Reconfigurable Software Defined Radio and Its Applications

Chi-Yuan Chen¹, Fan-Hsun Tseng², Kai-Di Chang¹, Han-Chieh Chao¹²* and Jiann-Liang Chen³

¹Department of Electrical Engineering, National Dong Hwa University, Hualien, Taiwan 974, R.O.C.
²Institute of Computer Science & Information Engineering and Department of Electronic Engineering, National I-Lan University, I-Lan, Taiwan 260, R.O.C.
³Department of Electrical Engineering, National Taiwan University of Science and Technology, Taipei, Taiwan 106, R.O.C.

Abstract

Currently, most Software Defined Radio (SDR) related products and researches focus on analog communication and voice transmission. In this paper, we propose a SDR platform with digital data communication capability. This platform consists of Field Programmable Gate Array (FPGA) based radio hardware and open source SDR software modules. The main features include: 1) Radio Spectrum Sensing; 2) Reconfigurable Radio Modules; 3) Link for Digital Data Communication. Based on the proposed SDR platform, we could easily reconfigure its radio modules and discover the spectrum hole to achieve better communication quality. These features are important basis to accomplish Cognitive Radio (CR) technologies.

Key Words: Digital Data Communication, Spectrum Sensing, Software Defined Radio, Cognitive Radio

1. Introduction

The mature development of radio technology brings novel wireless applications into people’s life. The mobile devices can afford the high speed and complex computation owing to the advance in computing ability of the processor, such as PDA (Personal Digital Assistant), Smart Phone, or UMPC (Ultra-Mobile PC). Most of these mobile devices equipped with Wi-Fi, WiMAX or other wireless modules making people be able to access services anywhere. However, different radio technologies and protocol standards need to be realized through different IC (Integrated Circuit) chips. How to integrate the various protocols and radio frequency (RF) chips into a small device is the most important challenge in recent years. Therefore, there is a design trade-off between the application variety and the size minimization of user device.

The traditional hardware radio system consist a variety of analogy elements such as filters, converters, modulators and demodulators. The hardware is expensive in cost and low compatibility with other components. The reason why Software Defined Radio (SDR) becomes popular is that people could use SDR technology to realize many applications without a lot of efforts in the integration of different components. We can change the different software module to adapt different modulators and demodulators in the SDR platform. The most radio and wireless related applications could be achieved.

Users can use SDR on personal wireless device. For example, the vendors could integrate GSM (Global System for Mobile Communications), WCDMA (Wide band Code Division Multiple Access), GPRS (General Packet Radio Service), IS-95, EV-DO, Wi-Fi, WiMAX or Bluetooth in a single device and update the newest radio modules by download software modules. In the military applications, such as U.S. DoD Joint Tactical Radio Sys-
tem (JTRS) program [1] develops a military radio communication device which supports more than 20 different communicational standards. In an emergency situation, the gateway device based on SDR could be used to bridge various types of incompatible radio equipments or establish a temporary communications infrastructure through SDR equipment.

Therefore, the goal of the user device development is to minimize its size, decrease the number of ASIC chips but keep more radio applications. To achieve the software radio functions, the base protocol of software modules, ADC/DAC conversion of hardware radios and multi-band antennas are necessary. Although the concept of SDR has been proposed for a long time, the implementation was stuck due to the insufficient technology until recent years. Most products and research developments of SDR focus on the voice transmission. This paper utilizes Universal Software Radio Peripheral (USRP) [2] and GNU Radio [3] to implement a reconfigurable SDR platform which can support digital communications and wireless spectrum sensing.

This paper is organized as follows. We introduce the available SDR resources and our platform, including the design of GNU Radio and the architecture of USRP in section 2. The radio spectrum sensing and the reconfigurable digital communication, the implementation and experiment results are showed in section 3. Conclusion are finally drawn in section 4.

