Wideband RF Front End Design Considerations for a Flexible White Space Software Defined Radio

S.M. Shajedul Hasan, Randall Nealy, Terrence J. Brisebois, Timothy R. Newman, Tamal Bose, and Jeffrey H. Reed

Wireless @ Virginia Tech, Virginia Polytechnic Institute & State University, 432 Durham Hall (0350), Blacksburg, VA 24061

Abstract — **This paper introduces a flexible RF front end for whitespace communication. The designed front end can operate over any frequency from 100 MHz to 2.5 GHz and the channel bandwidth can be programmable from 4.5 kHz to 10 MHz. This large frequency range and wide bandwidth makes this hardware suitable for implementing most wireless standards. A direct conversion RFIC developed by Motorola, drives the core of the RF front end. The various RF parameters can be changed by programming this RFIC through a serial peripheral interface (SPI). As part of this work we further develop an intelligent software driver to control different parameters of the RFIC. Thus the combination of highly flexible front end and flexible software driver makes this hardware an excellent choice for whitespace devices. The performance of this front end has been tested and measured and has been integrated into a daughterboard format for the Universal Software Radio Peripheral (USRP), a hardware device which enables the rapid design and implementation of software defined radio (SDR).**

Index Terms — **Radio transceivers, CMOS, Direct Conversion, SDR, Transceivers.**

I. Introduction

The availability of unlicensed whitespace spectrum opens up a new door for wireless communication. However, this new opportunity also brings new and more difficult requirements for whitespace devices which should be more cognitive in nature. Not only should they be able to free up the space if a primary user arrives, but also dynamically adjust various parameters such as gain, bandwidth, and sidelobe attenuation levels in order to perform efficient interference management subject to the radio's operating condition and surroundings. Therefore, flexible hardware is needed along with software reconfigurability. Historically it is always difficult to achieve a flexible RF hardware which can operate over large tuning ranges with the capability of variable bandwidth.

In this paper we introduce a flexible RF front end which is designed using a Motorola SDR RFIC [1]. Figure 1 shows an image of this RFIC board. This front end is developed to work as a daughter board for USRP [2], which is a popular hardware platform for rapid prototyping of software defined radios. The software driver for this board is developed in the GNU Radio environment, which is also an open source software architecture for implementing SDR [3]. The combination of flexible RF front end, USRP, and GNU Radio architecture makes this hardware platform a perfect choice for the implementation of a whitespace software defined radio. Any frequency can be selected between 100 MHz to 2.5 GHz and the channel bandwidth can be varied from 4.5 kHz to 10 MHz in approximately 10% steps. Furthermore, it is possible to implement most of the wireless standards in the GNU Radio software environment.

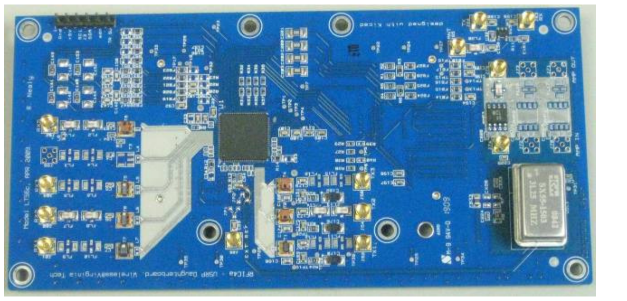

Fig. 1. Image of the designed RF front end board.

This paper is organized as follows. Section II provides a brief description of the Motorola RFIC and Section III describes the hardware design. Section IV presents the description of the software driver to operate and control the various parameters of the board. Section V briefly discusses the application of dynamic spectrum access using this board. This paper is concluded in Section VI.

