boards. These daughters implement the specific radio frequency front end for a given range of frequencies. Thus the motherboard only performs baseband (or intermediate frequency) processing of the signals. On the USRP motherboard the transmitter expansion slots are labelled TXA and TXB, while the receiver expansion slots are labeled RXA and RXB. Each transmitter expansion slot has access to two high speed DACs; as the motherboard has four DACs with two connected to TXA and two to TXB. Each receiver expansion slot has access to two high speed ADCs, as the motherboard has four ADCs with two for RXA and two for RXB.

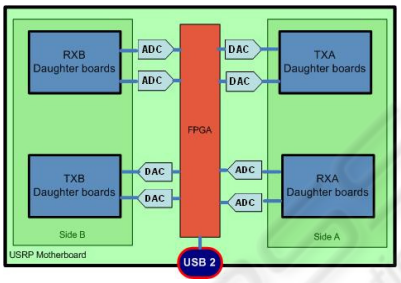

Figure 2: USRP Block Diagram.

4 GNU RADIO

GNU Radio is free Python-based software architecture implemented to run on a Linux platform. More specifically, GNU Radio provides a collection of signal possessing blocks that support the USRP.

The GNU Radio code is written in both C++ and Python. The computationally intensive processing blocks are implemented in C++, while Python is used for developing applications that sit on top (and control) these blocks. The GNU radio code assumes that the FPGA has already been programmed with a configuration suitable for use by the GNU radio code.

4.1 Installing the GNU Radio

This section describes how to build GNU Radio version 3.2.2 - released on July 15, 2009. In this project we experienced problems installing GNU Radio as described in this release's build guide [9].

The basic concepts underlying the GNU Radio are flow graphs and blocks (nodes of the graph). The blocks carry out the actual signal processing. The data passed between these blocks could be of any kind.

4.2 GNU Radio Signal Processing Blocks

The GNU Radio project provides many signal processing blocks (implemented in C++) as a library

and supports the ability to be establish connections between these blocks. The programmer develops a radio by building a flow graph in which the signal processing blocks are represented as vertices and the data flow between them is represented as edges. Blocks' attributes specify the number of input ports and/or output ports and the data type (for example: short, float, and complex) for this port.

5 LABORATORY EXPERIMENTS

This section describes some of the laboratory exercises that have been designed during this project.

5.1 Experiment 1: Simplex Data Transmission

In this exercise we will learn how simplex data communication can be implemented. In this case there will not be any feedback from the receiver packets arrival of the packets at the receiver. The transmitter sends 5 packets, then waits one second and sends the next 5 packets. The equipment required for this exercise is a PC, together with one USRP, Basic TX, Basic RX, and an RF cable. Based on the exercise plan, the following objectives were developed for the simplex data transmission.

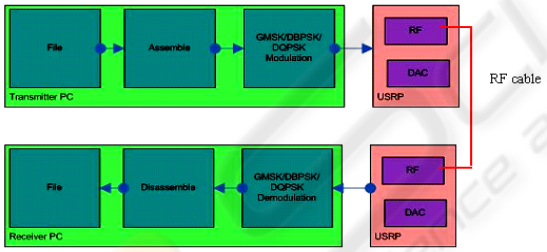

Figure 3: Simplex data transmitter and receiver.

5.2 Experiment 2: Voice Transmission

This experiment is similar to experiment 1; but instead of a file we are sending and receiving a voice signal. The code uses GSM-FR encoder and decoder to as a voice CODEC. The code used in this exercise is part of the GNU Radio examples.

5.3 Experiment 3: Carrier Sense Multiple Access Protocol

In this experiment we introduce Carrier Sense Multiple Access (CSMA) (without collision detection) as a link layer protocol. This experiment illustrates a common media access and control protocol (MAC). Its also provides a framework for students to build their own MACs, by modifying the code. In this experiment we use the "TUN/TAP" Linux interface to intercept frames that are being sent to (or received from) a virtual network interface. This enables the student to run any network protocol or higher level protocol of their choice – while seeng the frames passed to their MAC and physical layer.

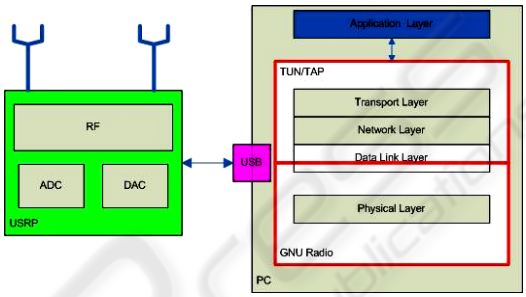

Figure 4: TUN/TAP and GNU Radio.

5.4 Experiment 4: Bluetooth (or IEEE 802.15.4) Sniffer

It is hard to sniff Bluetooth because of its wide frequency band and fast random hopping (calculated by the master device). We need eleven USRPs to sniff the 83.5 MHz wide band (USRP can work with 8 MHz wide band centred in a frequency), or we can use four USRP2.