2. Background

2.1 Software Defined Radio

Traditional hardware radios are implemented with analog and solid poly-Si elements. In SDR, the traditional hardware is replaced by software modules such as Figure 1. SDR was proposed by Joseph Mitola in the beginning of 1990 [4]. Unlike adopt Application Specific Integrated Circuit (ASIC) to implement radio elements in the past, the technologies such as Field Programmable Gate Array (FPGA), Digital Signal Processor (DSP) and General-Purpose Processor (GPP) are used to build the software radio elements. These components have reconfigurable capability which making these components tend to generalization in order to implement a variety of different radio applications.

The fundamental architecture of SDR is shown in Figure 2. It includes front-end, processing engine and application. The Radio Frequency (RF) front-end module digitizes the radio frequency data from antennas. After the baseband is digitized by front-end, the processing engine converts baseband data and date frames. The application side receives data frames at last.

2.2 USRP

Universal Software Radio Peripheral (USRP) was designed by Matt Ettus [2]. It was combined with radio front-end, Analog to Digital Converter (ADC) and Digi-
tal to Analog Converter (DAC) via Universal Serial Bus 2.0 (USB 2.0) on GPP platform. According to the statements as mentioned above, the USRP is available to realize a reconfigurable and adaptable SDR.

Figure 3 shows the components on USRP motherboard. The 4 ADCs which can sample $60 \times 10^6$ times per second on each ADC, and 4 DACs which samples $128 \times 10^6$ times per second on every DAC. Additionally, there are one Altera Cyclone EP1C12 FPGA chip and one programmable Cypress FX2 USB 2.0 controller on the USRP motherboard.

In USRP, the block diagram as shown in Figure 4 represents whole work flow and function components. It can be divided into two parts based on the transmission path. There are transmitting signal path and receiving signal path. For example on transmit signal path, users can define the setting parameters by software on personal computer such as radio protocols, modulation types, frequency of spectrum modulation. Then the USRP receives the parameters, and FPGA executes Intermediate Frequency (IF) processing on Digital Up Converter (DUC) and Digital Down Converter (DDC). After IF process, users adjust the baseband to the frequency band selected before. The last step on USRP motherboard is that DAC converts the digital signal into analog signal. Finally, the analog signal is transmitted to the antenna through the interface side A or side B on the daughterboard. According to the above procedure, we can confirm that one per-

Figure 3. USRP motherboard.

Figure 4. USRP block diagram.
sonal computer with one USRP hardware device achieve SDR’s goal certainly.

Table 1 shows the current commercially available daughterboards of USRP which equal to the radio front-end in SDR. The basic daughterboards such as BasicRx and BasicTx are the entrance and exit of signal. There is no mixer, filter or amplifier on them. There is only transmit or receive single functionality, these two daughterboards can’t be serving as a transceiver. The RFX series daughterboards can transmit and receive at the same timeslot, so they always treated as MIMO tools. In this paper, we select RFX2400 to be our daughterboard in the reconfigurable SDR platform.

Figure 5 shows RFX2400 daughterboard that we utilized in this paper. RFX2400 daughterboard supports the frequency band from 2300 MHz to 2900 MHz. This frequency coverage includes the Industrial, Scientific and Medical (ISM) band. Thus, RFX2400 daughterboard can be implemented as an IEEE 802.11 transceiver/receiver. RFX2400 daughterboard structures with the phase modulator AD8347 and phase demodulator AD8349. Throughout these two analog chips, the signal can be directly converted to baseband without any intermediate stage conversion. It is divided into two stages. First of all, the daughterboard will try to tune its central frequency to the desired frequency, then DUC and DDC will structure on different frequency bands after tuning the frequency of daughterboard. This feature is the tunable ability of RFX series daughterboards.