II. DESCRIPTION OF THE SDR RFIC

The Motorola RFIC is a direct conversion based up/down converter, which contains five receive inputs and three transmit outputs, and requires a single source of external reference frequency (e.g., 31.25 MHz) for its operation. Only one receive input and transmit output can be selected at a time. Independent I-Q baseband signal paths are provided for each direction. Analog baseband active

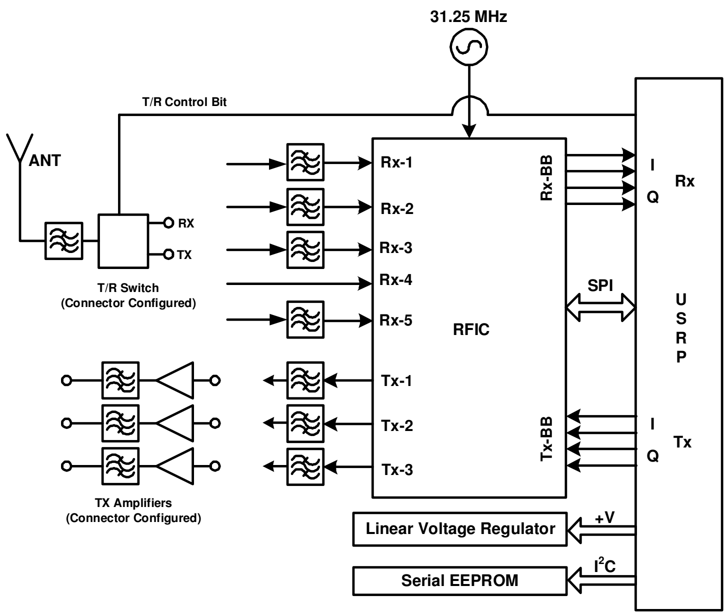

Fig. 2. Overall block diagram of the RFIC board.

filtering is also provided for each path. All the parameters of this RFIC can be controlled by programming some internal registers through a SPI. Receiver baseband filter bandwidth and gain is also controlled by register settings. Bandwidth is variable and can be selected as any value between 4.5 kHz and 10 MHz in approximately 10% steps.

Traditionally direct conversion topology suffers from many design issues such as DC offset, LO phase noise, even-order distortion, and flicker noise [4]. However, most of these problems have been mitigated or reduced by employing chopping mixer and using a differential topology in the Motorola RFIC. Prior to using this RFIC in our design, we evaluated it using several stand alone evaluation boards and found performance to be generally consistent with specifications published by Motorola [5]. We also tested it by designing a complete multiband multimode radio for public safety applications [6].

III. HARDWARE DESIGN

An overall block diagram of our RFIC board is shown in Figure 2. RFIC obtains RF signals from the receive inputs. The signals are amplified, filtered, and down-converted to baseband inside the chip. They are then sent to the USRP motherboard via the baseband receive outputs shown on the right side of the figure. Transmitting a signal works in the opposite direction. The USRP sends baseband signals to the RFIC through the baseband transmit input. These signals are transmitted through one of the three RF transmit outputs after amplification, filtering, and up-conversion to the carrier frequency.

Since receive inputs and transmit outputs of the RFIC require or produce differential signals, transformers have been used to convert single ended signal to differential and vice versa. RX-1 to RX-3 use 1:1 impedance ratio ferrite core baluns with a frequency range 5 to 3000 MHz. RX-5 and TX-1 to TX-3 uses a 4:1 transformer with a frequency range of 500 to 2500 MHz. Note that the tuning range of this board is limited by the bandwidth of these baluns.

A 31.25 MHz reference oscillator is provided on the board for the generation of all the internal LO signals in the RFIC. The RFIC RX baseband output signal lines are DC coupled to the RX input of the USRP. Since the USRP TX output requires differential current instead of differential voltage, it is necessary to tie a resistor from each side of the USRP TX output to ground to make the RFIC TX work with the USRP. Level shifters are also required for the logic signals and the SPI control interface.

Three independent power amplifier/filter sections are provided on the board for TX ports. In order to use the amplifiers, it is necessary to use a MMCX cable to connect the TX output to the amplifier input. The amplifiers are intended to be used at frequencies between 500 and 2500 MHz. Power levels on the order of 100 mW may be obtained. A T/R RF switch is also provided on the board for use up to 2500 MHz. The state of this switch can be controlled from the USRP.

The antenna port of the RF switch is provided with a low pass filter to suppress harmonics generated in the circuits. Moreover, each TX and RX port is provided with footprints for a pair of 1206 size LTCC filters in cascade. Individual filters can be installed to cover the desired frequency ranges.