In this experiment we use gr-bluetooth. This code was developed by Dominic Spill and Michael Ossmann (10) and they made the code freely available (11). This experiment is used to listen to packets exchanged between a cellular phone and a Bluetooth headset.

5.5 Experiment 5: IEEE 802.11 Implementation

In this experiment we use the BBN 802.11 implementation by the Adaptive Dynamic Radio Open-source Intelligent Team and funded by DARPA's ACERT program. This project used GNU Radio and implemented an 802.11 receiver and transmitter (12).

6 EVALUATION AND ANALYSIS

In this section we will evaluate each of the laboratory experiment from a pedagogical point of view.

6.1 GNU Radio: Analysis

There is no enghough documentation of how GNU Radio is implemented, during runig application we found that there are some messages printed from diffrernt *classes* and tracing and understanding these message takes some time. The GNU Radio developers did not found acceptable way to provide unifed documentation for the system (15). The Gnu Radio has many releases developed. In release version **3.2.x** the higher block of the system is updated. This will affect applications developed under old release from running in new releses.

6.2 USRP: Analysis

The USRP is a device we used in this project to develop undergraduate's experiment . This device has various daughterboards which operate on different radio frequency bands (from DC to 2.9 GHz); you have to plug-in a sutable daughterboard for you application.

USRP2 was developed and goes to the market on May 25, 2009. There are some benefits of using USRP2 than USRP(5).

6.3 Laboratory Exercises: Analysis

The laboratory exercises were designed based upon the idea of step-by-step learning. The undergraduate student initialy follow the steps presented in each experiment to solve a problem and understands subject terms. These experiments start with simple communication systems first, a little bit complex systems, and finaly real world systems. In each experiment, the student must solve specific problems and submit a written report to the instructor. The instructor can choose which experiment are sutable for the students.

7 CONCLUSIONS

We developed laboratory experiment for undergraduate students to help them understands media and access control protocols protocol. The experiments are designed in a way that easy to understand experiments first, and the complicated experiments. Instructors might use these experiments and add more exercises to develop their own lessons plan and course material. You can find the full work and codes in (15).

In conclusion, we can say that it is not easy job to implement applications using USRP and GNU Radio because of the weak documentation of the GNU Radio. And if we started this project again we would develop a documentations tool for GNU Radio to help developers to implement their own applications.

REFERENCES

Susan Karlin,"Tools & Toys: Hardware for your Software Radio", IEEE Spectrum, 34(10) , Oct. 2006, pp51-54.

Bruce A. Fette, et al, "Cognitive Radio Technology" Newnes, 2006, 656 pages, ISBN-10: 0750679522, ISBN-13: 978-0750679527.

Greg Colvin and Beman Dawes, Smart Pointers, Web Page, March 11, 2009, http:// www.boost.org/libs/smart_ptr/smart_ptr.htm.

Walter Tuttlebee, et al, "Software Defined Radio: Enabling Technology", USA, John Wiley & Sons Ltd, 2002.

Ettus Research LLC, web page " www.ettus.com", visited 2009-11-11.

Analog Design, AD9862 12-/14-Bit Mixed Signal Front-End (MxFE®) Processor for Broadband Communications, Data Sheet, Revision 0, Dec. 2002, internet site "http://www.analog.com/static/imported-files/data_sheets/AD9860_9862.pdf"

GNU Radio, web page" www.gnuradio.org ", visited 2009-02-10

Greg Colvin and Beman Dawes, Smart Pointers, Web Page, March 11, 2009, http:// www.boost.org/libs/smart_ptr/smart_ptr.htm.

Build Guide- GNU Radio, web page,2009-11-05 " http://gnuradio.org/trac/wiki/BuildGuide".

Michael Ossmann and Dominic Spill, "Building an All-Challe Bluetooth Monitor", ShmooCon 2009, 6 February 2009.

Gr-Bluetooth, web page, Aug 18, 2009," http://sourceforge.net/projects/gr-bluetooth/"

Troxel Gregory D, Blossom Eric, et al "Adaptive Dynamic Radio Open-source Intelligent Team (ADROIT): Cognitively-controlled Collaboration among SDR Nodes", Networking Technologies for Software Defined Radio Networks, 2006. SDR '06.1st IEEE Workshop, Sep 2006, pp 8-17, ISBN: 1-4244-0733-8.

BBN80211 - The Comprehensive GNU Radio Archive Network, web page, "https://128.2.212.19/ wiki/BBN80211", visited Nov 7, 2009.

GNU Radio 3.2svn C++ API Documentation, web page, "http://gnuradio.org/doc/doxygen/index.html", May 22, 2009.

Alaelddin Mohammed, "Studying Media Access and Control Protocols", Master Thesis, 2010-02-19, "http://web.it.kth.se/~maguire/DEGREE-PROJECT-REPORTS/100119-Alalelddin_Mohammed-with-cover.pdf".