### Table 1. List of USRP daughterboards

<table>
<thead>
<tr>
<th>Name</th>
<th>Functionality</th>
<th>Spectrum (MHz)</th>
</tr>
</thead>
<tbody>
<tr>
<td>BasicRx</td>
<td>Rx</td>
<td>2–300</td>
</tr>
<tr>
<td>BasicTx</td>
<td>Tx</td>
<td>2–200</td>
</tr>
<tr>
<td>LFRX</td>
<td>Rx</td>
<td>0–30</td>
</tr>
<tr>
<td>LFTX</td>
<td>Tx</td>
<td>0–30</td>
</tr>
<tr>
<td>TVRX</td>
<td>Rx</td>
<td>50–70</td>
</tr>
<tr>
<td>DBSRX</td>
<td>Rx</td>
<td>800–2400</td>
</tr>
<tr>
<td>RFX400</td>
<td>Tx/Rx</td>
<td>400–500</td>
</tr>
<tr>
<td>RFX900</td>
<td>Tx/Rx</td>
<td>800–1000</td>
</tr>
<tr>
<td>RFX1200</td>
<td>Tx/Rx</td>
<td>1150–1450</td>
</tr>
<tr>
<td>RFX1800</td>
<td>Tx/Rx</td>
<td>1500–2100</td>
</tr>
<tr>
<td>RFX2400</td>
<td>Rx/Rx</td>
<td>2300–2900</td>
</tr>
</tbody>
</table>

2.3 GNU Radio Toolkit

GNU Radio [3] is an open source project for SDR Software Development Kit, the original version was proposed by Eric Blossom and a group of developing research and design team from A.D. 2000. Its features include mixed programming languages, high performances, fixed changeable blocks, reconfigurable, Graph User Interface (GUI) and variety of sink and block modules. It is easy to integrate with USRP for realizing the reconfigurable SDR.

GNU Radio was original designed to use in GPP. It combined with less number of analog radio devices, generated signal waveform, modulation methods, signal processing and communication protocols through software radio. The signal processing database in GNU Radio includes the most of signal processing functions, such as signal waveform modulations and most kinds of filters. Figure 6 shows the architecture of GNU Radio, the program for GNU Radio platform is written in a higher level of language – Python, then combining the lower
level C++ program. The signal processing blocks in GNU Radio are written in C++ and the graph constructions are written in the Python. Simplified Wrapper and Interface Generator (SWIG) is a middle interface between Python and C++. Python access the C++ signal processing blocks through interfaces generated by SWIG for Python, and establish a connection with USRP through USB 2.0. In terms of this architecture, GNU Radio integrates the USRP hardware devices to realize that using software to define the radio settings.

There are many practical modules in GNU Radio as follows:

(1) GPS receiver – DBSRX daughterboard receives Global Positioning System (GPS) signal and can be integrated with Google Earth interfaces to a complete GPS receiver.

(2) DVB-T receiver – using Basic series daughterboard as receiver to fetch Digital Video Broadcasting-Terrestrial (DVB-T) signal, then we can watch digital television channels through GNU Radio and USRP.

(3) FM receiver – To receive the FM signals, we can use BasicRx daughterboard to demodulate and filter the FM signal by modules in GNU Radio. Finally, the radio content in FM channel can be played out by GNU Radio on the personal computer side.

Except the three modules as mentioned above, there are many practical modules but still under development, such as:

(1) BBN 802.11 receiver [5] – The BBN 802.11 receiver which sponsored by the team of Adaptive Cognition-Enhanced Radio Teams (ACERT) in the Defense Advanced Research Projects Agency (DAPRA). This project implemented a fundamental 802.11 transmitters and receiver which are able to decode low rate 802.11 packets.

(2) Utah Sensing and Processing Across Networks (SPAN) 802.11b receiver [6] – This receive module was developed by the university of Utah Sensing and Processing Across Networks (SPAN). The main contribution has been to implement the full-rate despreading operation in the Altera FPGA on USRP. This project allowed reception of 802.11b signals from more distant transmitters and from devices transmitting a higher data rate.

(3) Bluetooth receiver – It was developed by the department of Computer Science at University College London. This project was set out to implement some functions of the Bluetooth protocol by using GNU Radio and USRP to replace the Bluetooth hardware. The developers implemented the Bluetooth protocol stack for SDR and this module could be able to sniff and inject packets.

(4) UCLA Zigbee receiver – This module was developed by University of California, Los Angeles (UCLA). The physical layer and Media Access Control (MAC) layer modules on IEEE 802.15.4 protocol could be monitor and debug by this project.

According to these completed or being developed modules, it is obvious to understand that the evolution of SDR would be popular in the future.