The circuit board has been fabricated on the four layers PCB. The uppermost layer is the primary routing layer containing all RF components and traces. The middle two layers are ground and power planes respectively. Control and voltage regulator circuits are located on the bottom layer along with secondary routing for various functions.

IV. SOFTWARE DEVICE DRIVER

A software driver has been developed to control the parameters and operation of this board in the GNU radio environment using SPI, which is a standard interface by which two or more devices can transmit and receive data and instructions with one another. There is a "master" chip, in this case the FX2 USB controller on the USRP, and a "slave" chip, in this case the RFIC. The Motorola RFIC has approximately 262 8-bit registers, each of which may contain up to 8 variables which influence the operation of the chip. The master must write the contents of these registers, controlling each variable precisely, in order to ensure proper operation of the RFIC. This is how the software driver controls the RFIC.

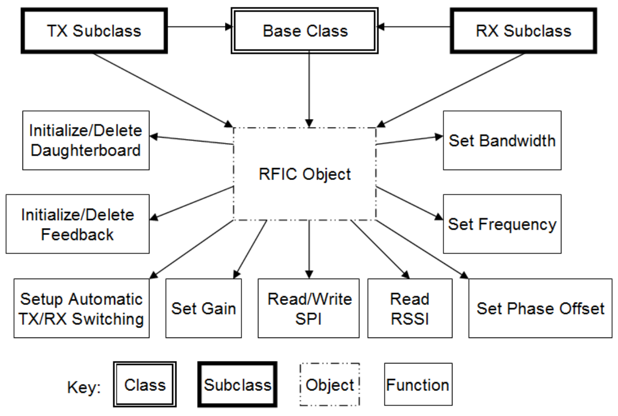

Fig. 3. Software driver block diagram.

Figure 3 shows the block diagram of the software driver. The actual operation of the driver requires one class, two subclasses, and one object. TX and RX subclasses are used when the daughterboard is transmitting and receiving, respectively. Whichever one is in use creates the base class. The base class, in turn, sets up the RFIC object. The RFIC object sets up the RFIC chip itself. It sets registers on the RFIC and variables and parameters within GNU Radio for general purpose operation. The TX and RX subclasses set up the RFIC for transmitting or receiving. As shown in the diagram above, the RFIC object contains nearly all of the operating functions of the chip. The TX and RX subclasses call these functions in order to set the radio parameters.

This software driver configures various parameters such as setting center frequency, gain, bandwidth, phase offset, and RSSI. The user can easily set the bandwidth of the transmit, receive, or feedback sections of the RFIC. By simply calling a function, and including the desired bandwidth, the bandwidth function in the driver will set the bandwidth of the transmitter and receiver. Phase offset can be set anywhere from 0 to 360 degrees, in any of the three RF paths. The RSSI, or received signal strength indicator, is another useful tool. It can show the user how strong the received signal is, which is useful for automatic gain control (AGC). The RF performance can be improved by optimizing many variables in the RFIC registers. A detailed description of this software driver can be found in [7].

V. DYNAMIC SPECTRUM ACCESS USING VT CROSS

To demonstrate the capabilities of the RFIC board in a more practical scenario, we've developed a Dynamic Spectrum Access (DSA) application that is able to operate within the flexible limits of the daughterboard. Developed using the GNU Radio [3] software framework, we have implemented a simple DSA protocol that could be used by secondary users in a next-generation wireless network to avoid primary users and rendezvous without using a control channel.

The primary users were tested on signals emitted from standard two-way Motorola developed radios operating in the FRS band. While this specific testing environment can not demonstrate the ultimate flexibility of the radio hardware, we found this configuration an extremely easy way to test the reconfigurability of the radio using COTS equipment. Note that it is only a small software change to configure the operational frequency channels that this application uses to communicate; in no way does the radio hardware restrict these tests to only FRS channels.

The software application observes energy on the channel and checks if it is greater than the pre-determined threshold. If a primary user has been detected, the DSA radio nodes must depart the current channel and rendezvous on another channel, without any control side channel to exchange information about where they are going next. Independently, each node in the DSA network makes the decision about the next channel.

The rendezvous protocol that is used by the DSA network nodes is a beacon-based protocol similar to the CIREN protocol developed for the SDR Forum Smart Radio Challenge [8]. In this design, the nodes hop over a set of channels sending identification beacons. When two nodes hop into the same channel, one of the nodes will observe the other nodes' beacon signal and reply back with an acknowledgement. At this point the nodes have rendezvoused and can begin communication. Originally designed for a two node network, we have adapted the protocol to include a master/client node relationship in order to enable a larger networks of nodes.

The determination of the next channel is implemented using two different methods and is configurable as a command line parameter. This channel selection function is executed any time a DSA node needs to change channels. This can happen if a node is networked with another node and a primary user signal is detected or if a node is in rendezvous mode and has sent out a beacon and has gotten no response. The first method we use is the most trivial, a random access method. To determine the next channel, a channel, not equal to the current channel, is selected at random. Once the node is tuned to another channel, the energy is sensed to determine if a primary user is present and if no signal is present a beacon is sent, otherwise another channel is selected and this process continues.

The second and much more complex way of determining the appropriate channel to use involves the Cognitive Radio Open Source System (CROSS) developed at Virginia Tech [9]. The CROSS architecture is a modular cognitive radio system framework that provides portability and interoperability between components that may be individually developed even in different programming languages. A CROSS cognitive engine component was developed for this application that uses the average amount of effective communication time as an input to a fitness function. The cognitive engine selects a channel in a probabilistic manner, with channels having higher fitness scores having a higher probability of being chosen. Using a probabilistic approach avoids a scenario where the same channels are constantly being used. It allows the radio to explore channels, even if they haven't been the best in the past.

The GNU Radio DSA application and the CROSS architecture are both openly available to test and supplement development, as they use an open source license.

VI. Conclusion

One of the major objectives of this work is to develop a flexible RF front end for implementing wideband whitespace software radio using direct conversion based RFIC. We have successfully designed and implemented the board and tested the performance demonstrating a simple dynamic spectrum access protocol using the USRP and GNU radio software architecture. Although overall performance of this RFIC is satisfactory, we have found some issues, specifically image rejection and selectivity, which need to be improved. All the experiments were done using default parameters. Optimization of programmable parameters in the RFIC should be executed to improve the performance. A good user interface is also necessary to control this board efficiently.

Acknowledgement

The authors wish to thank the Motorola Research Laboratories for providing RFIC chips as well as extensive and patient technical assistance. This research work is supported by the Laboratory for Telecommunication Sciences (LTS) and Army CERDEC.

References

[1] G. Cafaro et al., "A 100 MHz–2.5 GHz Direct Conversion CMOS Transceiver for SDR Applications," in *Proc. IEEE Radio Frequency Integrated Circuits (RFIC) Symposium*, Honolulu, Hawaii, June 2007.
[2] USRP, [Online] Available: www.ettus.com
[3] GNU Radio, [Online] Available: www.gnuradio.org
[4] B. Razavi, *RF Microelectronics*, 1st ed. Upper Saddle River, NJ: Prentice Hall, 1997.
[5] S. M. Hasan, M. Harun and S. W. Ellingson, "Performance Evaluation of RFIC Ver. 4 in Public Safety Frequency Bands," Virginia Tech, VA, Tech. Rep. 21, July 2007. [Online]. Available: http://www.ece.vt.edu/swe/chamrad/
[6] S. M. Hasan and S.W. Ellingson, "Multiband Public Safety Radio using a Multiband RFIC with an Multiplexer-Based Antenna Interface," in *Proc. SDR Forum Technical Conference*, Washington DC, 2008.
[7] T. Brisebois, "Wideband RF Front End Daughterboard Based on the Motorola RFIC," Masters Thesis, Virginia Tech, Blacksburg, Virginia, July 2009.
[8] C. R. Aguayo Gonzalez, J. Reed, "Dynamic Spectrum Access Assessment in Cognitive Radios", in *Proc. SDR Forum Technical Conference*, Denver, CO, 2007.
[9] Cognitive Radio Open Source System, Virginia Tech, [Online] Available: https://cornet.wireless.vt.edu.