2.4 GNU Radio Companion

GNU Radio Companion (GRC) [7] is a signal flow chart generator tool in GNU Radio. The interface of GRC is shown as Figure 7; the flow chart represents an example code for FM radio receiver. We could build a signal flow chart through this GUI tool and also review the source code to map this flow. Each block has a relative parameter XML file, GRC will automatically identify the block’s definition when it is executing. In other words, GRC has the automatic recognition error ability.

The properties of every block can be adjusted through GRC design tool. For instance, Figure 8 shows a FFT sink setting in the FM radio receiver.

After setting the relevant block attributes and de-
signing the signal processing flow, we can start the program which mapping to the Python source code through GRC’s executive function. As shown in Figure 9, there is the graphical interface after executing the FM receiver module by GRC. It can easily adjust the signal processing flows and re-modify the signal processing modules and attributes through GRC tool. We can easily reconfigure the SDR through this function.

3. Accomplish Digital Data Communication

We connect the USRP hardware device with the GNU Radio software tool, and then we use software to define the parameters about radio communication. After setting up the connection between two SDR platforms, we utilize the spectrum sensing ability to detect the free spectrum bands in the environment. The tunable feature can reconstruct the connection at a spectrum hole, so that we can use the finite spectrum to accomplish digital data communication. In the ultimate aim, we attain to enhance the spectrum utility rate by reusing the limited spectrum resource efficiently.

3.1 Spectrum Sensing

Federal Communications Commission (FCC) defines the usage of spectrum allocation explicitly. There is a serious impact on the emerging multi-media applications because it’s unable to use the limited spectrum resource efficiently. Cognitive radio [8–10] is a popular technology due to it is based on software to define the wireless sensing techniques. Thus, it is able to enhance the spectrum utility rate evidently. The concept of Cog-
nitive Radio is originated from: 1) radio sensing and learning; 2) recognizes and allocates spectrum opportunity; 3) realizes spectrum opportunity. And CR utilizes the intelligent sensing method to acquire the spectrum usage information and environment parameters then chooses the most feasible network or the spectrum reconfigurable network architecture. Therefore the spectrum sensing ability in SDR is extremely important in order to accomplish the goal of digital data communication.

In this paper, we implement the spectrum sensing program in GNU Radio and integrate it with GNUPlot function in order to present the spectrum utilization in graphics. The GNUPlot function draws the spectrum sensing information through Python language, thus it can detect the spectrum holes by the results of spectrum usage. The range of our spectrum sensing is from 2.397 GHz to 2.479 GHz, which is the standard range of IEEE 802.11 standard, and the allocation of IEEE 802.11 channels’ plan is shown in Table 2.

In order to verify our spectrum sensing function, we use a laptop to get the wireless network usage authority from National I-Lan University. The Access Point (AP) in our environment is allocated at Channel 6, and we download a file from Internet through this AP. After that, we take another laptop which equips with our SDR platform, which includes USRP motherboard, RFX 2400 daughterboard, GNU Radio and our spectrum sensing program. We can analyze the spectrum usage through this platform. As Figure 10 shows, the range of red frame is Channel 6 in the IEEE 802.11 standard, and it is known that Channel 6 is allocated from 2.422 GHz to 2.452 GHz in IEEE 802.11 from Table 2. When we start to download the file, the Power dB value will rise distinctly. We can comprehend that Channel 6 is used from this phenomenon.

3.2 Digital Data Communication

The environment of our digital data communication is shown as Figure 11. Both two laptops are equipped our SDR platform and Linux based operating system (OS). The two laptops are named Host A and Host B respectively. The two hosts connect to the USRP devices through USB 2.0 and GNU Radio provides a simple Media Access Control (MAC) layer example to do the data connection. Finally, the virtual interface TUN/TAP [12] in our laptops will configure the USRP device as a virtual network interface, and then connect to another host through IP protocol to achieve the digital data communication experiment.

The parameters allocation to the digital data transmission device is shown as Figure 12. We use -m or

Table 2. The spectrum allocation of 802.11 channel plan

<table>
<thead>
<tr>
<th>Channel ID</th>
<th>Central Frequency (GHz)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2.412 (2.397 GHz–2.427 GHz)</td>
</tr>
<tr>
<td>2</td>
<td>2.417</td>
</tr>
<tr>
<td>3</td>
<td>2.422</td>
</tr>
<tr>
<td>4</td>
<td>2.427</td>
</tr>
<tr>
<td>5</td>
<td>2.432</td>
</tr>
<tr>
<td>6</td>
<td>2.437 (2.422 GHz–2.452 GHz)</td>
</tr>
<tr>
<td>7</td>
<td>2.442</td>
</tr>
<tr>
<td>8</td>
<td>2.447</td>
</tr>
<tr>
<td>9</td>
<td>2.452</td>
</tr>
<tr>
<td>10</td>
<td>2.457</td>
</tr>
<tr>
<td>11</td>
<td>2.462 (2.447 GHz–2.479 GHz)</td>
</tr>
</tbody>
</table>
modulation to define the modulation type directly, CPM, D8PSK, QAM8, DPBSK, GMSK are the modulation types supported by GNU Radio. The default modulation type is GMSK.

From the spectrum sensing results, we could make a decision that Channel 11 (2.462 GHz) is an unused spectrum hole and it is accessible. Then we try to establish the connection between host A and host B at 2.462 GHz. Figure 13 shows host A’s configurations such as transmission frequency, name of daughterboard, transmission amplitude, modulation type, received gain and so on.

When we start the connection, the TUN/TAP will configure a virtual interface named gr0. Then we use “ifconfig” command to configure IP address for each host. For instance, Figure 14 shows the terminal of host B. In our experiment environment, we configure the gr0 interface to host B in 192.168.0.2, and we observe the variation number of transmission and reception. We can discover that the digital data communication has started through our SDR platforms.

As shown in Figure 15, after the communication established between two hosts, we can observe the payload length of the received packets in each terminal.

Finally, we utilize Wireshark to capture the transmitted and received packets from gr0. As Figure 16 shows, the Internet Control Message Protocol (ICMP) ping request sends from host A and the ICMP ping reply from host B work correctly. It means the connection has been established, and we can start to communicate between two hosts.

---

**Figure 10.** The spectrum usage status in 2.4 GHz.

**Figure 11.** Set up the experiment environment.

**Figure 12.** The parameter configuration of USRP device.
Figure 13. Execute the digital communication equipment.

Figure 14. Configure the virtual interface gr0.

Figure 15. Debug mode.

Figure 16. Utilize Wireshark to capture the packet content.
Through our SDR platform, which consists of GNU Radio software and USRP hardware device, we can dynamically adjust the central frequency of the digital data communication service and choose the unlicensed band as long as we want. Because GNU Radio provides high instantaneous and accurate spectrum sensing ability, we can efficiently utilize SDR to achieve digital data communication under the current limited spectrum resource.

Comparing with existing military SDR application [1] or other commercial product, most of them focus on analog communication and voice transmission. In this work, we presented a SDR platform based on available open source software and hardware. We also discussed some digital data communication application to fulfill the requirement of cognitive radios.

4. Conclusion

We proposed and implemented a reconfigurable SDR platform by combing USRP and GNU Radio. Furthermore we realize digital data communication by applying SDR applications. In the scarce radio spectrum resource, we perform spectrum usage sensing at first. Then we find available spectrum holes to establish the digital data communication link to transmit the digital data. From the steps above, we can avoid the radio interference, which causes lower transmit performance, and provide an efficient wireless digital communication.

We also use the flexible feature of SDR to switch the communicational spectrum in our research. When the primary user or interference appears in the current frequency, we reconfigure SDR to do spectrum hopping. Thus, the spectrum band is feasible in our environment. In order to enhance spectrum utilization, we can adjust the working frequency to other available spectrum bands through SDR’s reconfigurable capability. These functions are necessary conditions to accomplish the popular cognitive radio.

Acknowledgement

This research was partly funded by the National Science Council of the R.O.C. under grants NSC 98-2219-E-197-001 and NSC 98-2219-E-197-002.

References


Manuscript Received: Jan. 8, 2010
Accepted: Feb. 22, 2010